QoS-Transit Services: End-to-End Quality of Service Control in the Internet Using Dynamic Pricing

By

Steven John Roy Shelford
B.Sc., University of Victoria, 2001
M.Sc., University of Victoria, 2003

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Computer Science

QoS-Transit Services: End-to-End Quality of Service Control in the Internet Using Dynamic Pricing

By

Steven John Roy Shelford
B.Sc., University of Victoria, 2001
M.Sc., University of Victoria, 2003

Supervisory Committee

Dr. Eric G. Manning (Departments of Computer Science and Electrical & Computer Engineering)

Supervisor

Dr. Gholamali C. Shoja (Department of Computer Science)

Supervisor

Dr. John A. Ellis (Department of Computer Science)

Departmental Member

Dr. Kin F. Li (Department of Electrical & Computer Engineering)

Outside Member

Dr. Linda A. Welling (Department of Economics)

Outside Member

Dr. Son T. Vuong (Department of Computer Science, University of British Columbia)

External Examiner

Supervisory Committee

Dr. Eric G. Manning (Departments of Computer Science and Electrical & Computer Engineering)

Supervisor

Dr. Gholamali C. Shoja (Department of Computer Science)

Supervisor

Dr. John A. Ellis (Department of Computer Science)

Departmental Member

Dr. Kin F. Li (Department of Electrical & Computer Engineering)

Outside Member

Dr. Linda A. Welling (Department of Economics)

Outside Member

Dr. Son T. Vuong (Department of Computer Science, University of British Columbia)

External Examiner

ABSTRACT

The Internet does not currently provide end-to-end Quality of Service (QoS) guarantees across multiple network providers. We demonstrate that networks, by using dynamic pricing, can provide end-to-end QoS guarantees for those applications that need it.

We propose the concept of QoS-Transit Services: a set of primitive services offered by an Internet Service Provider (ISP) in order to deliver packets with statistical performance guarantees within its network, using dynamic pricing. ISPs can choose their own pricing schemes, as long as QoS is guaranteed. Through simulation, we demonstrate that even simple pricing mechanisms can guarantee the advertised performance of a QoS-Transit Service. End-to-end QoS across multiple networks is achieved by using multiple QoS-Transit Services, from different ISPs, in sequence.

Since QoS-Transit Services are offered by ISPs to generate revenues, we determine how to allocate bandwidth among the services in order to maximize revenue, assuming that demand functions can be estimated. We propose the IterLP and IterGreedy heuristics to determine the optimal allocation of bandwidth on predefined paths. IterLP achieves

revenue close to 99% of the optimal solution, achieving this result quickly. IterGreedy achieves 90-95% optimality, but executes faster than IterLP.

Additionally, to determine the paths on which to route the QoS-Transit Services so as to maximize the ISP's revenue, we propose three heuristics with different specific advantages: Service Grouping, Iterative Bottleneck Avoidance, and Iterative Bottleneck Avoidance with Tabu. We demonstrate that Iterative Bottleneck Avoidance with Tabu achieves approximately 98% of an optimal solution. Route selection is also shown to be more important when fewer QoS-Transit Services are offered.

When demand functions cannot be adequately estimated, an ISP can use our Iterative Allocation Adjustment heuristic to find the optimal allocation of bandwidth for a set QoS-Transit Services. The heuristic achieves over 95% of the optimal revenue for an ISP. We then examine how better routes can be identified by valuing the links in the network to identify rerouting possibilities, or to identify paths for new QoS-Transit Services.

To receive a specified end-to-end QoS, customers may use concatenated QoS-Transit Services. We propose the *Hub Charging Model* to provide scalable charging, using *Brokers* as middle-men. Additionally, we propose the concept of Overlay ISPs: ISPs that provide QoS-Transit Services by controlling an overlay network. An *Overlay ISP*, a type of virtual network operator, can offer QoS over a large geographic area without universal ISP support for QoS-Transit Services.

Finally, we detail an architecture for offering QoS-Transit Services using existing protocol standards. By using a *Broker-ISP* as an intermediary between customers and the ISPs offering QoS-Transit Services, routing and charging complexities can be hidden from the customers. Additionally we describe how security, billing, metering, and policing can be achieved.

The competitive environment assumed within this dissertation is now emerging, with ISPs charging content providers in order to deliver content at a superior quality level to the ISP's customers. The time is therefore ripe for dynamic pricing to be deployed.

Table of Contents

List of Tables

List of Figures

Glossary of Abbreviations

AH	Authentication Header
BBLP	Branch and Bound with Linear Programming
BGP	Border Gateway Protocol
B-ISP	Broker-ISP
DiffServ	Differentiated Services
ECN	Explicit Congestion Notification
HCM	Hub Charging Model
HEU	Heuristic (a specific knapsack heuristic)
IAA	Iterative Allocation Adjustment
IACM	Interconnection Agreement Charging Model
IBA	Iterative Bottleneck Avoidance
IBA-Tabu	Iterative Bottleneck Avoidance with Tabu
IBX	Internet Business Exchange
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IPsec	Internet Protocol Security
IntServ	Integrated Services
ISP	Internet Service Provider
IX	Internet Exchange
LSP	Label Switched Path
LSR	Label Switched Router
MMKP	Multiple-Choice Multidimensional Knapsack Problem
MPLS	MultiProtocol Label Switching
O-ISP	Overlay ISP
OIR	Overlay ISP Router
PG	Pricing Gateway
QoS	Quality of Service
RED	Random Early Detection
RSVP	Resource Reservation Protocol

SG	Service Grouping
SLA	Service Level Agreement
TCP	Transport Control Protocol
UDP	User Datagram Protocol
ULT	Utilization Level Tatonnement

Glossary of Variables and Functions

α	effective bandwidth
β	length of a probability interval
γ	maximum percentage change of a price
ζ	maximum packet loss
ξ	price elasticity of demand
π	producer surplus (profit)
σ	maximum percentage change of a bandwidth allocation
τ	upper bound of a probability interval
ℓ	link
φ	average number of links per path
d	destination, or the bandwidth assigned by an optimal allocation algorithm
g	price radius of ULT, or the number of groups in the SG heuristic
k	number of paths, or a constant
l	latency, or the number of choices per group in the MMKP
m	number of links
n	number of QoS-Transit Services
p	price
q	QoS-Transit Service
r	a path (route)
s	source
t	time
v	value
w	weight
x	demand
x_{goal}	goal demand of ULT
y	supply
z	granularity of optimal allocation solutions, or a flow
$A_{q\ell}$	indicates if QoS-Transit Service q is routed over link ℓ

C	capacity
L	set of links
P	set of paths
Q	set of QoS-Transit Services
$\overline{R}$	path vector
S	set of upper bounds of a series of probability intervals
T	Tabu list
MC	marginal cost
MR	marginal revenue

Acknowledgement

I would first like to thank my supervisors for their support and direction. Dr. Manning, I especially appreciated your encouragement and our discussions while I explored my topic and eventually determined the focus of my underlying work. Dr. Shoja, your perseverance and guidance to strengthen my work is greatly respected. The time you give your students is beyond what should be expected, but your commitment is certainly noticed.

I would further like to acknowledge my supervisory committee. Dr. Welling, I value your sense of adventure for jumping into the unknown and for providing another point of view. Dr. Ellis, thank you for your attention to detail and your obvious algorithm expertise. Dr. Li, thank you for your insight and experience. I have learned much from each of you.

While I have poured a lot of time into my research over the last three years, it was my family and friends that kept me sane. Thank you, Emma, for your understanding and making my life so enjoyable. I would like to thank my mom, Joe, and Melinda for their continuous support during this immensely busy time. Thank you, Gill and Guy, for your wisdom, food, and the occasional scuba dive. And finally, I appreciate my friends for easing those often stressful periods of time.

Lastly, I would like to acknowledge the generous financial support of the Natural Sciences and Engineering Research Council (NSERC) of Canada for the Doctoral Postgraduate Scholarship and the University of Victoria for the University of Victoria Fellowship and the President's Research Scholarships.

1 Introduction

The Internet does not currently provide end-to-end guaranteed Quality of Service (QoS) guarantees across multiple network providers. We demonstrate that networks, by using *dynamic pricing*, can provide end-to-end QoS guarantees for those applications that need it. Additionally, we determine how Internet Service Providers (ISPs) can maximize their revenue if they use such *dynamic pricing*. *Dynamic pricing* is a form of pricing in which advertised prices may vary in time in order to affect the demand of a set of goods.

1.1 QoS Guarantees: Who Cares?

Emerging broadband technologies promise faster download times and support for emerging multimedia applications such as video-conferencing, telephony, and video-on-demand. Internet applications requiring QoS suffer from the lack of QoS guarantees provided by the current Internet. This lack of guarantees limits both the performance of current applications and the offering of new applications requiring QoS that can be developed, and charged for.

As a result of the dot-com bubble, large investments were made in backbones, resulting in massive over-provisioning. Hence, in 2003, the North American optical backbones were only about 10% utilized [1]. Only 11% of the links on the Sprint backbone (in early 2001) had a load of 30% or more for at least 10% of the time, and 69% of the links never reached a load of 30% [2]. However, IP traffic continues to double every year [1,3], and the future, by definition, is uncertain. Demand for traffic could expand through the increased use of current applications, the adoption of new bandwidth-intensive "killer" applications, or the offering of higher bandwidth connections to users [4]. In Japan, "there is a strong concern that... Internet backbone technologies will not be able to keep up with the rapidly growing residential traffic" [4].

In [1], Odlyzko recounts the growth rate in the demand for hard disks, in which demand grew at about 30% per year until the 1990s, 60-70% per year through the early 1990s,

100% per year through the late 1990s, finally returning to 60% in early 2000. This realization that future demand remains uncertain poses problems not only for lightly loaded backbones, but more so for the heavily used metropolitan networks.

A single wavelength within an optical fibre is capable of 40 Gbps (OC-768), with existing commercial equipment capable of transmitting data at 2.56 Tbps per fibre using 64 wavelengths [5]. If all the fibre within the backbone networks used such capable fibre, they would be very lightly utilized indeed. Upgrading networks is, however, financially expensive. New end equipment needs to be purchased, and new, higher quality, optical fibre may need to be installed as well. While this may be financially feasible for large backbones, which consist of a small number of links and serve a large number of users, the cost of smaller (metropolitan or access) networks make over-provisioning less practical [6(p.11)]. Regardless, depending on over-provisioning to achieve QoS guarantees makes the dubious assumptions that over-provisioning will persist indefinitely, and that it will be universal.

If we are to support and perhaps rely on applications with QoS requirements, then we need some means of QoS control not only for the present, but also for the uncertain future. QoS is unlikely to be guaranteed without admission or congestion control due to the folly of relying on over-provisioning, and what Garrett Hardin called the "tragedy of the commons" [7]. Goods available to everybody, or common goods, result in overuse because self-interest says to maximize one's own gain, irrespective of the cost to others. Hardin contends that we need to use "mutual coercion, mutually agreed upon" to prevent the tragedy of the commons. We must agree upon a set of rules for using a common good so that no individual can take undue advantage of it. Laws are an example of mutual coercion, where we have established a set of rules, with consequences for violators, to prevent such tragedies.

Flat fee pricing schemes, as are common with many residential and commercial Internet connections today, encourage behaviour leading to the tragedy of the commons. Users do not consider the effect of their traffic on others, in the form of congestion – resulting in additional delays and packet loss – and so they will attempt to maximize the value of

their flat fee, by sending as much data as is useful to them. "Ruin is the destination toward which all men rush, each pursuing his own best interest in a society that believes in the freedom of the commons" [7]. Usage and/or quality of service fees help us avoid the tragedy of the commons.

There has been ample research focused on the use of pricing as a means of congestion control in networks [8-16]. Charging is used as a means of traffic management and congestion control, and also to fund the use of any future expansion of the network. However, there exists little work [17-20] on the use of charging for congestion control across a set of autonomous interconnected networks, where ISPs can each deploy their own pricing schemes.

1.2 Problem Definition

The present Internet does not readily support Quality of Service (QoS) guarantees. Further, the network operators require financial incentives to provide such support. Most research in network pricing to improve QoS focuses on a single network [9-16]. To use these approaches on a set of interconnected networks would require extensive negotiation, making the world of interconnection agreements, among ISPs, even more complicated than it is now. Additionally, an ISP must be able to choose its own pricing schemes.

Many network providers support networks offering value-added services, as little money can be made selling commodity "bit pipes." Odlyzko advocates such a smart network, but with a very simple interface [21,22]. However, as Isenberg notes, "Today's optimization is tomorrow's bottleneck" [23]. Network architecture should remain as simple as possible so that it can be easily altered for the future demands of traffic. As a result of emerging bandwidth markets, networks appear to be moving in favour of the customer, towards cheap, commoditized, network services [24]. It is within these contexts that a pricing model needs to be devised.

Network owners are beginning to adopt alternate business models that involve charging to improve the delivery of content across their networks. Shaw Communications, a Canadian Cable Internet provider, levies an optional charge on users in order to prioritize third-party VoIP (voice over IP) traffic [25]. BellSouth is considering charging content providers a fee to improve the QoS of the content provider's traffic. For example, an Internet-based phone company could pay BellSouth a monthly fee to improve the quality of its calls on BellSouth's networks, or a video-on-demand provider could share revenues with BellSouth to deliver its video with a high quality [26].

It cannot be stressed enough that the Internet must allow the development of new applications that can be rapidly adopted, and which alter the way the network has been traditionally used [21,27,28]. Pricing frameworks must be simple and general enough to support these applications and must be able to adapt to the new ways the network is used. A recent example was the creation of peer-to-peer applications. No longer do the usual client-server traffic flows apply, but now a client may be expected to send as much data as it receives. The advent of peer-to-peer applications also boosted the amount of traffic a typical user generates, much as the way the popularity of the world-wide-web did in the mid 1990s.

Deterministic, or absolute, QoS can only be guaranteed through resource reservations. In this dissertation QoS guarantees are probabilistic, though near absolute guarantees are possible. As a result of not requiring deterministic QoS guarantees, network utilization can be higher and costs can be lower. We also assume that each user uses only a negligible portion of the load on any particular link. Similarly, we assume that users' traffic flows are independent.

In order to deliver user traffic with QoS guarantees, we propose the concept of *QoS-Transit Services*: a set of primitive services offered by an ISP in order to deliver packets with statistical performance guarantees within its network, using dynamic pricing. Network operators can choose their own pricing schemes, as long as QoS is statistically guaranteed. End-to-end QoS can be achieved by using several of these services, concatenated, perhaps from different ISPs.

1.3 Scope and Focus

The focus of this dissertation is to demonstrate how end-to-end QoS can be achieved using the dynamic pricing of QoS-Transit Services. We first examine the case of a single ISP, showing that even a simple pricing mechanism can provide probabilistic QoS, and determining how maximal revenues can be achieved by an ISP. We then demonstrate how QoS-Transit Services can be used across multiple ISPs, with emphasis on the scalability of charging and routing.

1.4 Outline

This dissertation is organized as follows:

- Chapter 1 provides motivation for our research and explains our objective.
- Chapter 2 surveys related work and introduces the required background material.
- Chapter 3 proposes QoS-Transit Services: a set of primitive services offered by an ISP in order to deliver packets with statistical performance guarantees within its network, using dynamic pricing.
- Chapter 4 defines the problems of optimal bandwidth allocation and routing of dynamically priced services, when demand functions are known, and subsequently proposes new heuristics.
- Chapter 5 proposes a heuristic for the optimal bandwidth allocation of dynamically priced services when no demand function is known, and then outlines a method of valuing the network to identify opportunities for rerouting.
- Chapter 6 demonstrates how end-to-end QoS can be achieved using QoS-Transit Services.
- Chapter 7 outlines an architecture for deploying QoS-Transit Services using existing technologies.
- Chapter 8 summarizes our contributions and proposes future work.

2 Background

2.1 Economics

A fundamental concept of Economics is that of *supply* and *demand*. This section outlines some of the primary theory as it relates to this dissertation and can be skipped by readers familiar with basic microeconomic theory.

The number of units of a commodity that a consumer demands can be conveyed through a *demand function*, $x(p)$, where p indicates the price of the good. For our work we use the simplifying assumption that the demand for a good is independent of the prices of other goods.[1] Further, we assume that the demand functions of consumers are downward sloping, indicating that consumers value their n^{th} unit of bandwidth more than their $n+1^{th}$ unit of bandwidth. The reader should take note that as prices decrease, demand generally increases, and when prices increase, demand generally decreases.

Similarly, for producers (network providers) we define a *supply function*. A producer's supply function for a particular good, $y(\overline{p})$, is the number of units of that good that the producer will produce given the price vector, $\overline{p}$, defining the price of all goods. The work in this dissertation concerns the supplying of bandwidth in the short-term, where link capacities are fixed. Therefore, $y(\overline{p}) \leq C$, where C is the capacity of the supplied link.

We can further define an *aggregate demand function*, which defines the overall demand for a commodity. The aggregate demand function is the sum of the demand functions of each of the consumers.

---

[1] While the assumption is not true, it simplifies our models, and we do not believe the assumption to affect our results.

The goal of consumers is to maximize their *consumer surplus*: the amount that a consumer is willing to pay for a bundle of goods minus the price they pay for it. Consider the demand function for a single good portrayed in Figure 2.1. The consumer is willing to purchase one item of the good for p_1, or 2 items at a price of p_2 each, or 3 items at a price of p_3 each. If the consumer purchases 3 units at a price of p_3, then she receives a net gain of $p_1 - p_3$ for the first unit and $p_2 - p_3$ for the second unit. Her total consumer surplus is $p_1 - p_3 + p_2 - p_3$.

Figure 2.1: Consumer surplus for the purchase of 3 items.

The area under the demand curve and above the given price indicates the consumer surplus that a consumer will achieve, as illustrated by the triangle a-p-b in Figure 2.2. The goal of a consumer is to maximize her total *consumer surplus* across all goods.

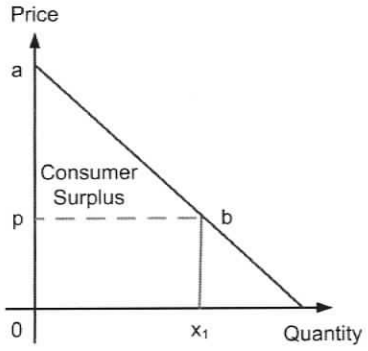


Figure 2.2: Consumer surplus of a good, defined by the area a-p-b.

Producers aim to maximize their total producer surplus (profit). Given a producer's *revenue function* $r(\bar{y})$ and *cost function* $c(\bar{y})$, where the vector $\bar{y}$ defines the number of supplied goods, the producer's maximization of *producer surplus* is defined as:

$$\pi = \max[r(\bar{y}) - c(\bar{y})].$$

For much of our work, the cost of supplying additional bandwidth is 0. A network owner often has a fixed link capacity, in the short-term, and thus the marginal cost (MC) of supplying addition bandwidth is:

$$MC = \begin{cases} 0, & y \le C \\ high, & y > C \end{cases}.$$

In the short-term, if additional capacity cannot be purchased, the producer simply maximizes her revenue, while respecting her capacity constraint. If we further assume that all units of a particular good are priced uniformly (non-discriminatory pricing), then the maximization of producer surplus can be simplified to:

$$\pi = \max[\bar{p}^T \cdot \bar{y}], \text{ such that } \bar{y} \le \bar{C},$$

where $\bar{C}$ is the vector of the producer's network link capacities.

A market is considered to be economically efficient when social surplus is maximized, where *social surplus* is the sum of the consumer and producer surpluses.

Another important concept is that of *marginal revenue*. *Marginal revenue* is the increase in revenue due to the sale of an additional unit of a commodity. For example, if we can sell 5 widgets at \$8/widget, but we can alternatively sell 6 widgets at \$7/widget, then our marginal revenue for the 6^{th} widget is:

\$7/widget×6widgets – \$8/widget×5widgets = \$2.

Additionally, a *marginal revenue curve* indicates the marginal revenue that is earned for each additional unit of a commodity that is sold.

Finally, *price elasticity of demand* is a measure of the percentage change in demand that results from a given percentage change in the price of a good. A good is considered *elastic* when a 1% change in price results in at least a 1% change in demand for that good. A good is considered *inelastic* when a 1% change in price results in less than a 1% change in demand for that good. Further, a good is considered *unitary elastic* when a 1% change in price results in a 1% change in demand. Additionally, we consider an application to be *adaptive* if it can vary the amount of bandwidth it uses over time, in response to price changes. *Non-adaptive* applications have strict bandwidth requirements, and thus cannot alter bandwidth demands due to price changes. For this dissertation we consider aggregated traffic which, as a whole, is *adaptive* and is assumed to not be extremely *inelastic*.

2.2 Congestion and Admission Control

Congestion control provides a means of controlling the amount of congestion in the links of the network. Mechanisms for congestion control may, as a secondary goal, provide a level of fairness in their allocation of bandwidth to flows.

An allocation may be considered fair if it satisfies max-min fairness. Max-min fairness is achieved when it is not possible to increase the allocated bandwidth of some flow without decreasing the allocated bandwidth of a smaller flow. Consider a network with a set of links, and a number of flows across the network. An example of an algorithm for determining max-min fairness follows [6(p.238)]:

1. Let all the flows start with zero bandwidth.
2. Increase the bandwidth of all non-fixed flows gradually, at the same rate, until a link reaches capacity.
3. Fix the flows that use that link and loop to step 2 until all flows have their allocations fixed.

Such an allocation is max-min fair because it is not possible to increase the bandwidth of any flow without decreasing the bandwidth of a smaller flow, while respecting all link capacities.

The Transmission Control Protocol (TCP) provides a means of per-flow congestion control, known as flow control, which aims to provide max-min fairness. TCP aims to discover free bandwidth on a flow's path, slowly increasing the rate of the flow, and then backing off when congestion occurs [29,30]. Routers implement Random Early Detection (RED) [31,32] to improve the efficiency of link use, by dropping TCP packets from random flows before congestion can occur, in an attempt to prevent congestion. The TCP flows that experience packet loss then react with exponential back-off as if congestion had occurred.

Congestion control algorithms like RED will not affect User Datagram Protocol (UDP) traffic as they do TCP traffic; UDP does not alter its transmission rate in the event of packet loss. Also, packets are dropped randomly, and thus latency-sensitive packets might be dropped from applications that are sensitive to packet loss. It has been proposed that Explicit Congestion Notification (ECN) be implemented, using 2 bits from the Type of Service field in the Internet Protocol (IP) header, where a Congestion Experienced bit is set as an alternative means of indicating congestion, rather than dropping packets [33].

Tom Kelly studied the use of ECN probe packets as a measure for connection acceptance control [34]. A connection is accepted if the number of probe packets that experience congestion is below a certain threshold. His method requires only a small change in routers, requiring them to mark packets for congestion before the onset of buffer overflow.

Other efforts have been made to improve the quality of IP traffic across the network. The two main efforts include Differentiated Services (DiffServ) [35] and Integrated Services (IntServ) [36]. DiffServ provides relative guarantees while IntServ tries to deliver absolute guarantees.

In DiffServ, traffic is categorized at an ingress router and scheduled according to its categorization at each subsequent router. There are two scheduling behaviours currently standardized: Expedited Forwarding and Assured Forwarding. In Expedited Forwarding, routers service packets at least as fast as they arrive. Assured Forwarding traffic is actually assigned to one of 4 classes, plus a drop precedence of high, medium, or low; thus, Assured Forwarding actually consists of a group of 12 scheduling behaviours.

IntServ uses the Resource Reservation Protocol (RSVP) [37] as a signaling protocol to support two traffic classes: Guaranteed Service and Controlled Load. Guaranteed Service, used for real-time, non-adaptive applications, guarantees maximum delay and bandwidth; though no guarantees are made for average or minimum delays [38]. Controlled Load, used for adaptive applications, provides behaviour similar to a best-effort service "under unloaded conditions" [39]. Admission control is implicit in achieving the above goals.

An overview of traffic engineering in today's IP networks is given in [40].

There has been considerable work done at the University of Victoria on optimal admission control. Khan initially developed the Utility Model for controlling admission to a multimedia server, formulated as a knapsack problem [41]. The model includes an admission controller that receives bids from users who want to access the server. The controller determines the amount of resources that each user requires and then rewards service to users such that revenue is maximized for the server and resources are not over-consumed. Watson subsequently formulated the problem for admission control in networks [42,43], and Akbar extended the solution for traffic that can span multiple networks [44,45].

2.3 Internet Architecture

The Internet consists of a large number of interconnected networks. The networks are owned by a variety of entities; the most relevant from a routing standpoint are the ISPs.

ISPs provide access to the Internet to their customers. The businesses and relationships of the ISPs vary depending on their size, as we will see later.

One or more networks form an *Autonomous System*. An *Autonomous System* (AS) "is a connected group of one or more IP prefixes run by one or more network operators which has a SINGLE and CLEARLY DEFINED routing policy" [46]. An ISP, thus, (usually) operates one or more interconnected ASes.

Each ISP gains value by having its networks connected to additional networks. Consider a small ISP that is not interconnected with any other ISPs. That ISP can provide little value to its customers, as its customers can only communicate with each other. The more networks with which a network is interconnected, the more destinations that are reachable, and thus, the more value the ISP can provide to its customers [47].

Currently, ISPs connect through bilateral interconnection agreements of two general types: *peering* and *transit* [48,49,50]. A *peering agreement* between two ISPs allows each ISP to send traffic destined to a customer of the other ISP, directly to that ISP's network, for no charge. Peering is either *direct*, in which case a direct physical connection exists between the two networks, or via *exchange point peering*, where it occurs through a common exchange point. Peering agreements are often formed between networks with similar traffic characteristics, if it is less costly to peer than to send traffic to the ISP through a *transit agreement* with that, or another, ISP. Peering agreements do provide advantages over transit agreements, including the potential to improve performance and cost [49].

A *transit agreement* between two network providers exists when one ISP provides another ISP with access to the entire Internet. The charging ISP accepts not only traffic for its internal customers, but also for traffic destined anywhere else on the Internet. The charging ISP must be able to deliver the traffic itself, or have agreements with other networks to deliver it.

We can classify ISPs as Tier-1, Tier-2, or Tier-3/Tier-4, as shown in Figure 2.3. There are no hard and fast rules on how to classify ISPs into these classifications. Tier-1 ISPs are often referred to as Internet Backbone Providers, as their networks make up the bulk of the Internet's core network. Tier-2 ISPs are often referred to as regional ISPs. Tier-3/Tier-4 ISPs are the remaining ISPs, which are smaller than the regional ISPs.

Tier-1 ISPs do not enter into transit agreements to deliver any of their data. Tier-1 ISPs enjoy a mutual benefit by peering with other Tier-1 ISPs to extend their reach. This relationship is shown in Figure 2.3, where each of the Tier-1 ISPs peers with the others.

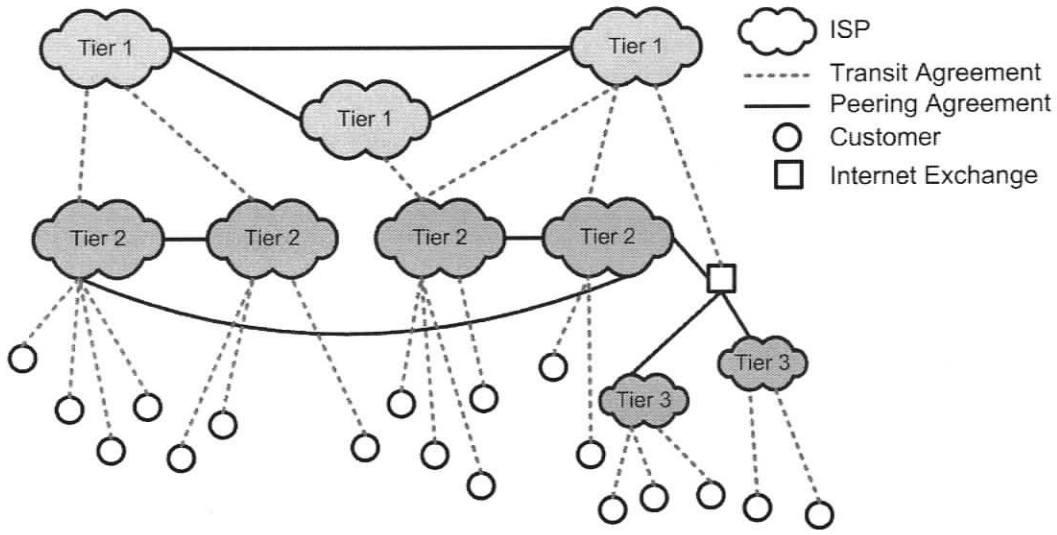


Figure 2.3: The hierarchical relationship of ISPs in the Internet.

Tier-2 ISPs may also form peering relationships with one another when economically attractive. However, to reach the entire Internet for their customers, they must engage in a transit agreement with at least one ISP. Tier-1 ISPs have no obvious motivation to peer with a Tier-2 ISP, as the Tier-2 ISP would receive a disproportionate benefit from the relationship.

Like a Tier-2 ISP, Tier-3 (and Tier-4) ISPs must connect with another ISP through a transit agreement. To reduce costs, Tier-3 ISPs can form peering relationships themselves, or aggregate their traffic to peer with a Tier-2 ISP. Such aggregation and peering is often done through an Internet Exchange, as illustrated in Figure 2.3.

There are a number of different types of *exchange points* [49]. An *Internet Exchange* (IX) is a facility where ISPs can physically connect, assisting them to join in peering relationships. A network access point (NAP) was one of the original IXs that were funded through the National Science Foundation. Such public IXs generally do not allow the connecting parties to engage in transit relationships. Internet Business Exchanges (IBXs) allow transit and peering relationships to exist. Besides ISPs, IBXs also encourage the providers of web caching, web hosting, content distribution, and other value-added services, to collocate their services at the IBX. Within this dissertation, we will refer to all exchange points as IXs, for simplicity.

Pan [51] studied the traffic characteristics at several ISPs to determine where bandwidth bottleneck links are, if over-provisioning was a valid solution, and whether a large percentage of end-users had traffic that traversed these bottleneck links. Pan observed that private peering links are often the congested links, and that most links within a network are not congested, except for transatlantic links. Pan argued that access links cannot be simply over-provisioned, due to the high cost of access links, many of which are leased from telephone companies. There was also implicit evidence, and a specific example, that some of these access links were deliberately made smaller, and more congested, to improve performance for the ISP's internal users.

Finally, Pan noted that the more ISPs a packet traverses, the higher the probability it will traverse a bottleneck link. A large provider, with a dense network, will have less inter-ISP traffic than smaller providers. However, large percentages of traffic from non-US customers are destined for US networks, and thus are more likely to cross bottleneck links [51].

Akella et al. [52] investigated where potential bottlenecks on the Internet were, by determining which links had less than 50 Mbps of available bandwidth. They determined that the number of potential bottlenecks was roughly evenly split between intra-ISP links and inter-ISP links. However, as there are more intra-ISP links, they are less likely to act as bottlenecks. Indeed, 36.5% percent of the paths that traversed an IX were shown to have potential bottlenecks, while the IX would be the bottleneck in 41% of those paths.

The authors also showed that Tier-1 ISPs had fewer potential bottleneck links than the smaller Tier-4 ISPs. The results are consistent with Pan's: intra-ISP links are not the main source of bottlenecks. Instead, the bottleneck links are the links used for peering: either direct links between ISPs, links to/from IXs, or links within IXs.

The MESCAL project [53] aims at alleviating some of the problems in the current interconnection agreements by advocating Service Level Specifications between peers [54]. The Service Level Specifications state performance requirements between providers, allowing for QoS guarantees between ISPs.

2.4 Transit Pricing

While peering contracts generally do not incur any financial obligations, transit services are sold to interested customers. Transit services are traditionally priced according to one of three major methods:

- Volume pricing,
- Committed rate pricing, and
- Percentile pricing.

Volume pricing consists of an ISP selling the right to transmit traffic based on a total amount of traffic that is sent and received, and is usually billed by the gigabyte. Volume discounts are the norm, to attract, and encourage, high volume customers. While an initial volume (credit) is purchased at the beginning of the contract period, extra volume can be purchased later, if required, for a higher price.

In *committed rate pricing*, the customer purchases the right to use a fixed capacity connection to the Internet. For example, the customer purchases a connection limited in rate to 4 Mbps from the ISP. In this case, the total amount transferred is usually unmetered, and thus unlimited within the capacity of the connection. Customers who require higher QoS may purchase a higher capacity connection to ensure lower delays.

Percentile pricing allows a customer to transmit bursty traffic without being charged at his peak rate (for incoming and/or outgoing traffic). Normally the 95th percentile of a customer's used bandwidth, measured over 5-minute intervals, is billed to the customer over the length of his contract. If a customer's traffic flowed at most 40 Mbps for 95 percent of the 5-minute time periods, then the customer is billed at 40 Mbps for the entire length of his contract, even though his peak usage may have been 100 Mbps. The 95th percentile of a customer's used bandwidth will usually be much greater than his average used bandwidth, but does allow the customer to send and receive bursty traffic, without having to purchase a fixed connection (using committed rate pricing) equal to his traffic's peak bandwidth requirement.

While the above three transit pricing schemes are the most common, to be competitive, new schemes are appearing in the marketplace, such as *partial transit* and *liquid bandwidth*.

Partial transit service providers provide transit services, but only to a subset of the entire Internet. Usually, a partial transit service provides regional connectivity, and is the result of the ISP providing its customers with access to the ISP's peering networks, but not to its upstream transit provider. This reduction in cost for the ISP results in significantly cheaper service for the customer. A customer can then purchase partial transit from one transit provider, for local traffic, while purchasing full transit services from another transit provider, thus reducing her overall transit costs.

Liquid bandwidth, or *bandwidth on demand,* is a term that has been heavily used in the industry. Many ISPs conceive, and advertise, that a 48-hour provisioning time is considered liquid bandwidth.

One company, InvisibleHand Networks Inc., has taken steps to improve the liquidity of bandwidth by creating a bandwidth market-enabling system called Merkato [55]. Merkato revolves around the concept of a micro-market: a marketplace for a single good, such as the bandwidth of a single link. A Merkato server manages one or more micro-markets and communicates with the necessary systems to provision bandwidth. Seller

and buyer agents control the amount of resources for a seller to sell, and the amount of resources for a buyer to buy. These agents are configured by their users and operate on their behalf.

An example of an ISP that uses the Merkato platform is Streaming Hand Services [56]. Streaming Hand Services connects to several Tier-1 ISPs, which are available at the collocation facility it resides in, and purchases their transit services. It is the bandwidth available to these Tier-1 ISPs that Streaming Hand Services sells to its customers. By using the Merkato platform, Streaming Hand Services auctions off the bandwidth in as little as 5-minute increments, using a type of generalized Vickrey auction, allowing its customers to bid for the right to use available bandwidth.

While the concept of liquid bandwidth for a single ISP is intriguing from a pricing perspective, it still does not address QoS issues, except within the originating ISP, as only its resources are not oversold.

2.5 Theoretical Pricing of IP Networks

The Internet Demand Experiment (INDEX) at Berkeley studied users' reactions to a service provider providing differentiated services with different prices to users [57,58]. INDEX showed that demand is sensitive to price and quality with persistent and large differences in demand among users. The top 10% of users generate 30 times more traffic then the lowest 10%. Thus the median traffic rate is much lower than the mean.

The authors showed how the large variation in demand among users leads to major inter-user subsidies under flat-rate pricing [57]. Under non-flat-rate pricing, the results of INDEX showed that average weekly expenditures vary greatly too. The authors note that 80% of the users subsidize the top 20% of the users; yet another example of the infamous 80/20 rule.

The argument of flat-rate versus usage-based charging is complicated. While the results of INDEX argue against flat-rate pricing, [59] notes that some users react unfavourably to

it. [59] argues that "usage-sensitive pricing tends to discourage the use of the Internet," which is contrary to the argument posed by the INDEX experiment. Even though a user may pay less under a usage-based system, that does not imply he will prefer such a system. A user may prefer flat-rate because the charge is predictable, transparent, and auditable [6(p.76)]. It is perhaps more interesting to note that users prefer flat-rate pricing when it comes to communication services, yet accept usage-based pricing for other utilities [6(p.77)].

However, it is also conceivable to use usage-based pricing to supplement, not replace, fixed pricing. In [60], Kirkby envisions a "turbo button" to increase the quality (speed) of the current session. The user, in effect, can use a second set of networks, with quality guarantees, but for a price.

McKie-Mason and Varian formulated the generalized Vickrey auction [10,61]. The original Vickrey auction is a second price sealed auction: bidders submit sealed bids for a single item, and the winner pays the value of the second highest bid. It can be proved that this auction format induces the bidders to submit their true value for the good, as a bidder's payment is not a function of his bid.

McKie-Mason and Varian designed the smart-market [11], a generalized Vickrey auction for the admission of packets across a link, such that each user pays the price per admitted packet equal to the highest bid of the packets not permitted to use the link. For example, assume 500 packets seek to be transmitted across a link at time t, but the link can only transmit 200 of these packets during the time interval. If there is a bid associated with each packet, then the 200 packets with the highest bids will be transmitted, but at the cost of the 201^{st} highest bid. The smart-market is hard to implement, especially across networks where the auction must occur at multiple links [62,63].

In [12], Courcoubetis et al. propose multi-unit descending price auctions, such that one auction exists per link. Customers can easily bid on multiple auctions, winning instantly, and having their bandwidth immediately allocated.

Odlyzko proposed Paris Metro Pricing, where the network is split into several logical sub-networks [13]. Here splitting was not geographical, but by bandwidth, such that each sub-network retains the topology of the original network, but with a fraction of the bandwidth. The network provider prices each network differently, allowing the customers to sort themselves into classes. It can be shown, as expected, that the higher priced network will experience lower congestion than the lower priced network. Pricing may vary from flat-rate to per-packet, or may allow the purchase of n packet credits which are valid for a specified time period (perhaps 10 seconds).

Frank Kelly has done fundamental work on defining the *effective bandwidth* of a session [64]. *Effective bandwidth* is the bandwidth required for a session to guarantee its performance. Sessions that are easy to multiplex have lower effective bandwidths than those that are hard to multiplex. For example, bursty real-time traffic may require its peak bandwidth to be reserved. Much of the previous work for effective bandwidth is summarized in [64].

Effective bandwidth pricing [6(p.205),8,14,64] entices users to determine and report their best expectations of effective bandwidth usage *a priori*. Users are charged not only on their bandwidth usage, but on the duration of their connection as well. Let $\alpha_{on-off}(M,h)$ be the effective bandwidth of an on-off source with mean data rate M and peak data rate h. Consider the case where the network measures a user's mean data rate, and polices for the peak data rate h. Now let m be the mean data rate reported in advance by the user, M be the mean data rate as measured by the network, and T be the duration of the connection. A linear function $f(m,h) = a(m,h) + b(m,h) \cdot M$ is calculated tangent to the effective bandwidth curve at point m, as illustrated in Figure 2.4. The user is charged $p \cdot T \cdot f(m,h)$, where p is a pricing scalar. The user is enticed to report her expected transmission rate accurately, because doing so will result in the best charging rate for her.

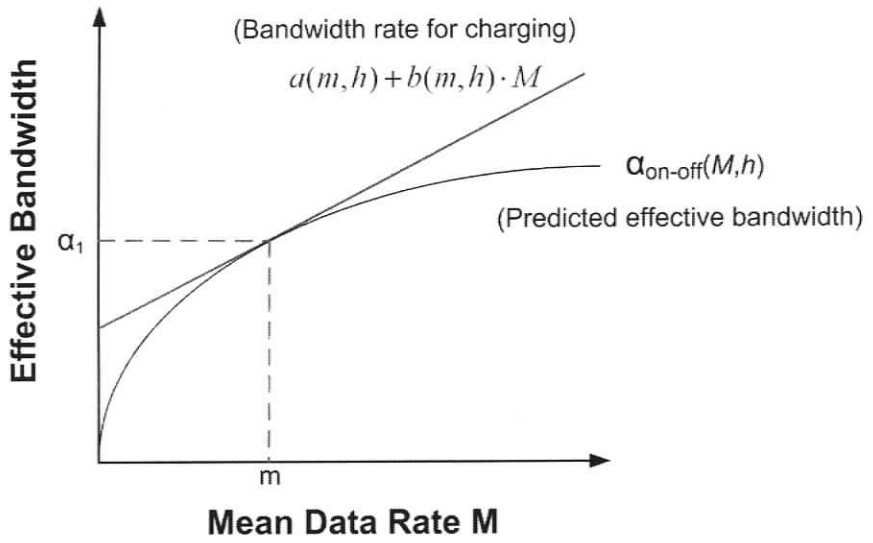


Figure 2.4: Effective bandwidth pricing.

Proportional fairness pricing [8,9] results in the proportional allocation of resources to users based on their willingness to pay. Each resource has a set price per unit, based on market clearing prices (where supply meets demand), and the amount of bandwidth a user receives is proportional to the amount they pay.

In [15], MacKie-Mason and Varian explain the theory of pricing congestible resources. The authors describe generalized consumer and producer optimization equations and many characteristics of a competitive marketplace. Their discussion essentially describes a generalized Paris Metro Pricing scheme. The market price for a service (or quality) is set where the price a user is willing to pay for additional usage equals the price other users are willing to pay the provider to refrain from selling additional usage.

In [16], the authors propose a pricing scheme where a charge is based on the amount of congestion. User adaptation is performed in advanced, based on the predictions of the charge.

Wang and Schulzrinne designed the Resource Negotiation and Pricing (RNAP) protocol [17]. The protocol follows a similar structure to the RSVP protocol, except that it relays pricing information at each node as well. To address the scalability issues, the authors studied how to aggregate and disaggregate messages that start from one domain, and are

destined for a shared domain. Like RSVP, reservations are confirmed periodically so only soft-state information needs to be stored.

Li et al. proposed a tariff-based framework to integrate pricing and admission control [18]. Their scheme included the services of DiffServ core networks to be posted in global price tables at the access networks. Prices are adjusted in a manner to avoid over-utilization. Admission of a session, across the core, could then be based on the posted price at the access networks, providing distributed admission control.

There are, however, many difficulties with Li's framework. First, prices are posted at an access network based on the route as determined by the Border Gateway Protocol (BGP). There is incentive for a network to inflate its price if price is not considered in the route determination. Secondly, the concept of admission control is loosely defined in the author's paper. It is implied that a user is charged at the posted rate at the start of his connection. There is incentive for such a user to sell the use of his connection to others (in his access network) if his purchase price is lower than the current market price. This, however, leads to the collapse of the framework, as over-utilization can occur. Third, as the author briefly addressed, there is no clear concept of what should happen if the routing tables, and thus the posted prices, change. Finally, there is the issue of synchronizing the global price tables with the actual routing, and the time to propagate new pricing information.

In [19], the authors present a contracting protocol for QoS in a multi-provider environment. The protocol negotiates a contract to deliver QoS between a set of ISPs; however, the protocol works on a per stream basis, making scalability a concern.

Jensen et al. desribe a framework for managing QoS in a multi-provider environment [20]. They assume a "one-stop responsibility," where the sole provider must guarantee QoS for the user, even if the traffic crosses multiple ISPs.

2.6 Bandwidth Trading

From the mid 1990s to early 2000 there was an excess of bandwidth installed as a result of technological development and the technology investment boom due to unrealistic expectations of the growth of the Internet. It cannot be stressed enough that the Internet is still in its infancy, and while there may appear to be a bandwidth glut, that glut may be short-lived. Also, "as for a North American bandwidth glut, if there is one, it's a transitory phenomenon. The real problem facing carriers is not an oversupply of capacity but their inability to figure out a way to charge customers for anything other than voice traffic" [65 (as referenced in [66])]. This view is also reinforced in [67].

Bandwidth is rapidly becoming a commodity, due to the advent of bandwidth exchanges. Bandwidth markets have been advancing quickly. Bandwidth exchanges are in their infancy, with the first commodity bandwidth trade occurring in December 1999 by Enron [68], though the industry is quickly evolving. It was only a few years ago when bandwidth was purchased for a minimum of 6 months to a year, and required a provisioning time of approximately 6 months [69]. Now, with the advent of *pooling points*, provisioning can occur within minutes [70].

Pooling points serve as physical locations where buyers and sellers connect their networks, enabling buyers to directly access resources they purchase. There are two distinct types of pooling points, as shown in Figure 2.5: centralized/physical and metro/virtual pooling points [72].

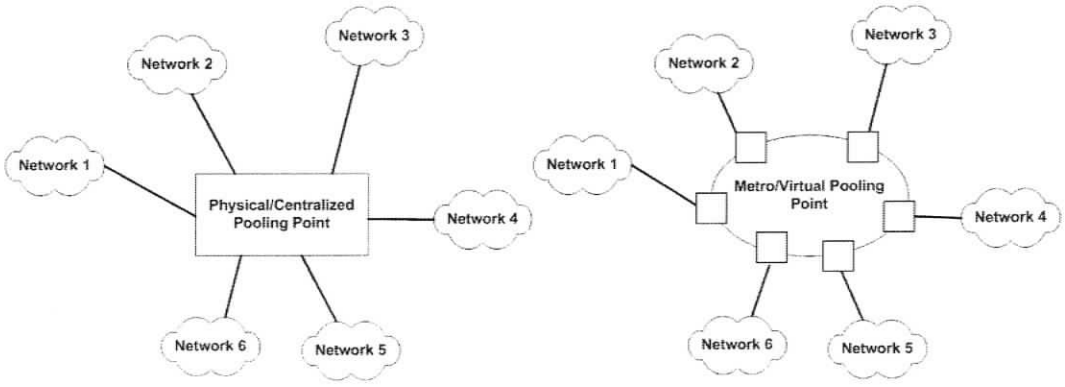


Figure 2.5: Centralized and virtual pooling points.

Centralized pooling points consist of a single physical location to which buyers and sellers connect. A virtual pooling point has a more decentralized structure. In a virtual pooling point, there are several attachment sites, potentially lowering the cost for connectivity, as sites can be distributed closer to customers.

As provisioning times fall, the possibility of short-term contracts grows. With the use of intelligent software agents, researchers envision trading for extremely short-term contracts (seconds or less) [24]. In [71], the authors propose that "future applications can expect bandwidth to be available on demand in a model similar to main computer memory on demand. Just as the operating system makes memory available to running programs on an as needed basis, future distributed operating systems should be able to make telecom bandwidth available to applications on demand through bandwidth traders."

In [72] the authors categorize bandwidth products into 6 classes:

0. Enabling Bandwidth – products that enable bandwidth, such as cable conduits.
1. Physical Plant Bandwidth – physical media such as dark fibre.
2. Wavelength Channel – an optical channel between pooling points, such as a wavelength or lightpath.
3. Clear Channel – a digital communication channel between two pooling points, with defined throughput and frame format. Can have hard, statistical, or best-effort guarantees.

4. Telephony Minutes – minutes of telephony that are routed through the seller's network from a designated pooling point, to the destination city.
5. Best-Effort Internet Access – access to the Internet through the seller's network.

The main traded bandwidth commodities include switched minutes and DS-3 lines. IP QoS flows (via MPLS [73] or DiffServ/IntServ) and optical wavelengths are probably major future commodities, but standards need to be established before they are really feasible [72]. Arbinet Communications sells telephony minutes from major exchanges, differentiating products by their quality (i.e. call completion rates), and by offering special deals to customers such as multi-day rate locking and building circuits on premium routes. With the creation of new IP phone services, such as through Primus and Vonage, these telephony minutes might even become tradable by home users, for long distance calls.

Ferreira et al. [74] showed that volatility still occurs even though prices of bandwidth are decreasing. Volatility is one aspect that encourages buyers and sellers to trade a commodity. Additionally, we repeat that while the supply of bandwidth is great now, relative to demand, it should not be presumed that this will always be so.

3 QoS-Transit Services

In this chapter we describe primitive services, offered by ISPs, which provide QoS for traffic flows. The performance of each service is guaranteed by controlling demand through dynamic pricing. We believe session-level admission and congestion control schemes should be engineered at the edges of the network, on top of a primitive, QoS-aware, packet delivery mechanism. By using dynamic pricing, packet-level congestion control is provided.

We first motivate the idea behind QoS-Transit Services. We then define QoS-Transit Services: a set of primitive services offered by an ISP in order to deliver packets with statistical performance guarantees within its network, using dynamic pricing. We then discuss the fairness properties of such QoS services and define the requirements for an ISP to offer them. To illustrate how QoS is guaranteed with dynamic pricing, we define and provide examples of pricing mechanisms that dynamically adjust prices for QoS-Transit Services. Finally, we provide simulation results of one of our pricing mechanisms, demonstrating that even a simple pricing mechanism can provide QoS guarantees.

3.1 Overview

QoS in current transit agreements is specified by a Service Level Agreement (SLA), which usually defines the maximum delay and packet loss for a network. The maximum bandwidth of a transit agreement is usually determined by the customer's connection to their ISP. Today's ISPs can engineer their networks to achieve good performance for their customers, because their number of customers remains nearly constant for durations of a few weeks or more, and their requirements are known in advance, or can at least be statistically approximated. However, user traffic typically traverses multiple networks. To achieve a specified level of end-to-end QoS, there must be no bottlenecks on any of

the networks that a flow traverses, either from poor network management, network failure, or due to unexpected demand.

Transit agreements allow the transmission of packets across a network to which they are not destined. These agreements often detail QoS guarantees, but only across the network of the ISP from which the transit agreement was purchased. We envision that a user be able to transmit data across a set of networks, where each network guarantees the stated QoS across itself only. Dynamic pricing can be used by a network provider to control the demand of traffic and thus achieve its stated QoS across its network. The concept is illustrated in Figure 3.1. End-users need not be aware of these QoS-enabled ISPs, depending instead on their local ISPs to interact with the QoS-enabled ISPs on their behalf.

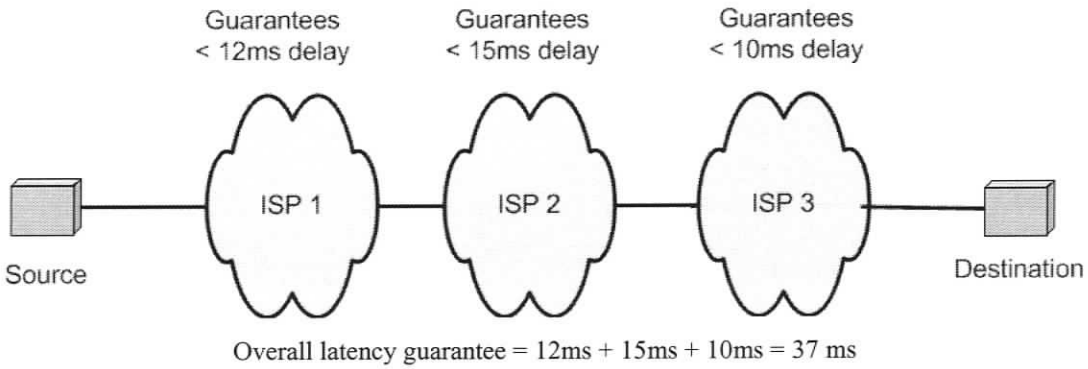


Figure 3.1: Traffic transiting three ISPs where each ISP guarantees the performance across its network.

While there are many definitions of QoS, there exist a number of similarities among them. QoS is generally defined in terms of packet loss, jitter, maximum delay bounds, and bandwidth requirements (such as peak [75] or effective bandwidth [14]).

Our idea is to provide a general framework for providing QoS over a nearly-stateless network, to which functionality can be added at the edges to provide higher-level QoS guarantees and admission control. This end-to-end approach [76] is intended to allow the creation of a model that is easy to implement and flexible, yet designed to allow the development of higher-level end components for additional functionality such as session-level QoS, call admission control, multicasting, complex charging contracts, billing, etc.

The design of the present Internet has amply proven the advantages of simplicity and minimal network state for scalability and fault tolerance.

3.2 Definitions

We propose that customers be able to change their network requirements and their transit providers, based on the dynamic, time-varying, requirements of their sessions. An ISP may then have a fluctuating number of customers, and moreover, each of those customer's requirements might change rapidly in the short-term (say, less than one second). To allow ISPs to offer such flexibility efficiently to customers, we propose the use of QoS-Transit Services.

QoS-Transit Services are primitive services offered by a network in order to deliver packets with performance guarantees within the network, between two defined points, using dynamic pricing. We represent a QoS-Transit Service by a source, destination, maximum delay, maximum packet loss, maximum jitter, *statistical calculation period*, and a *tariff definition*. A QoS-Transit Service thus specifies a path across the network with stated performance criteria. An ISP that provides QoS-Transit Services is a *QoS-ISP*.

The *statistical calculation period* is the period for which pricing and statistical guarantees are determined. For example, if a customer is being charged per kilobit per second of traffic that they generate, then the statistical calculation period determines over what interval this rate is determined. For dynamic pricing, we propose that the period be short, such as 1 second. The actual period chosen will depend on the tariff chosen and the dynamics of the traffic.

The *tariff definition* defines the form of the tariff. Example types of tariff definitions include:

- price per packet, regardless of size,
- price per packet, considering size,

- price per Mbps over each statistical calculation period,
- price discount for heavy users,
- price penalty for heavy users,
- percentile pricing over numerous statistical calculation periods,
- purchase credit to send n packets, possibly with expiry time and size bounds, and
- purchase credit to send n bits, possibly with an expiry time.

The tariff definition defines the relationship between bandwidth and pricing. Bandwidth may be sold based on a data rate, such as megabits per second, or by data volume, such as in gigabytes. Consider data rate pricing, where the price per unit is reduced for heavier users, as shown in Table 3.1, where p represents the current price of the QoS-Transit Service. Data rate can be measured as a peak or an average data rate (or based on percentile pricing), determined over specified time intervals.

Table 3.1: Sample pricing for a service with data rate pricing.

Rate (Mbps)	Price ($/Mbps/second)
0-4	p
4-8	$0.75p$
8-12	$0.65p$
12+	$0.5p$

For another example of a pricing relationship we consider volume pricing, where purchases have an expiry time. An example of a pricing function is:

$$p = cb + rt, \tag{3.1}$$

where p is the cost to purchase credit for the transmission of b gigabytes of data within the next t seconds. Here r represents the price of extending the transmission time for each additional second and c is the price per gigabyte.

The period at which new prices are calculated is determined by the QoS-ISP. New prices may be calculated at very short time intervals (e.g. less than a second), or they may be updated at longer intervals, perhaps as long as a week. The longer the period between pricing updates, the more bandwidth that may need to be over-provisioned in order to

guarantee the advertised QoS for a QoS-Transit Service, and the less efficient the network. However, prices that change often are less predictable, and potentially less attractive to customers of the QoS-Transit Service.

Figure 3.2 is a sample network with three QoS-Transit Services. QoS-transit Services are defined between edge nodes and multiple services can exist between a pair of routers, as each service may guarantee different levels of service.

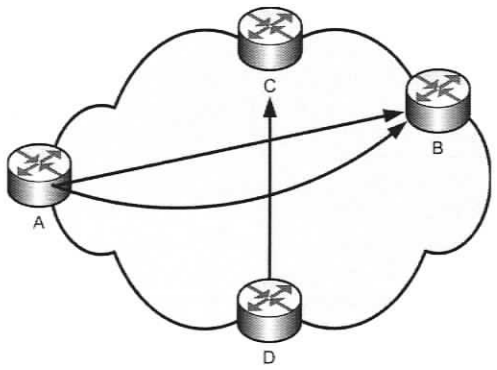


Figure 3.2: Sample network with three QoS-Transit services.

QoS-Transit Services guarantee the delivery of packets within certain performance criteria. The QoS resulting from QoS-Transit Services does not guarantee bandwidth over the length of a session. Prices may increase, due to high demand, prompting a customer to cease, or reduce, the use of a QoS-Transit Service. However, recall that this behaviour is a low-level service, and that higher-level services such as admission control, multicast, and sender or receiver-pay billing can be implemented using these low-level services.

Higher-level services may be implemented either at the level of the user's ISP, or by other customers of QoS-Transit Services. A user's ISP may purchase the use of QoS-Transit Services in order to improve QoS for certain users, or for certain traffic (i.e. multimedia traffic). As another example, a customer may be a multicast service provider, purchasing QoS-Transit Services to form multicast trees, and charging its users for use of those multicast trees. End-users thus need not be aware of the dynamic pricing of QoS-Transit Services.

We stress that, for competitive reasons, providers must be able to choose their own tariffs. We will later discuss how the resulting complexity of tariffs can be hidden from the user.

Throughout this dissertation we carefully differentiate between *customers* and *users*. *Customers* are entities that purchase QoS-Transit Services and *users* are entities responsible for creating traffic. A customer purchases QoS-Transit Services for an aggregate of users. For example, a customer may be a large enterprise, whereas a user may be an individual within that organization. While we do not disallow the possibility that a customer may be an individual end-user, we expect that is not likely to be the case.

3.3 Fairness

As with any pricing mechanism, it is important to discuss fairness. Since we have argued that QoS-ISPs can price their services as they wish, and may even price discriminate for certain customers, we do not guarantee any type of fairness. However, if the same pricing parameter is advertised to all customers, we can claim QoS-Transit Services to be fair.

If only a single price is given in the tariff definition (e.g. $\$p$/Mbps/minute), then pricing is proportionally fair, as the rate of transmitted traffic by each customer is proportional to the amount each customer pays. Given a set of customers with traffic to transmit across a service, each customer will likely transmit some of its traffic, at the stated price. However, those customers who have a higher valuation of their traffic will likely send more.

Given a price p, the amount of traffic each customer will send is determined by their demand function. If we assume each customer's demand function to be strictly decreasing, then the first unit of each customer's traffic (e.g. the first kbps) is the most valuable to the customer, making it likely that each customer has an opportunity to send some traffic. This also supports our assumption that a service will carry traffic from

numerous customers. Additionally, recall that each customer's traffic is actually aggregated from numerous users.

3.4 Requirements to Offer QoS-Transit Services

We define the conditions to provide QoS-Transit services across a single network. Ultimately, the QoS-ISPs must provide pricing feedback often enough to ensure that the performance of all the QoS-Transit Services meets their advertised guarantees. In some instances, where demand follows predictable cyclic patterns, this may be done by over-provisioning the network and using a price schedule, such as with time-of-day pricing. However, in this section we examine service requirements for the more general case, where demand is not always predictable, or if it is generally predictable, cases where unexpected fluctuations in demand may occur.

The network may have circuit-switching or label-switching capabilities. While not required, these capabilities would significantly aid in achieving the quality guarantees for QoS-Transit Services. MPLS is our chosen technology for our possible architecture, discussed in Chapter 7.

To guarantee the stated delivery of packets, queues cannot be unexpectedly overrun. Traffic for a QoS-Transit Service may be dropped or queued, but only within the acceptable limits of the QoS guarantees. We assume that all incoming traffic will be charged based on currently stated prices and that customers are aware of and agree to this fact. If a network is publicly accessible, with publicly posted prices, then to guarantee all QoS-Transit Services the network must set prices to guarantee that bandwidth is not over-consumed.

Ingress routers must identify the requested service and the entity to be billed for charging purposes for incoming packets. Ways to do this are discussed in Chapter 7. We stress that it should not be assumed that the sender is charged. It is equally likely that the receiver, or a third party, is liable or, especially in the case of multicast, that the costs may be shared amongst a group of participants.

3.5 Pricing Mechanisms for QoS-Transit Services

To achieve performance guarantees for QoS-Transit Services, QoS-ISPs implement dynamic pricing to control demand, and thus indirectly control the performance received by transmitted packets. In this section we describe two simple pricing mechanisms to demonstrate how a QoS-ISP can dynamically set prices for each of its services.

Both mechanisms employ per packet pricing, in which a customer is charged an amount per packet equal to the currently posted price for that QoS-Transit Service.

3.5.1 Reaction Point Pricing

Using a similar approach to Li et al. [18], we use a hyperbolic growth function for pricing once demand reaches a predetermined point, called the *reaction point,* y_r. A hyperbolic function is suitable because it allows an infinite value to be achieved in a finite time. Before demand for a QoS-Transit Service reaches the reaction point, its price is set at a constant value, the base price, p_0. Let the current measured demand be x and the amount of bandwidth available be y. The price of a QoS-Transit Service is then priced as follows:

$$p(x) = \begin{cases} p_0, & x \le y_r \\ p_0 \times \left(\dfrac{y - y_r}{y - x} \right)^{\lambda}, & x > y_r, \end{cases} \tag{3.2}$$

where λ alters the steepness of the hyperbolic curve. Figure 3.3 shows an example of the price function. Note that the values of λ and y_r will be set depending on the burstiness of the demand for the associated QoS-Transit Service.

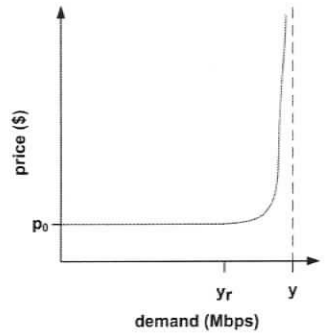


Figure 3.3: Reaction Point Pricing adjusts the price when demand reaches the reaction point, y_r.

3.5.2 Utilization Level Tatonnement

Reaction Point Pricing provides simple congestion control through a pricing mechanism. It does not intuitively give maximum revenue; though it does provide a constant price until congestion prevention is required. Such a constant price could be preferred by customers.

Alternatively, a *tatonnement*-like process can be implemented to use dynamic prices to attain a goal level of demand. *Tatonnement* is an iterative process where prices are adjusted until supply equals demand; however, we aim to set prices so demand is less than supply to allow for transient bursts, as well as to give the network time to adjust to future demand increases.

Utilization Level Tatonnement (ULT) focuses primarily on the demand for a QoS-Transit Service. The idea is to use iterative price adjustments to achieve a desired demand for a QoS-Transit Service. Let the demand for a time interval be an exponentially weighted moving average:

$$x(t) = w \cdot x_{measure}(t) + (1-w) \cdot x(t-1), \tag{3.3}$$

where $x(t)$ is the exponentially weighted moving average demand calculated for time t, $x_{measure}(t)$ is the measured average demand for time t, and w denotes the weighting of the measured average demand.

Let y be the available bandwidth along the path on which a QoS-Transit Service is routed. Additionally, let x_{goal} be the goal demand, where $x_{goal} < y$. The expected spare bandwidth, $y - x_{goal}$, is held in reserve in the event of traffic bursts. Thus the quantity of expected excess supply should match the expected level of burstiness of the traffic. Highly bursty traffic requires a large amount of excess supply, while constant traffic requires none.

ULT's price adjustments should approximate

$$p(t+1) = p(t) + \Delta p, \text{ where } \begin{cases} \Delta p > 0, x(t) > x_{goal} \\ \Delta p = 0, x(t) = x_{goal} \\ \Delta p < 0, x(t) < x_{goal}, \end{cases} \tag{3.4}$$

where Δp is calculated to let $x(p[t+1]) \approx x_{goal}$, and $p(t)$ is the price at time t. We consider time to be discrete due to the delay in feedback from pricing. Prices are updated periodically, and not continuously. To measure the customer's reaction to a price change, the price not only needs to be transmitted and received by the customer, but the customer's traffic whose rate is adjusted due to the price change must reach the ingress node of the QoS-Transit Service.

ULT periodically adjusts the price of a QoS-Transit Service so that the desired demand, x_{goal}, is maintained. Figure 3.4 demonstrates how the price adjustment, Δp, can be calculated for Equation 3.4, where g is the *price radius*. The *price radius* is the amount that demand must vary from x_{goal} before a maximum price changes occurs. A more complicated price adjustment scheme may be employed, considering not only the current level of demand, but the direction and amount of change in demand due to recent price adjustments.

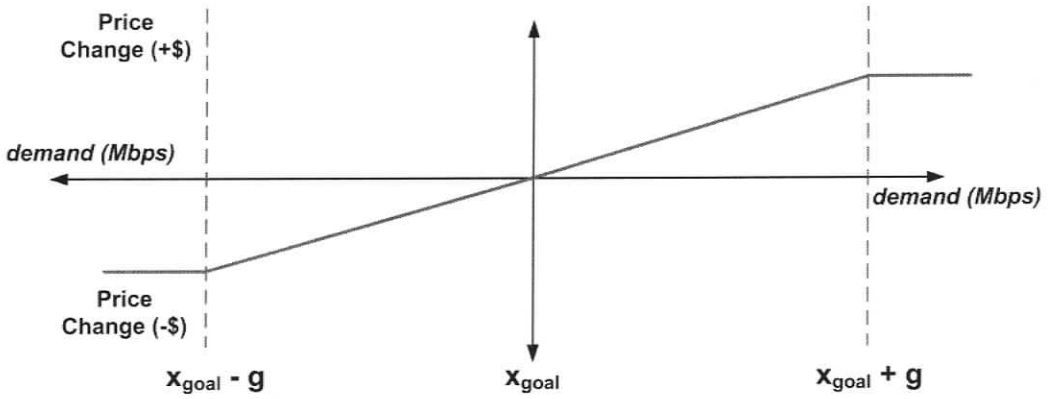


Figure 3.4: Price adjustments in ULT to achieve the goal demand.

Let $\gamma(t)$ be the distance from the goal demand:

$$\gamma(t) = \begin{cases} -1, & \frac{x(t) - x_{goal}}{g} < -1 \\ +1, & \frac{x(t) - x_{goal}}{g} > 1 \\ \frac{x(t) - x_{goal}}{g}, & \text{otherwise,} \end{cases} \tag{3.5}$$

where $-1 \le \gamma(t) \le 1$ and $x(t)$ is the exponentially weighted moving average of demand for the QoS-Transit Service at time t.

Similar to [9], we define price adjustments for a QoS-Transit Service as:

$$p(t+1) = w' \cdot p(t) + (1 - w') \cdot p(t - RTT_{avg}) + \lambda \cdot \gamma(t), \tag{3.6}$$

where λ is a pricing scalar and RTT_{avg} is the average Round Trip Time between the ingress router for the QoS-Transit Service and a customer. The effects of a price change are deferred, on average, by RTT_{avg}: the time for the price change to reach a customer, followed by the time for her altered traffic to reach the ingress router. Consequently, in addition to considering the current price, we also consider the price at time $t - RTT_{avg}$. The price at time $t - RTT_{avg}$ was the last price sent to customers that affected the present

demand. The variable w' denotes the weighting between the present price and the past price.

3.6 Simulations of QoS-Transit Services

In this section we demonstrate that even a simple pricing mechanism can achieve respectable guarantees for QoS-Transit Services. To do so, we simulated ULT, the pricing mechanism described in the previous section, using ns-2 (Network Simulator 2 [77]).

3.6.1 Overview

Through our simulations we demonstrate the feasibility of dynamic pricing to provide QoS guarantees to aggregated traffic (i.e. QoS-Transit Services) across a network. The design of our simulation consists of a single link, representing a path with reserved bandwidth across a network. The ingress node monitors observed demand, returning an updated price to the user, as shown in Figure 3.5.

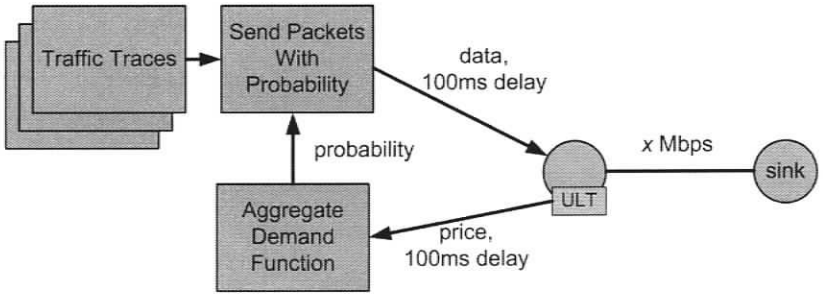


Figure 3.5: Simulation setup including the modeling of an aggregate of users.

The traffic consists of traffic traces from a core traffic trace, an access link traffic trace, or traffic traces of MPEG movies. The MPEG movies contain only slight traffic bursts, especially when multiplexed. The core traffic trace represents traffic that is moderately bursty. The access link traffic trace represents extremely bursty traffic.

Customers react according to a demand function. The demand function defines the probability of sending each packet from the traffic traces: simulating users adjusting their

data rates based on price information. We use a demand function derived from the INDEX experiment, as outlined in Appendix A and derived in Appendix B. In Section 3.6.3, we derive the simulated price elasticity of demand of the demand function, which we show to be 0.37, regardless of the currently posted price.

For all tests, we consider 200 ms intervals for calculating statistics and for advertising new prices. The maximum amount to change a price is 10% of the current price. We assume a constant delay of 100 ms for updating the customers of any price changes, as well as 100 ms for the altered traffic to reach the ingress node. We also assume that the QoS-Transit Services guarantee at most 1% packet loss. Packets that would have been delayed beyond the maximum delay are simply dropped.

Details of the setup of the simulations are given in Appendix A: Simulation Setup.

3.6.2 Varying Maximum Price Changes

Our first simulation consists of varying the maximum amount that the price of a QoS-Transit Service can change at once. While it is understood that prices must change enough to control demand, we show that varying prices too much at one time can adversely affect ULT's ability to control the demand for a QoS-Transit Service. The maximum price change for a given pricing interval is 2%, 4%, 8%, 16%, or 32%, on traffic consisting of 6 access link traffic traces. The capacity of the path is 125 Mbps, and ULT is set to have a goal demand of 112 Mbps and a price radius of 13 Mbps. (Goal demand and price radius, which are parameters of ULT, were defined previously in Section 3.5.2.)

Figure 3.6 shows the probability of occurrence of the average bandwidth demanded by a QoS-Transit Service for each of the maximum price changes. As can be seen, the maximum price change of 2% results in traffic controlled around the 112 Mbps goal rate. However, as the maximum price change increases, so does the inability of the pricing mechanism to control the demand for the QoS-Transit Service. With a 32% maximum price change, it can be seen that the average bandwidth demanded is not centered around

the goal demand, due to ULT adjusting prices, and thus affecting demand, by too-large increments. Price changes of up to 2-16% did not result in appreciable changes in the bandwidth demands; however, maximum price changes of 32% drastically affected the ability of ULT to control the bandwidth demanded.

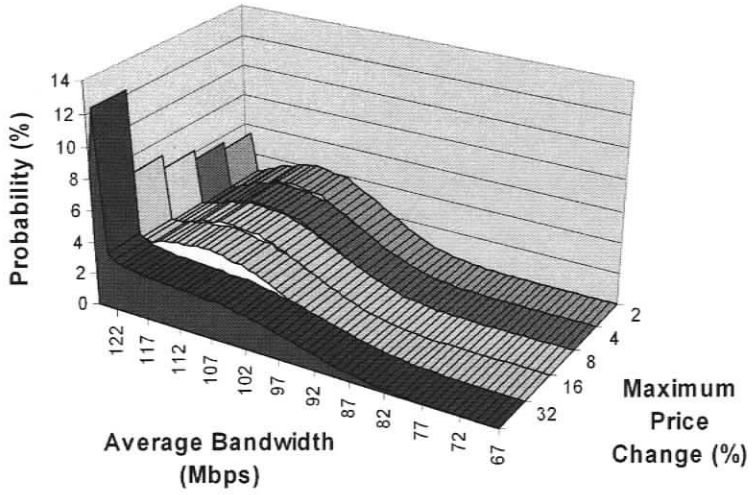


Figure 3.6: Probability of occurrence of the average bandwidth demands of a QoS-Transit Service for differing maximum price changes.

3.6.3 Varying Price Elasticity of Demand

We now test the ability of ULT to control demand with different price elasticities of demand. The capacity of the path is 125 Mbps, and ULT is set to have a goal demand of 112 Mbps and a price radius of 13 Mbps. The traffic consists of 6 access link traffic traces.

The demand function we use to represent the aggregate demand of users, as defined and developed in Appendix B, is $x(p) = 10^{0.37(1-\log p)} = 10^{0.37} p^{-0.37}$. Price elasticity of demand is defined by:

$$\xi = \frac{\Delta x / x(p)}{\Delta p / p} = \frac{p}{x(p)} \cdot \frac{\Delta x}{\Delta p} = \frac{p}{x(p)} \cdot x'(p). \tag{3.7}$$

We define the constant $b = 0.37$ for the purpose of our derivation. Then, using $x'(p) = -0.37 \cdot 10^{0.37} p^{-1.37}$, as defined by Equation B.9 in Appendix B, we can determine the price elasticity of demand of our demand function:

$$\xi = \frac{p}{x(p)} \cdot x'(p) = \frac{p}{10^b p^{-b}} \cdot (-b \cdot 10^b p^{-b-1}) = \frac{-p \cdot b \cdot 10^b p^{-b} \cdot p^{-1}}{10^b p^{-b}} = -b . \qquad (3.8)$$

We can therefore define the demand function for various price elasticities of demands:

$$x(p) = 10^{-\xi(1-\log p)}, \qquad (3.9)$$

where ξ defines the price elasticity of demand of the aggregate of users.

Having previously shown that a pricing mechanism should not adjust prices too much at once, we set the maximum price change relative to the price elasticity of demand. Table 3.2 shows the maximum price change associated with each price elasticity of demand that we test.

Table 3.2: Maximum price changes for various price elasticities of demand.

Price Elasticity of Demand	-0.1	-0.3	-0.5	-0.7	-0.9	-1.1
Maximum Price Change (%)	20	6.67	4	2.86	2.22	1.82

Figures 3.7 and 3.8 show the probability of occurrence of the average bandwidth demands and the average delays[2], respectively, for each of the price elasticities of demand. There is no appreciable difference in the probability of occurrence of the average bandwidths demands or of the average delays, for the various price elasticities of

[2] To graph the delays, the delays are accumulated based on the mathematical floor of the delays. That is 0.9ms is considered as 0ms for the purpose of the graphs.

demand. Thus, if the maximum price change is set appropriately, then ULT is very effective in controlling traffic, regardless of its price elasticity of demand.

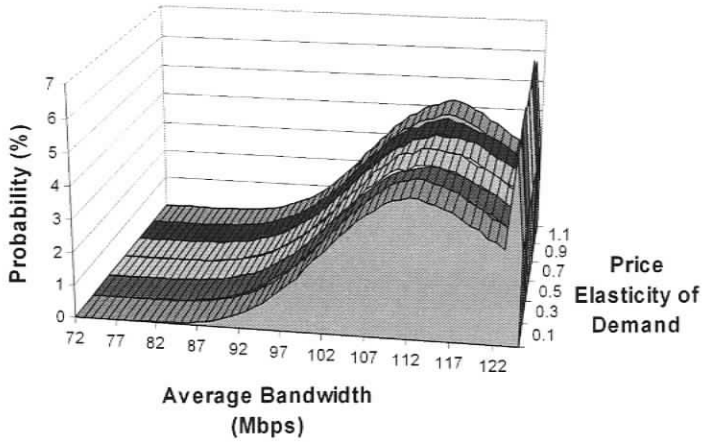


Figure 3.7: Probability of occurrence of the average bandwidth demands of a QoS-Transit Service for varying price elasticities of demand.

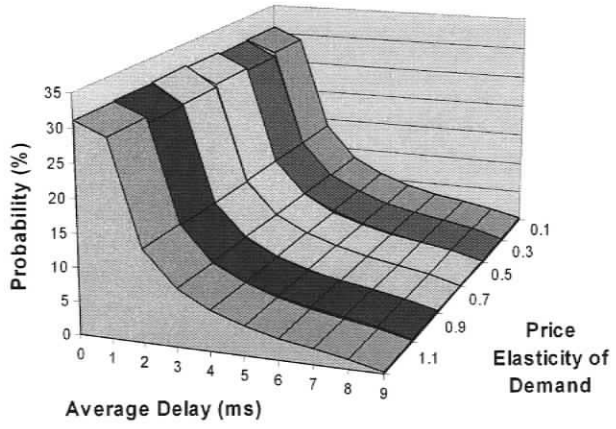


Figure 3.8: Probability of occurrence of the average delays of traffic of a QoS-Transit Service for varying price elasticities of demand.

3.6.4 Multiplexing Gains

In this section we demonstrate the gains that can be achieved through multiplexing several QoS-Transit Services on a shared path within a QoS-ISP. Multiplexing QoS-Transit Services on a path can improve the QoS guarantees that a QoS-ISP can offer.

Multiplexing QoS-Transit Services across individual links, instead of across an entire path, should provide similar benefits.

We multiplex from 1 to 10 QoS-Transit Services on a path, where the traffic for each QoS-Transit Service is an instance of an access link traffic trace. In each test, the capacity of the link is 40 Mbps multiplied by the number of QoS-Transit Services being simulated. For each service, ULT is set to have a goal demand of 25 Mbps and a price radius of 8 Mbps.

The percentage of time that packet loss exceeded 1% is shown in Table 3.3. The results demonstrate that QoS-Transit Services can be multiplexed to provide fairly strong packet loss guarantees without allocating excessive bandwidth. The data in Table 3.3 is expanded in Figure 3.9, which demonstrates the probability of occurrence of packet loss in any 200 ms time interval.

Table 3.3: Percentage of time that packet loss exceeds 1%.

Number of Services	1	2	3	4	5	6	7	8	9	10
Packet Loss > 1% (% of time)	12.29	3.69	2.12	1.11	0.68	0.52	0.40	0.33	0.28	0.24

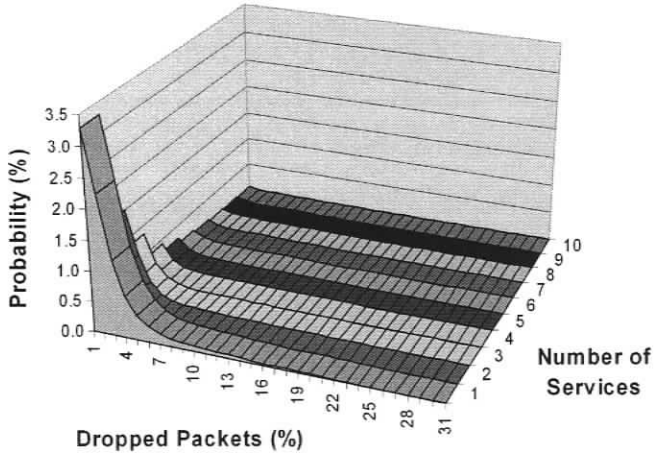


Figure 3.9: Probability of occurrence of average packet loss with respect to the degree of multiplexing.

ISPs often guarantee packet loss, across their networks only, between 1 and 0.1%, averaged over a month. Infinity Internet guarantees the monthly average of packet loss to be below 1% [96]. MCI (owned by Verizon), over the last year, provided between 0 and 0.56% packet loss, depending on location, based on a monthly averages [95]. Similarly, looking at the May 2005 packet loss averages for British Telecom, common packet loss was between 0.2 and 0.5%, especially in Latin America, Asia, and regions of Europe [78]. The multiplexed traffic within QoS-Transit Services can achieve lower overall packet loss, as shown in Table 3.4. The packet loss achieved by multiplexing 10 very bursty QoS-Transit services was only 0.016%.

Table 3.4: Percentage of packet loss for multiplexed traffic.

Number of Services	1	2	3	4	5	6	7	8	9	10
Packet Loss (%)	0.455	0.157	0.089	0.059	0.045	0.034	0.028	0.023	0.019	0.016

Figures 3.10 and 3.11 show the probability of average and maximum delays with respect to the degree of multiplexing. The average delays quickly drop below 1 ms when multiplexing is considered. The expected maximum delays are reduced with the degree of multiplexing. With 10 multiplexed services, a maximum delay of 2 ms or less occurred 99.502% of the time.

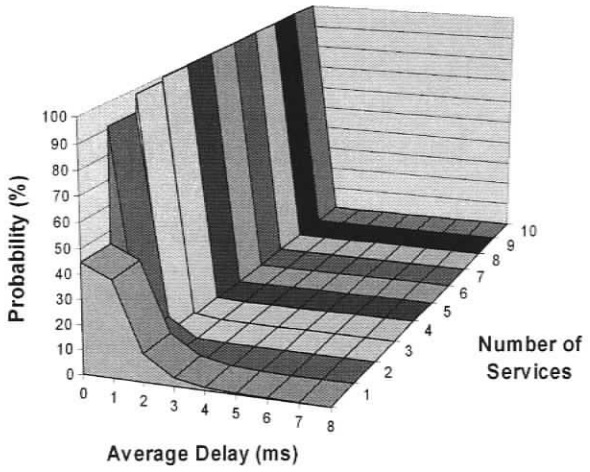


Figure 3.10: Probability of occurrence of average delays with respect to the degree of multiplexing.

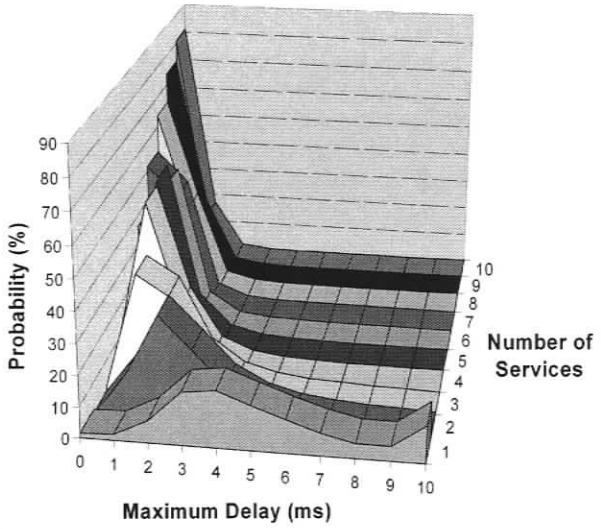


Figure 3.11: Probability of occurrence of maximum delays with respect to the degree of multiplexing.

Multiplexing gains come not only as a result of an increase of traffic, which may smooth the induced traffic, but also due to ULT's inability to precisely control the average rate of traffic. At any point in time, the data rate consumed by a QoS-Transit Service may be above or below its intended goal demand. If the queue of one or more QoS-Transit Services is empty, then the time allocated for those services is spent serving the other services. Also, with the increase in the data rate required to serve an increased number of services, larger amounts of traffic can be removed from the queues in the same amount of time. Multiplexing QoS-Transit Services can thus improve the performance received by all of the QoS-Transit Services.

If the aggregated traffic carried by a QoS-Transit Services is bursty, multiplexing can be used to improve the guarantees provided by the QoS-ISP.

3.6.5 Increasing Demand

While changes in demand can be slow (such as over the course of a day), or temporary (such as a short burst of traffic), they can also be sudden and sustained. We demonstrate the ability of ULT to respond to sustained increases in demand. Traffic is initially injected into the network for a stabilization period of 2 minutes. Demand is then increased by a factor of 4, at a rate of r percent per second. The capacity of the path is 62.5 Mbps, and ULT is set to have a goal demand of 50Mbps and a price radius of 12.5 Mbps. We first test for video traffic, then consider core network traffic, and finally consider multiplexed access link traffic.

Figure 3.12 demonstrates that non-bursty traffic can be controlled well, even during long periods with a high rate of increasing demand, by considering multiplexed movie traces. The demanded bandwidth increased gradually as the rate of demand increases increased. Even with demand increasing at a rate of 10% per second, packet loss exceeded 1% in only 0.5% of the 200 ms time intervals for the 600 movies used in the simulation, as shown in Figure 3.13. The average and maximum delays were also controlled, resulting in little difference in the delays for different rates of demand increases. For example, Figure 3.14 shows the probability of occurrence of the maximum delays of traffic using the QoS-Transit Service. There was only a slight increase in the measured maximum delays as the rate of demand increased in this simulation.

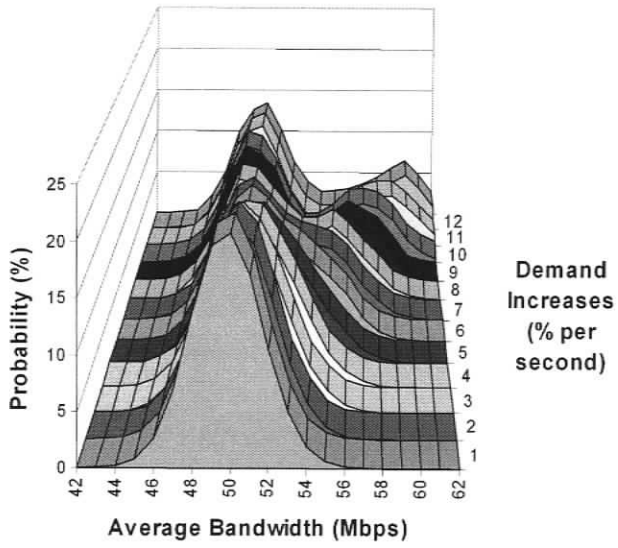


Figure 3.12: Probability of occurrence of the average bandwidth demands for varying demand increases of multiplexed movie traffic.

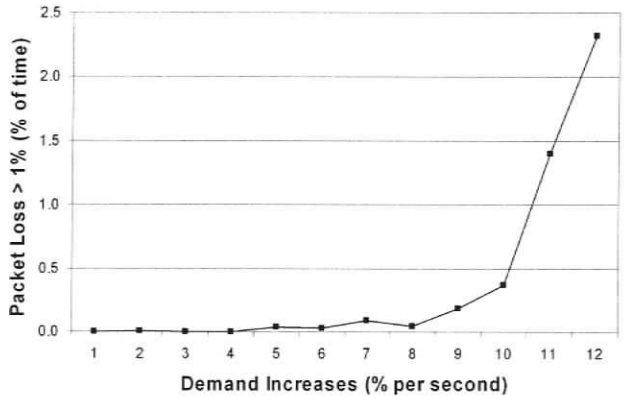


Figure 3.13: Percentage of time that packet loss exceeds 1% for varying demand increases of multiplexed movie traffic.

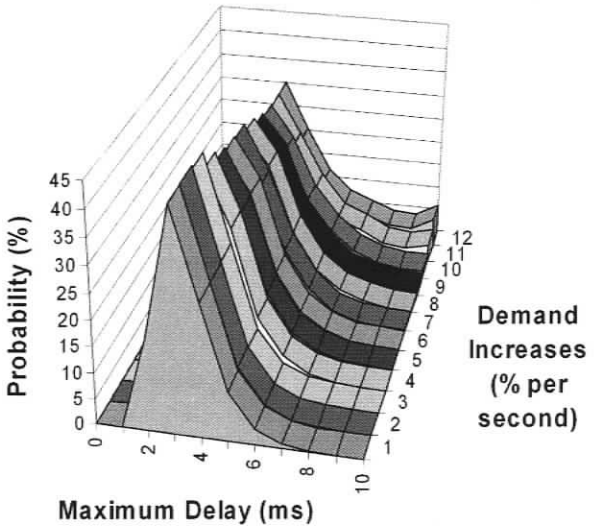


Figure 3.14: Probability of occurrence of the maximum delay for varying demand increases of multiplexed movie traffic.

We next test ULT's ability to respond to demand increases using a core traffic trace. Figure 3.15 shows that the average bandwidth demands of a QoS-Transit Service are not easily controlled when demand increases at a high rate for a long period of time. Figure 3.16 shows the percentage of time that packet loss exceeds 1%, which is assumed to be the maximum packet loss guaranteed by the QoS-Transit Service. We can see that the guarantee is not violated in more than 1% of the time intervals until demand increases at a rate greater than 8% per second, for a sustained period of time.

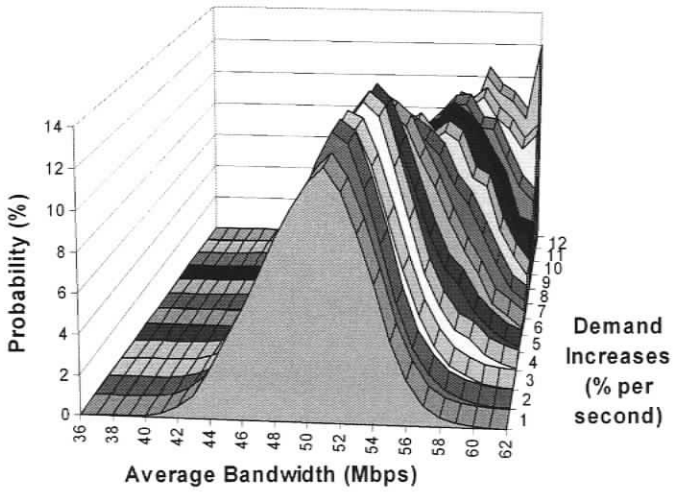


Figure 3.15: Probability of occurrence of the average bandwidth demands for varying demand increases of core link traffic.

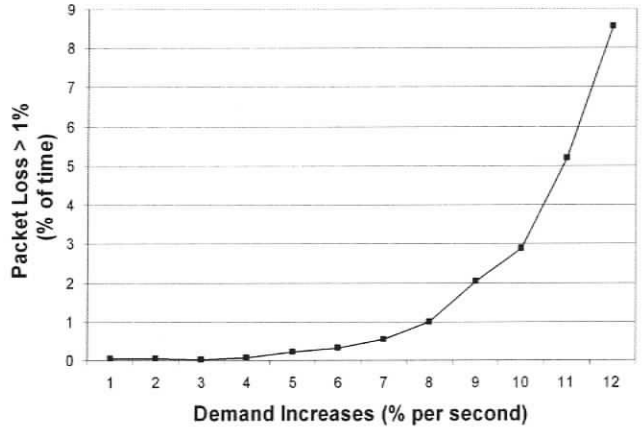


Figure 3.16: Percentage of time that packet loss exceeds 1% for varying demand increases of core link traffic.

We finally simulate the increase in demand for a bursty QoS-Transit Service with traffic consisting of 6 access link traffic traces. The aggregated traffic represents the traffic for a QoS-Transit Service from multiple customers. In Figure 3.17 we can see that the pricing mechanism is unable to control the average bandwidth demands for high rates of demand increases. As expected, the more demand increases, the more difficult it is for the pricing mechanism to maintain a designated QoS. If demand fluctuating at a high rate is expected, then either greater spare bandwidth needs to be maintained for the QoS-Transit Service, or lower performance should be guaranteed.

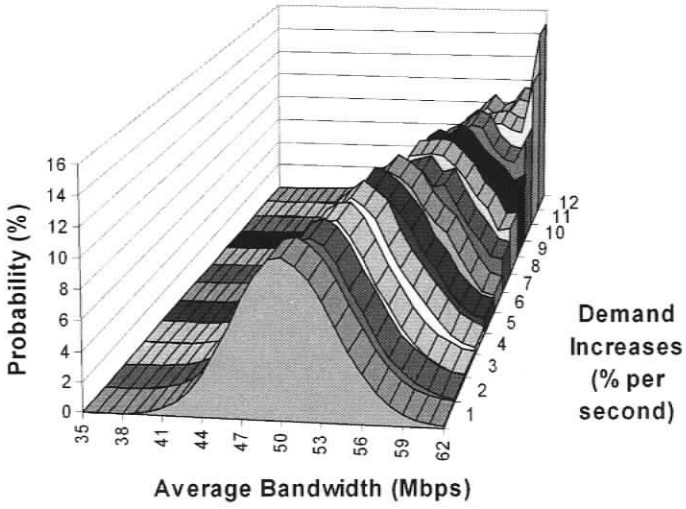


Figure 3.17: Probability of occurrence of the average bandwidth demands for varying demand increases of access link traffic.

Figure 3.18 shows the percentage of time that packets loss exceeds 1% for the multiplexed access link traffic. If the simulated QoS-Transit Service guarantees at most 1% packets loss, then the QoS-Transit Service violates that agreement less than 1% of the time even when demand quadruples, increasing at a rate of 4% per second.

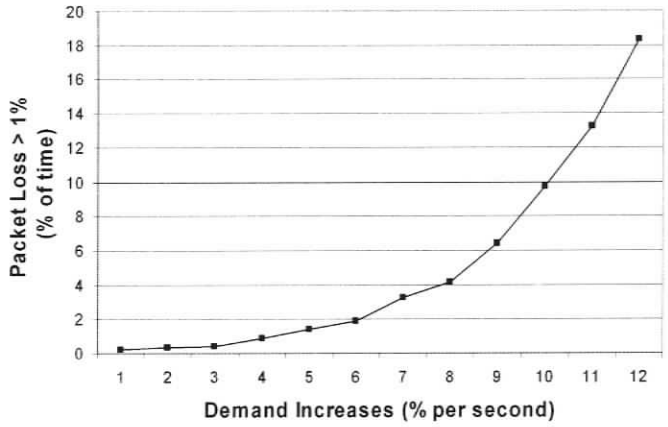


Figure 3.18: Percentage of time that packet loss exceeds 1% for varying demand increases of access link traffic.

The results here show great promise. Even a simple pricing mechanism, such as ULT, can properly respond to the quadrupling of traffic when demand increases at a rate of 4-10% per second. Demand increasing at a rate of 10% per second results in traffic doubling every 7.3 seconds. Adequate response to even more rapid changes in demand may be possible by altering ULT's parameters, or by modifying its design.

3.7 Summary and Conclusions

We have presented the concept of QoS-Transit Services to provide guaranteed QoS to traffic within networks. Using dynamic pricing as a feedback control mechanism to control the traffic ensures resources are not over-consumed. QoS-Transit Services are defined by maximum delay, maximum jitter, maximum packet loss, a statistical calculation period, and a tariff definition, between two points on a network.

By using dynamic pricing to control demand so that QoS can be guaranteed, a QoS-Transit Service transmits the traffic that values the service the most. While customers

who value their traffic more will send a larger proportion of packets to a QoS-Transit Service, a large number of customers will transmit at least a small amount of traffic. Using dynamic pricing does not block customers from using the QoS-Transit Services, and in this sense can be considered fair.

We gave two sample dynamic pricing mechanisms to guarantee the performance of QoS-Transit Services. The first was Reaction Point Pricing, which is a pricing mechanism similar to that proposed by Li et al. [18], which begins to adjust prices once demand increases past a specified point. We then introduced ULT, which uses iterative price adjustments to maintain the demand around a specified goal demand.

To conclude the chapter we evaluated the ability of ULT, a simple pricing mechanism, to guarantee the performance of a QoS-Transit Service. Our simulations demonstrated that posted prices should not change by too-large increments, as the resulting demand may be unstable over time. The price change increment should depend on the expected price elasticity of demand. The more inelastic demand is, the more prices need to change in order to affect that demand. Further, we examined the effects of multiplexing, and demonstrated that better performance can be guaranteed if traffic is multiplexed on a link. Finally, we determined that bursty traffic may be controlled, without copious amounts of spare bandwidth on a link, allowing for sustained increases in demand of 4-10% per second.

In the next two chapters we focus on how a single QoS-ISP can provision its QoS-Transit Services in order to maximize its revenue. In the next chapter we examine how revenue maximization can be done when a QoS-ISP can estimate the demand functions of its QoS-Transit Services. The subsequent chapter will then show how a QoS-ISP can maximize its revenue when the demand functions cannot be determined.

4 Optimal Provisioning of QoS-Transit Services with Demand Functions

The optimal provisioning of QoS-Transit Services offered by a QoS-ISP raises two problems: optimal bandwidth allocation given predetermined paths, and the optimal determination of those paths. The optimal allocation is defined as the allocation of bandwidth to QoS-Transit Services such that, at a set of stated prices, the revenue of the ISP is maximized.

Further, these problems are solved differently depending on whether the demand functions of the QoS-Transit Services can be reasonably estimated. Table 4.1 shows the division of the problems and the chapters in which they are studied.

Table 4.1: Division of the optimal provisioning problem.

	Optimal Bandwidth Allocation	Optimal Routing
With Demand Functions	Chapter 4.1	Chapter 4.2
Without Demand Functions	Chapter 5.1	Chapter 5.2

There exists an implicit relationship between bandwidth allocation and pricing, as illustrated in Figure 4.1. Each QoS-Transit Service is managed by a pricing mechanism that adjusts the service's price in order to manage demand, and thus the QoS guarantees received by the traffic transmitted via the service. If the bandwidth allocated to a QoS-Transit Service is changed, then the pricing mechanism will adjust prices in order to match the demand to the bandwidth allocated to that service. Therefore, the algorithms within these next two chapters do not address how to control congestion, but rather address how to optimally provision QoS-Transit Services – a set of congestion controlling services.

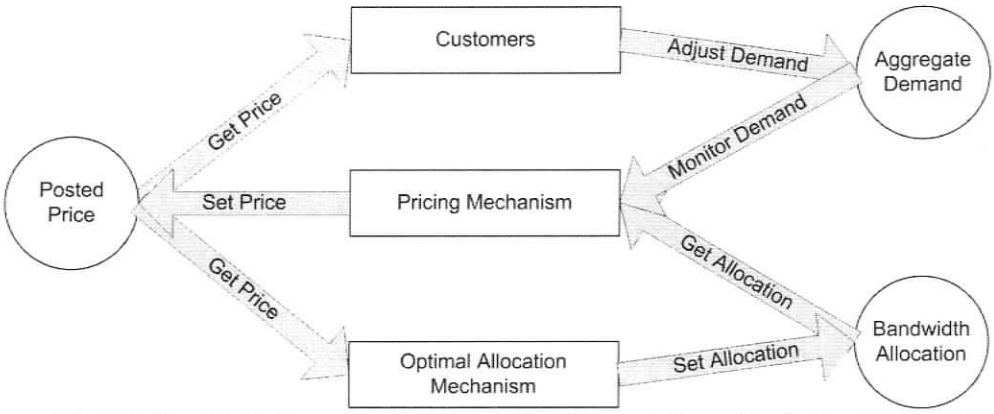


Figure 4.1: Relationship between the pricing mechanism and the optimal allocation mechanism.

For the approaches within this chapter to be used in practice, a QoS-ISP must be able to reasonably estimate (or know explicitly) the aggregate demand functions for the QoS-Transit Services it offers. A service delivers traffic for a large number of customers, making this assumption feasible, especially since dynamic pricing is used, and since the customers' reaction to pricing is constantly being monitored.

Throughout this dissertation we assume that the effective bandwidth [64,79] of a service can be determined through measurement. Effective bandwidth can be understood as the bandwidth that needs to be reserved for a session in order for the session to receive expected performance. For example, a QoS-Transit Service with high performance guarantees may require a large amount of allocated bandwidth in order to guarantee its stated performance in the event of traffic bursts. Additionally, we assume that a QoS-Transit Service has a measured effective bandwidth that is independent of the other QoS-Transit Services it may be multiplexed with on a link.

All prices within this chapter, and the next, are assumed to be normalized to a price per unit of effective bandwidth per unit of time (e.g. $/Mbps/minute). Note that posted prices for a QoS-Transit Services will not likely be posted in terms of effective bandwidth, but rather as actual bandwidth used.

4.1 Optimal Bandwidth Allocation

In this section we define and study several heuristics to find the optimal allocation of bandwidth for the QoS-Transit Services offered by a QoS-ISP, assuming that the services are routed on predefined paths, and the demand functions for each QoS-Transit Service can be reasonably estimated.

We first define the problem, and then define an optimal solution, which can be obtained by mapping the problem to a multiple-choice multidimensional knapsack problem (MMKP). Two known methods of solution, which were created for a similar problem, exist for the MMKP: Khan's HEU heuristic and Khan's Branch and Bound with Linear Programming (BBLP) exact algorithm [41]. (See Appendix C for a review of knapsack problems.)

We then define two heuristics that we have devised: the Iterative Greedy heuristic and the Iterative Linear Programming heuristic. In the Iterative Greedy heuristic, link bandwidth is greedily assigned in small increments to the set of QoS-Transit Services. The Iterative Linear Programming heuristic uses a similar approach, but a fraction of the link bandwidth is allocated to the QoS-Transit Services using linear programming.

4.1.1 Problem Definition

We define the *Optimal Bandwidth Allocation* problem to be the allocation of bandwidth within a QoS-ISP's network to a set of offered QoS-Transit Services, on predefined paths, so that the revenue of the QoS-ISP is maximized.

Let $p_q(\alpha_q)$ be the inverse demand function for QoS-Transit Service q. It defines the price per unit of effective bandwidth for QoS-Transit Service q that can be charged for when there is demanded effective bandwidth of α_q for the service. Mathematically, the QoS-ISP needs to solve the following problem:

$$\text{maximize} \quad \sum_{q \in Q} \alpha_q \cdot p_q(\alpha_q), \tag{4.1a}$$

$$\text{subject to} \quad \sum_{q \in Q} \alpha_q A_{q\ell} \leq C_\ell, \forall \ell \in L, \tag{4.1b}$$

where α_q is the effective bandwidth of QoS-Transit Service q, and $A_{q\ell}$ equals 1 if link ℓ is within the predefined path of QoS-Transit Service q, and it equals 0 if it is not. Additionally, Q is the set of QoS-Transit Services offered by the QoS-ISP, L is the set of links within the QoS-ISP's network, and C_ℓ is the capacity of link ℓ.

4.1.2 Optimal Solution

We assume that the marginal revenue curve is a non-increasing function, indicating that each additional unit of a commodity sold contributes a smaller amount of revenue than the previous unit. That is, the n^{th} unit of bandwidth is assumed to be more valuable than the $n+1^{\text{th}}$ unit of bandwidth. Since we are considering aggregate traffic, and not a single user's traffic, such a property is reasonable.

We map the Optimal Bandwidth Allocation problem to an MMKP (as described in Appendix C) to determine the exact solution. The mapping is as follows:

- Each group of items in the MMKP, i, corresponds to a QoS-Transit Service, $q \in Q$.
- Each resource of the MMKP, r, corresponds to a link in the network, $\ell \in L$.
- Each resource constraint of the MMKP, C_r, correspond to a link capacity of the network, C_ℓ.
- The decision variable of the MMKP, x_{ij}, corresponds to $d_{q,j}$, the decision to select the jth bandwidth allocation for QoS-Transit q.
- The value of each item of the MMKP, v_{ij}, corresponds to the value of a selecting the jth bandwidth allocation choice for QoS-Transit Service q, $v(q, j)$.

- The resource consumption of each item in the MMKP, $c_{ijr}, \forall r$, corresponds to the bandwidth allocation $\alpha_{q,j}$ for each link $\ell \in L_q$, where L_q is the set of links that define the path for QoS-Transit Service q.

We define $Cmin_q$ to be the smallest capacity of a link in L_q (i.e. $Cmin_q$ is the bottleneck capacity for QoS-Transit Service q). The bandwidth allocated to QoS-Transit Service q, if item j of its group is picked, is

$$\alpha_{q,j} = jz, \tag{4.2}$$

where

$$0 \le j \le \frac{Cmin_q}{z}. \tag{4.3}$$

z indicates the resolution of the our solution (e.g. $z = 2\text{Mbps}$ indicates that bandwidth will be allocated in 2Mbps blocks to the services). The global optimal solution is approached as $z \to 0$.

The value of each item in the MMKP is

$$v(q,j) = jz \cdot p_q(jz) = \alpha_{q,j} \cdot p_q(\alpha_{q,j}), \tag{4.4}$$

where p_q defines the inverse demand function for QoS-Transit Service q. Thus $p_q(\alpha_{q,j})$ indicates the price that $\alpha_{q,j}$ units of the service can be sold for and $\alpha_{q,j} \cdot p_q(\alpha_{q,j})$ indicates the resulting revenue for the QoS-ISP.

The optimal solution of the Optimal Bandwidth Allocation Problem is now defined as follows:

$$\text{maximize} \quad \sum_{q \in Q} \sum_{j=0}^{Cmin_q / z} d_{q,j} v(q,j), \tag{4.5a}$$

$$\text{subject to} \quad \sum_{q \in Q} \sum_{j=0}^{Cmin_q / z} d_{q,j} \alpha_{q,j} A_{q\ell} \leq C_\ell, \forall \ell \in L, \text{ and} \tag{4.5b}$$

$$\sum_{j=0}^{Cmin_q / z} d_{q,j} = 1, \forall q \in Q, \tag{4.5c}$$

where $d_{q,j} \in \{0,1\}$ indicates if item j of QoS-Transit Service q was picked (1) or not (0).

The MMKP consists of $\frac{nC}{z}+1$ items, where n is the number of QoS-Transit Services, z is the granularity of the solution, and C is the capacity of each of the links in the network (assuming that all links have the same capacity, for convenience). Recall that the optimal solution is approached as $z \to 0$, and note that for even modest values of z, the problem will be of large size.

4.1.3 Solving the Optimal Solution with Known MMKP Algorithms

Khan defined two algorithms for solving MMKPs under similar conditions: the HEU heuristic and the BBLP exact algorithm [41]. We describe these algorithms here since they are used to solve the above MMKP mapping of the Optimal Bandwidth Allocation problem, and can thus be used to determine the effectiveness of the heuristics we later develop.

Khan's algorithms are of interest because they were developed to solve the MMKP under conditions very similar to what we propose here. Khan studied admission control for a multimedia server, where the different items in a group related to different resource allocations for that session. This is very similar to our work if we consider a session to be a QoS-Transit Service. We, however, consider a much larger number of possible allocation choices for each session.

Khan defined a heuristic, named HEU, to solve MMKPs quickly. In HEU, an upgrade is defined by choosing a higher valued allocation for a session. HEU iteratively chooses the

best upgrade, where the *best upgrade* is defined as the upgrade that maximizes the value gain per unit of extra resources required. Heavily used resources are penalized in the calculation, allowing only high valued items to use heavily demanded resources. The complexity of HEU is $O(mn^2(C/z)^2)$, where m is the number of links in the network.

Khan also developed a branch and bound exact algorithm for determining the optimal solution of an MMKP. BBLP finds the optimal solution of an MMKP using a branch and bound approach, where the upper bound for branches is calculated through linear programming. BBLP thus finds the optimal solution to the Optimal Bandwidth Allocation problem, as $z \to 0$.

4.1.4 Iterative Greedy Heuristic

4.1.4.1 Description

In the Iterative Greedy (IterGreedy) heuristic, bandwidth is greedily allocated iteratively to the set of services. In each round, up to z units of each resource is allocated to the set of services. We therefore iteratively determine a set of services for which to upgrade resource allocations, where no service can be allocated more than z units along its path per iteration, bandwidth is allocated uniformly to all links of a path, and no more than z additional units of a resource are allocated per iteration. Due to our assumptions about the demand functions, smaller allocations to a QoS-Transit Service yield more revenue per unit resource.

The IterGreedy heuristic is defined as follows:

1. The effective bandwidth allocated to each service is set to 0 Mbps:

$$\alpha_q = 0, \forall q \in Q.$$

2. The effective bandwidth allocated on each link is set to 0 Mbps:

$$\alpha_\ell = 0, \forall \ell \in L.$$

3. Estimate the value of upgrading each service by z Mbps. Let the value of upgrading each service be

$$p_q(\alpha_q + z)\cdot(\alpha_q + z) - p_q(\alpha_q)\cdot\alpha_q . \tag{4.6}$$

4. Determine the services for which to increase the bandwidth allocation:
   a. Order the services in increasing order of their value
   b. Let the services be $q_1, q_2, ...q_n$, where q_1 is the lowest valued service, and q_n is the highest valued service.

   for $i = n$ to 1

   if each link $\ell \in L_{q_i}$ has not yet been allocated in this round, where L_{q_i} is the set of links that define the path for service q_i, then allocate z units of bandwidth from those links to q_i

5. Update the effective bandwidth variables for the services that were chosen to have their allocations increased:
   a. $\alpha_q \leftarrow \alpha_q + d_q z, \forall q \in Q,$ (4.7)

   where d_q is 1 if service q was chosen and 0 if it was not.

   b. $\alpha_\ell \leftarrow \alpha_\ell + d_q A_{q\ell} z, \forall \ell \in L_q, \forall q \in Q,$ (4.8)

   where $A_{q\ell}$ is 1 if service q is routed across link ℓ and 0 if it is not.

6. If any items were picked in step 4, then loop to step 3, otherwise stop.

IterGreedy may not find a near optimal allocation of bandwidth in some situations. Consider the network in Figure 4.2 with the three QoS-Transit Services shown. Assume that the links have equal capacities. If the marginal revenue of QoS-Transit Service q_1 is always slightly greater than that of services q_2 and q_3, then all the bandwidth of links A,B and B,C will be allocated to q_1, while the bandwidth of link C,A will not be assigned to any service. However, an optimal solution may result in an equal allocation of bandwidth to each of the services, using the bandwidth from all the links. This situation is important

to recognize because the heuristic may not perform well with certain topologies, such as ring topologies. The heuristic in the next section does not suffer from this limitation.

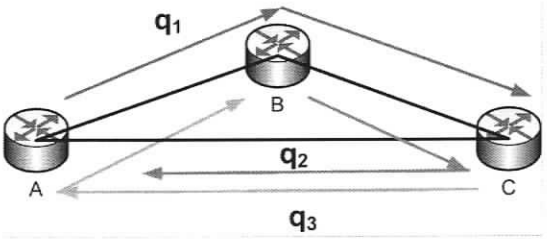


Figure 4.2: QoS-Transit Services with circular dependencies.

4.1.4.2 Complexity

Steps 1 and 3 of IterGreedy have a computational complexity of $O(n)$, while step 2 has a computational complexity of $O(m)$, where n is the number of QoS-Transit Services offered by a QoS-ISP, and m is the number of links in its network.

While the running time of step 4a depends on the sorting algorithm used, sorting can be performed in $O(n \log n)$ time. In step 4b, for each of the n QoS-Transit Services it is determined if any of the links that a service is routed on have had their bandwidth allocated yet this round. Step 4b can thus be performed in $O(n\varphi)$ time, where φ is the average number of links in the path of a QoS-Transit Service. Further, step 5a is solved in $O(n)$ time, and step 5b is solved in $O(n\varphi)$ time.

As defined by step 6, steps 3 – 5 are repeated until no items are picked in step 4. If we assume that all links have the same capacity, C, then bandwidth is assigned in chunks of C/z, where z is the resolution of our solution. If $m < n$, then in the worst case, only one link will have its chunk of bandwidth assigned per round, and thus there will be at most Cm/z iterations of steps 3 – 5. However, if $n < m$, then in the worst case, only one QoS-Transit Service will be assigned bandwidth each round, and thus there will be at most Cn/z iterations.

The overall computational complexity of IterGreedy is thus

$$O\left(\frac{C \cdot \min(n,m)}{z}(n\log n + n\varphi)\right).$$

4.1.5 Iterative Linear Programming Heuristic

4.1.5.1 Description

In this section we define the Iterative Linear Programming (IterLP) heuristic, which uses linear programming to solve the Optimal Bandwidth Allocation problem.

Let L_q be the set of links that define the path for QoS-Transit Service q. The IterLP heuristic iteratively allocates $1/b$ of the bandwidth of the links in the network to the set of QoS-Transit Services in order to maximize the revenue for the network provider, where b determines the number of iterations and the granularity of the solution.

The IterLP heuristic is defined as follows:

1. $\alpha_q = 0, \forall q \in Q$ and $loopcount = 0$.
2. Calculate $Cmin_q / b$ for each QoS-Transit Service q, where $Cmin_q$ is the capacity of the link with the smallest capacity in L_q and b is the number of rounds that will be executed for the heuristic.
3. $loopcount = loopcount + 1$.
4. Calculate the marginal revenue for each QoS-Transit Service. We calculate the marginal revenue, MR_q, as the change in revenue resulting from an increase in bandwidth of:

$$MR_q = (\alpha_q + Cmin_q / b) \cdot p_q(\alpha_q + Cmin_q / b) - \alpha_q \cdot p_q(\alpha_q). \quad (4.9)$$

5. For each link, allocate $z_\ell = C_\ell / b$ of the bandwidth of that link to the set of services. To determine the optimal allocations we solve the following linear programming problem:

$$\text{maximize} \quad \sum_{q \in Q} d_q MR_q, \quad (4.10a)$$

subject to $$\sum_{q \in Q} d_q A_{q\ell} \leq z_\ell, \forall \ell \in L, \quad (4.10b)$$

$$d_q \geq 0, \forall q \in Q, \quad (4.10c)$$

where d_q indicates the amount of additional bandwidth that QoS-Transit Service q is allocated.

6. Update the effective bandwidth variable for the QoS-Transit Services:

$$\alpha_q \leftarrow \alpha_q + d_q, \forall q \in Q. \quad (4.11)$$

7. If *loopcount* < b, then we still have resources left to allocate, so loop to step 3.

4.1.5.2 Complexity

Steps 1, 2, 4 and 6 of IterLP have a computational complexity of $O(n)$. Step 3 is performed in $O(1)$ time.

Step 5 is a linear programming problem. The setup of the array that describes the linear program in memory has a computational complexity of $O(nm)$, which involves the initializing of an $n \times m$ array, defining the objective function, and the defining of the constraints. The linear program is solved using a simplex algorithm, which in the worst case is $O(2^m)$. While examples can be constructed to demonstrate that the simplex algorithm may take exponential time to solve some problem instances [80], in practice, the algorithm performs in polynomial time [81]. Alternative algorithms do exist to solve linear programs in polynomial time [82], but are not considered as efficient as the simplex method for smaller problem instances.

Finally, as defined by step 7, steps 3 – 6 are repeated C/z times, assuming that all links have a capacity of C.

The overall computational complexity of IterLP is thus $O\left(\frac{C}{z}\left(nm+2^m\right)\right)$; however, recall that the exponential portion of the running time is the worst-case running time of the simplex method, which in practice will be much faster.

4.1.6 Results

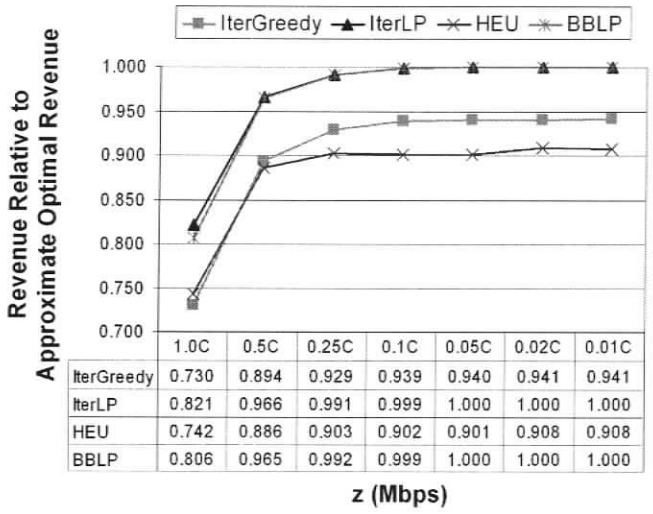
To test IterLP and IterGreedy, they are compared against Khan's HEU and BBLP [41] algorithms. The algorithms are tested on randomly created networks of two different sizes: a 10-node network with 15 links, and a 20-node network with 34 links. The capacity of all links is assumed to be constant, and denoted by C. Also, each service is routed on a shortest path between the two end-points that define the QoS-Transit Service.

The demand function for each QoS-Transit Service is a randomly created linear demand function, defined by two points. The first point, $x_q(0)$, is randomly chosen from the interval $(0,2\cdot|L_q|)$ and the second point, $x_q(2C)$ is randomly chosen from the interval $(0,7\cdot|L_q|)$, where $|L_q|$ is the number of links in QoS-Transit Service q's path. QoS-Transit Services using more links can thus have a higher value than those services using fewer links. More links may correspond to a further distance traveled, and thus a higher cost expectation.

Each of the solutions was coded in Java and simulated using a computer with an AMD Athlon64 2800+ CPU and 512MB of RAM. Additionally, the linear programming within IterLP was solved by *lp_solve* [83], with the default options.

We first determine how quickly BBLP approaches the optimal solution of the Optimal Bandwidth Allocation problem as $z \to 0$. We randomly create 20 services on a 10-node network and determine the optimal bandwidth allocation using each of the algorithms. In Figure 4.3 we compare the revenue that is achieved from the different algorithms using decreasing values of z. The z parameter, which was included in each of our described

algorithms, defines the granularity of the problem. Recall that the optimality of the solution should improve as $z \to 0$.



	1.0C	0.5C	0.25C	0.1C	0.05C	0.02C	0.01C
IterGreedy	0.730	0.894	0.929	0.939	0.940	0.941	0.941
IterLP	0.821	0.966	0.991	0.999	1.000	1.000	1.000
HEU	0.742	0.886	0.903	0.902	0.901	0.908	0.908
BBLP	0.806	0.965	0.992	0.999	1.000	1.000	1.000

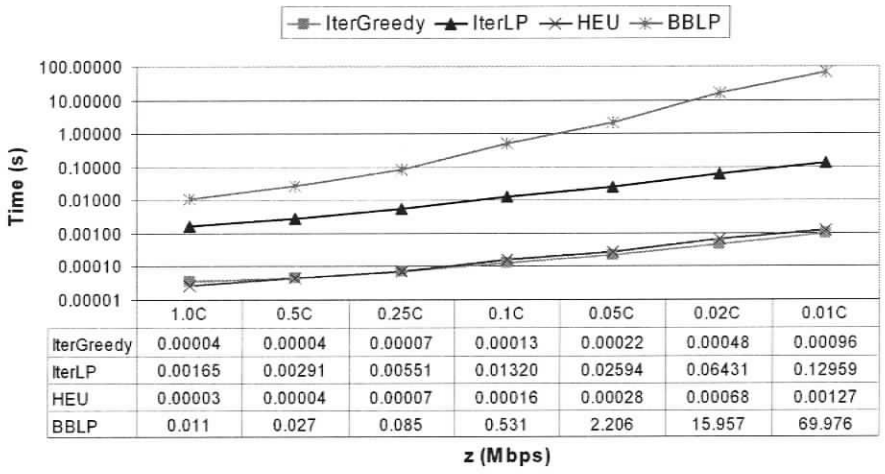
Figure 4.3: Average relative revenue as z approaches 0.

The z parameters can be considered as a percentage of the capacity (i.e. $z = 0.02C$ is 2% of the link capacity). The number of allocation choices in the MMKP is $\frac{C}{z}+1$. Thus, for each point along the x-axis in Figure 4.3 (and later in Figure 4.4), the number of allocation choices per service in the associated problem roughly double, because z roughly halves.

In Figure 4.3 we can see that the MMKP solution, as solved by BBLP, quickly approaches the optimal solution. For future comparisons, we approximate that BBLP using $z \leq 0.02C$ is optimal. It is interesting to see that IterLP can outperform BBLP when both use a large z parameter. This is because IterLP does not solve an MMKP, like BBLP does, but solves the Optimal Bandwidth Allocation problem in a different way. However, IterLP will not outperform BBLP as $z \to 0$.

Figure 4.4 indicates the time cost to increase the optimality of the results, due to decreasing the z parameter. IterGreedy and HEU execute very fast, and as seen in Figure 4.3, IterGreedy achieves higher revenue than HEU. Additionally, Figure 4.3 indicates that IterLP achieves approximately 99.99% of the result of BBLP for $z \leq 0.05C$ and

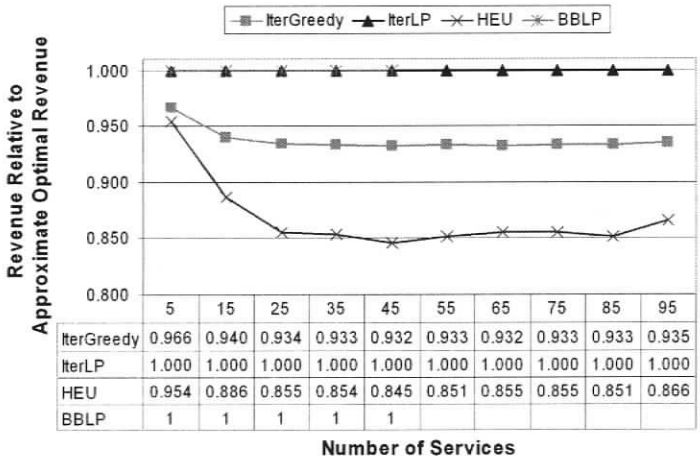
Figure 4.4 reveals that IterLP achieves these results efficiently. However, the running time of each algorithm grows exponentially as $z \to 0$.



	1.0C	0.5C	0.25C	0.1C	0.05C	0.02C	0.01C
IterGreedy	0.00004	0.00004	0.00007	0.00013	0.00022	0.00048	0.00096
IterLP	0.00165	0.00291	0.00551	0.01320	0.02594	0.06431	0.12959
HEU	0.00003	0.00004	0.00007	0.00016	0.00028	0.00068	0.00127
BBLP	0.011	0.027	0.085	0.531	2.206	15.957	69.976

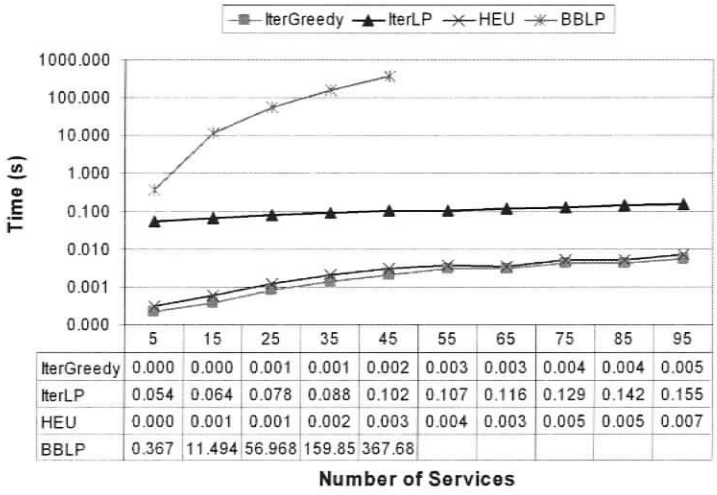
Figure 4.4: Average time to execute algorithms as z approaches 0.

We now compare the algorithms by varying the number of randomly created QoS-Transit Services on a 20-node network, using $z = 0.02C$. Figure 4.5 compares the revenues achieved by the algorithms for an increasing number of QoS-Transit Services to that of the optimal solution, where the optimal solution is deemed to be the solution of BBLP, or of IterLP when the problem size is too large for BBLP. Figure 4.6 shows the time required by each of the algorithms to achieve the results.



	5	15	25	35	45	55	65	75	85	95
IterGreedy	0.966	0.940	0.934	0.933	0.932	0.933	0.932	0.933	0.933	0.935
IterLP	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
HEU	0.954	0.886	0.855	0.854	0.845	0.851	0.855	0.855	0.851	0.866
BBLP	1	1	1	1	1					

Figure 4.5: Average relative revenue of algorithms against varying numbers of services.



	5	15	25	35	45	55	65	75	85	95
IterGreedy	0.000	0.000	0.001	0.001	0.002	0.003	0.003	0.004	0.004	0.005
IterLP	0.054	0.064	0.078	0.088	0.102	0.107	0.116	0.129	0.142	0.155
HEU	0.000	0.001	0.001	0.002	0.003	0.004	0.003	0.005	0.005	0.007
BBLP	0.367	11.494	56.968	159.85	367.68					

Figure 4.6: Average time to execute algorithms against varying number of services.

Each heuristic results in high optimality for a small number of services due to the lack of resource contention. When considering more than 15 services, IterGreedy achieves about 93% optimality, which is much higher than the 86% optimality achieved by HEU, and does so in approximately the same amount of time. IterLP achieves near-optimal results, due to the fractional bandwidth allocation of the linear programming.

While IterLP executes very fast relative to BBLP, there may be times when a heuristic that executes even faster is required. For example, to find the optimal paths to route the services on, the optimal bandwidth allocation (and thus the resulting revenue) may need to be calculated for a large number of path combinations. IterGreedy may be the preferred heuristic for such a problem.

We now determine the effects of network size and the *average degree of connectivity* on the algorithms. The *average degree of connectivity* of a network is equal to the computed average degree of the vertices of the network. We execute the algorithms on networks with varying average degrees of connectivity, where the networks with higher average degrees of connectivity also contain more links. The lowest average degree of connectivity for a connected 10-node network is 1.8: a network with a star topology. A *full mesh network*, in which all nodes are directly connected to all other nodes, conversely, has the highest average degree of connectivity, which for a 10-node network is 9. We thus randomly created 10-node networks that have an average degree of

connectivity of 2, 3, 4, 5, 6, 7, 8, and 9. The number of links within each network is equal to the average degree of connectivity of the networks multiplied by 5.

Figure 4.7 indicates the amount of revenue that can be achieved by the heuristics for the different 10-node networks, where 90 QoS-Transit Services are offered and $z = 0.02C$. As the average degree of connectivity increases, so does the optimality of each of the heuristics. The lower the average degree of connectivity of the network is, the more links (on average) that are contained in the path of a QoS-Transit Service, and the more important the heuristics are in solving the problem to achieve higher revenues. For the full mesh network, all heuristics achieve optimal results, due to the simple problem instances in which each QoS-Transit Service is routed across a single link.

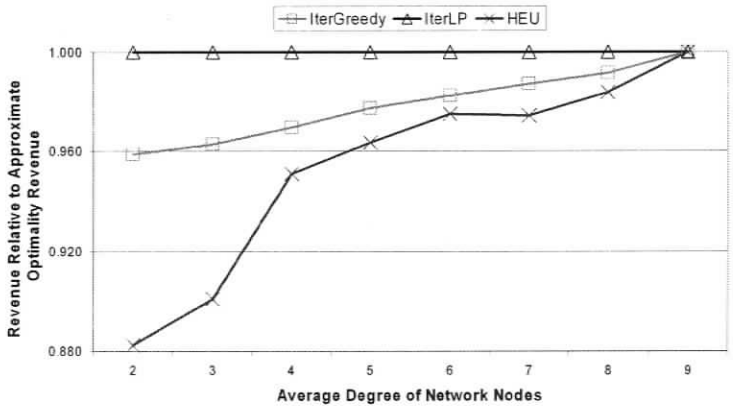
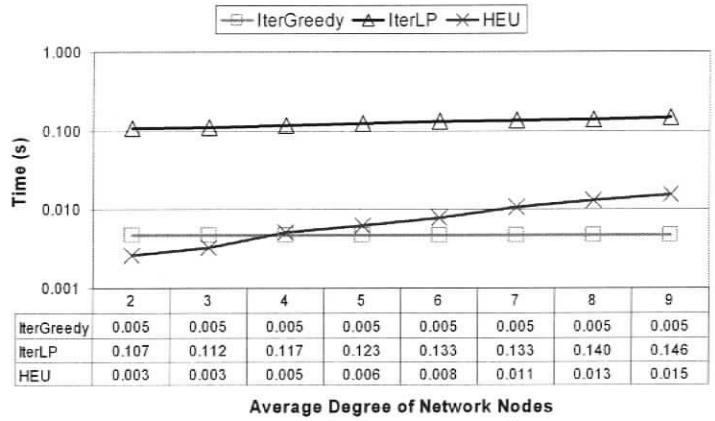


Figure 4.7: Average relative revenue of algorithms against varying average degrees of connectivity.

The time required by the heuristics to calculate the results of Figure 4.7 is shown in Figure 4.8. The running time of IterLP and HEU appear to increase linearly with respect to the number of network links being considered, while the running time of IterGreedy appears to be constant with respect to the number of links in the network. It is interesting to note that it is not even feasible to calculate optimal revenues with BBLP for the full mesh network, yet the results achieved quickly by the heuristics are still estimated to be optimal.



	2	3	4	5	6	7	8	9
IterGreedy	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
IterLP	0.107	0.112	0.117	0.123	0.133	0.133	0.140	0.146
HEU	0.003	0.003	0.005	0.006	0.008	0.011	0.013	0.015

Figure 4.8: Average time to execute algorithms against varying network connectivity with a constant number of QoS-Transit Services.

4.2 Optimal Routing

While the QoS-Transit Services of a network may be optimally allocated on a set of predefined paths, the paths on which they are routed across may not be optimal. In this section we define how a QoS-ISP can determine how to best route and allocate bandwidth for its offered QoS-Transit Services. We assume a service is routed on only one path at a time.

4.2.1 Problem Definition

A *path vector* is a vector that defines the paths that each QoS-Transit Service is routed on. We define a *path vector* $\overline{R}$ as:

$$\overline{R} = (path_1, path_2, path_3, ...path_n), \tag{4.12}$$

where $path_q$ is the path that QoS-Transit Service q is routed on.

A path vector is *optimal* if there is no other path vector that would result in higher revenue for the QoS-ISP, assuming that QoS-Transit Services can be optimally allocated along their defined paths.

The process of finding the optimal path vector to route QoS-Transit Services consists of two steps:

1. Identifying possible optimal path vectors, and
2. Optimally allocating bandwidth to each path vector.

The two steps are referred to as the problems of *Identifying Possible Optimal Path Vectors* and *Optimal Bandwidth Allocation*, respectively. The identified path vector with the highest revenue, as determined by an Optimal Bandwidth Allocation heuristic (e.g. IterGreedy or IterLP), defines the paths to best route each QoS-Transit Service.

Let $p_q(\alpha_q)$ be the inverse demand function for QoS-Transit Service q, which defines the value per unit of bandwidth for service q, when there is demand of α_q. Formally, the QoS-ISP needs to solve Equation 4.1, except that it must not only solve for α_q, but for $A_{q\ell}$ as well. That is, it must solve:

$$\text{maximize} \quad \sum_{q \in Q} \alpha_q \cdot p_q(\alpha_q), \tag{4.13a}$$

$$\text{subject to} \quad \sum_{q \in Q} \alpha_q A_{q\ell} \leq C_\ell, \forall \ell \in L, \tag{4.13b}$$

where α_q is the effective bandwidth of QoS-Transit Service q, and $A_{q\ell}$ equals 1 if link ℓ is within the path of QoS-Transit Service q, and it equals 0 if it is not. Additionally, Q is the set of QoS-Transit Services offered by the QoS-ISP, L is the set of links within the QoS-ISP's network, and C_ℓ is the capacity of link ℓ. The problem in this section is to select a route and allocation of bandwidth for each QoS-Transit Service so that the maximum profitability of the QoS-ISP is realized.

4.2.2 Optimal Solution

An optimal solution is to consider all possible path vectors. The path vector that results in the highest revenue, as determined by solving the Optimal Bandwidth Allocation

problem for each path vector, represents the best paths to route the QoS-Transit Services. However, the total number of possible path vectors is k^n, where n is the number of QoS-Transit Services offered by the QoS-ISP and k is the number of paths to consider for each QoS-Transit Service.

The number of path vectors is extremely large for even modest values of k and n. For example, if $k = 3$ and $n = 50$, then

$$k^n = 717,897,987,691,852,588,770,249. \tag{4.14}$$

Since we do not know of the existence of a fast exact algorithm for this problem, we choose to seek heuristics to identify possible optimal path vectors, thus reducing the total number of path vectors for which to solve the Optimal Bandwidth Allocation problem.

4.2.3 Identifying Possible Optimal Path Vectors

Since we can determine the value of routing QoS-Transit Services on given paths (as shown in Section 4.1), such as defined by a path vector, the remaining problem is how to determine which path vectors to compare. If k paths are considered for each of the n QoS-Transit Services, then k^n different path vectors exist. We define and evaluate three heuristics for identifying possible optimal path vectors: *Service Grouping*, *Iterative Bottleneck Avoidance*, and *Iterative Bottleneck Avoidance with Tabu*.

In the *Service Grouping* heuristic, the QoS-Transit Services are segmented into groups. Each group is assigned, *a priori*, a portion of the capacity of each link within the network. The optimal paths are determined for each group by considering all possible path vectors within the group, while being constrained by the assigned capacity.

In the *Iterative Bottleneck Avoidance* (IBA) heuristic, QoS-Transit Services are iteratively assigned to paths in order to avoid bottlenecks; highly valued QoS-Transit Services are assigned to paths first. The *IBA with Tabu* (IBA-Tabu) heuristic improves on the solution found with IBA by employing a *tabu search* [84] to further explore the neighboring solution space for a better solution.

4.2.4 Service Grouping Heuristic

Instead of solving the Optimal Bandwidth Allocation problem for all path vectors, the Service Grouping (SG) heuristic groups the QoS-Transit Services into g groups and solves the Optimal Bandwidth Allocation problem for all possible path vectors for each of the groups. Each group is assigned a fraction of the capacity of each link. The paths defined by the best path vector in each group are considered to be the best paths for the QoS-Transit Services in that group.

Instead of solving the Optimal Bandwidth Allocation problem for all k^n paths, we now calculate the optimal allocation for

$$gk^{n/g} \tag{4.15}$$

path vectors. For example, consider $g = 4$. We independently determine the optimal path vectors for QoS-Transit Services 1 to $n/4$, $n/4+1$ to $n/2$, $n/2+1$ to $3n/4$, and $3n/4+1$ to n. There are g groups, where the number of path vectors for each group is

$$k^{n/g}, \tag{4.16}$$

and thus Equation 4.15 defines the total number of path vectors considered. The number of path vectors considered is reduced by a factor of

$$\frac{1}{g}k^{n-n/g}. \tag{4.17}$$

Now consider if $k = 3$ and $n = 50$, then, by considering 5 groups (i.e. $g = 5$), the total number of path combinations that we will solve the Optimal Bandwidth Allocation problem for will be

$$gk^{n/g} = 295{,}245, \tag{4.18}$$

which is much smaller than if all the path combinations were considered, as was computed in Equation 4.14.

For this approach to work effectively, link capacities must be efficiently shared among the groups. We assign the bandwidth on a link to the groups based on the number of QoS-Transit Services within each group that use that link. Additionally, we give a higher weight to those groups containing QoS-Transit Services that use the link within their shortest path, rather than in one of their alternate paths.

Let Q be the set of all QoS-Transit Services and $Q_i \subseteq Q$ be the set of QoS-Transit Services within group i, where $1 \le i \le g$. Let $w(k')$ be the weight of the k' th shortest path, where $1 \le k' \le k$. For example,

$$w(k') = \begin{cases} 2(3-k'), & k' = 1, 2, \text{or } 3 \\ 0, & k' > 3 \end{cases}. \tag{4.19}$$

Define $c(\ell, k', i)$ to be the total number of services in group i that contain link ℓ in their k' shortest path. We can say that group i has a weight of $w(\ell, i)$ for link ℓ, where

$$w(\ell, i) = \sum_{k'=1}^{k} w(k') \cdot c(\ell, k', i). \tag{4.20}$$

The percentage of link ℓ 's capacity that will be assigned to group i is then the proportion of group i's weight relative to that of all the other groups:

$$\frac{w(\ell, i)}{\sum_{g'=1}^{g} w(\ell, g')}. \tag{4.21}$$

The Optimal Bandwidth Allocation problem is solved for all $k^{n/g}$ possible path vectors in each of the g groups, using heuristics such as IterGreedy and IterLP. The path vector in each group that results in the highest revenue, as determined by the Optimal

Bandwidth Allocation solutions, identifies the paths on which to route the QoS-Transit Services in that group.

The SG heuristic only determines the paths to route QoS-Transit Services. It does not define actual optimal bandwidth allocations. The overall expected revenue, and the associated optimal bandwidth allocations, is finally determined with an Optimal Bandwidth Allocation heuristic (e.g. IterLP), by using the path vector that results from combining the optimal path vectors from each of the groups.

4.2.5 Iterative Bottleneck Avoidance Heuristics

The IBA heuristic finds a single path vector to be used by an Optimal Bandwidth Allocation heuristic. In IBA-Tabu, the solution of IBA is used as the starting location for a *tabu search* heuristic that explores the solution space for better solutions.

Tabu searches are meta-heuristics that, in the general case, use short-term memory to overcome solutions with local optima [84]. The idea is to allow non-improving changes to a locally optimal solution in an effort to find higher, non-local, optima.

4.2.5.1 Iterative Bottleneck Avoidance

Let the number of links in the network be m, the number of services be n, and the average number of links used by each service be b. Assuming a fully occupied network, links will on average carry nb/m QoS-Transit Services.

We start by determining the path for each service that has the largest *bottleneck bandwidth*. The *bottleneck bandwidth* for a given path is the bandwidth of the link, within the path, that has the smallest unallocated bandwidth. It is thus the amount of bandwidth that is available to be allocated along that path to a service.

We then determine how much revenue each service would generate if it were allocated an amount of bandwidth proportional to its largest bottleneck bandwidth. Those services that generate the most revenue are routed on their preferred path, the available bandwidth

is accordingly reduced, and we assign more services to paths using the same process. Let P_q be the set of k shortest paths that QoS-Transit Service q can be routed on. The IBA heuristic is defined as follows:

1. Let $Q' = Q$, where Q is the set of QoS-Transit Services offered by an ISP and Q' is the set of QoS-Transit Services that have paths that remain to be chosen.
2. Let $C'_\ell = C_\ell$, where C_ℓ is the capacity of link ℓ, and C'_ℓ is its remaining bandwidth.
3. For each service $q \in Q'$
   a. Find $r_q \in P_q$, such that $\forall r'_q \in P_q$, $bottleneck(r_q) \geq bottleneck(r'_q)$, where $bottleneck(r)$ is the smallest remaining capacity along path r.
   b. Let $x_q = \dfrac{bottleneck(r_q)}{nb/m}$. (4.22)
   c. Calculate $p_q(x_q)$, where $p_q(\cdot)$ is the inverse demand function for QoS-Transit Service q, and thus $p_q(x_q)$ is the price per unit of bandwidth for the service, when there is demand of x_q.
4. Let Q'' be a set consisting of $|Q|/c$ QoS-Transit Services from Q' which have the c highest calculated $p_q(x_q)$ along their chosen route r_q.
5. Remove the items of set Q'' from set Q' and adjust the available link bandwidth for each of the links:

   $$\forall q \in Q'', \forall \ell \in r_q,\ C'_\ell \leftarrow C'_\ell - x_q . \quad (4.23)$$

   That is, the highly valued services are assigned to a path, and the remaining bandwidth of the links in their paths is reduced.

6. Loop to step 3 while $Q' \neq \phi$.

4.2.5.2 Tabu Searching

In IBA-Tabu we let the path vector of the initial solution be the solution found with IBA. In IBA-Tabu, the best solution is sought in the *neighbourhood* of the current solution. The *neighbourhood* of a path vector, $\overline{R}$, is the set of all path vectors that differ from $\overline{R}$

by exactly one path. The differing path is also restricted to be the next shortest, or the next longest path for its associated QoS-Transit Service. $\overline{R}$ is thus not in the neighbourhood of itself.

In tabu searching, certain moves within the neighbourhood are not allowed. Let $N(\overline{R})$ be the set of allowable neighbouring solutions to $\overline{R}$. $N(\overline{R})$ equals the neighbourhood of $\overline{R}$ minus those neighbours that result from modifying the paths of the QoS-Transit Services in the set T. T is *a tabu list* that contains a list of the QoS-Transit Services that have recently had a change in their paths resulting in an improved solution.

The *tabu list* ensures that new areas of the search space are explored, and prevents the immediate return to a previous solution. Items placed in a tabu list have a *tabu tenure*. A *tabu tenure* defines the number of rounds for which an item will remain in the tabu list.

For IBA-Tabu, we consider a tabu tenure of 5 rounds, and the heuristic is terminated if no better solution is found within the last 20 rounds.

Diversification was attempted, searching other areas of the search space, but very little improvement was found for even moderately sized problem instances.

4.2.6 Results

We first approximate the optimality of the IBA-Tabu heuristic, for comparing all heuristics against when considering larger problem instances. To approximate the optimality of IBA-Tabu, we compare it against the optimal routing solution for small numbers of QoS-Transit Services on randomly created networks of 4 different sizes: a 5-node, 7-link network; a 10-node, 15-link network; a 20-node, 34-link network; and a 30-node, 52-link network.

We assume that each QoS-Transit Service can be routed on any of its k shortest paths, where $k = 3$ (in [85] it was observed that little is gained from considering more than 3 or 4 paths). The optimal solution evaluates all possible path vectors (for $k = 3$) using the

IterGreedy heuristic. It is technically not optimal, but is our best approximation that can be reasonably calculated.

Figure 4.9 shows that IBA-Tabu achieves about 98% of the optimal solution when considering 12 QoS-Transit Services. The trend within the figure does not imply a sharp drop in the loss of optimality as the number of QoS-Transit Services increases. Optimality dropped approximately 1% when the number of services was increased 6-fold from 2 QoS-Transit Services to 12 QoS-Transit Services.

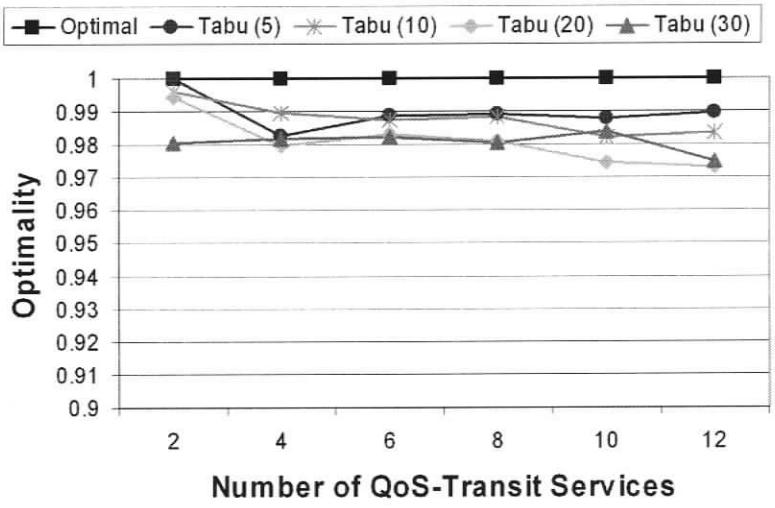


Figure 4.9: Optimality of IBA-Tabu for a small number of QoS-Transit Services on networks of different sizes.

We now evaluate the performance of the optimal routing heuristics for a varying number of QoS-Transit Services, using the 30-node network. Additionally, we use a group size of 10 services for the SG heuristic. The execution time grows exponentially with the size of the group, and a group of 10 services is near the limit of what can be practically solved.

As we can see in Figure 4.10, SG achieved the highest revenue for the ISP when the number of services was less than 50. IBA-Tabu greatly outperformed IBA when fewer than 50 services were considered as well, but executed much slower than IBA, though performed quicker than SG, as shown in Figure 4.11.

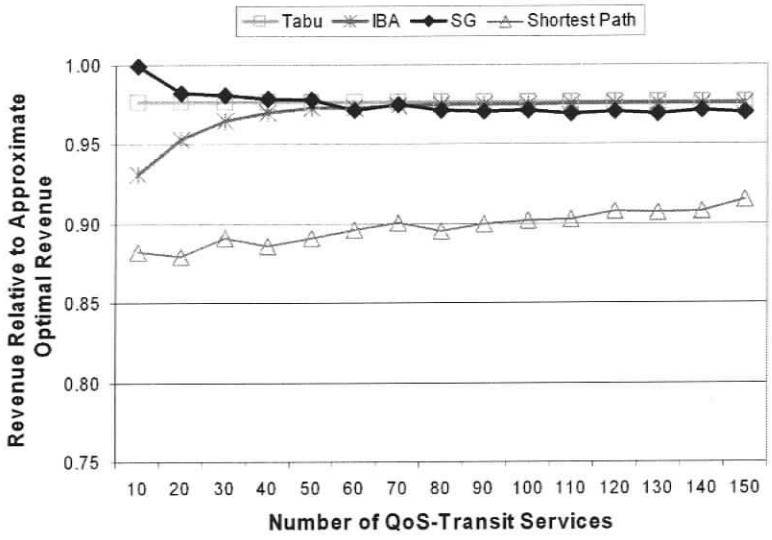
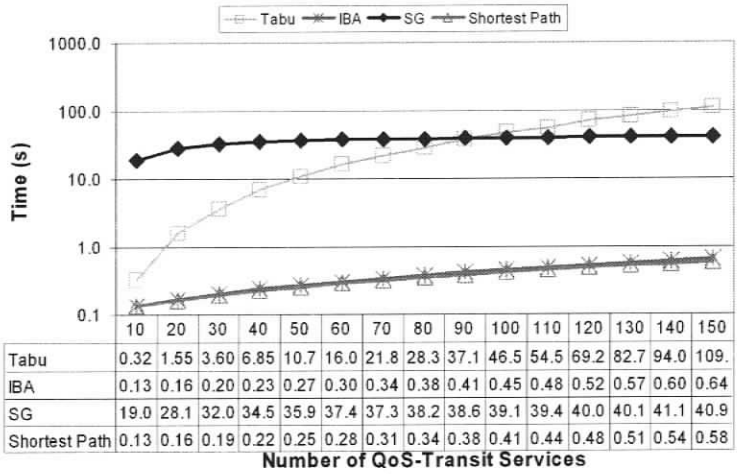


Figure 4.10: Average relative revenues resulting from the heuristics.



	10	20	30	40	50	60	70	80	90	100	110	120	130	140	150
Tabu	0.32	1.55	3.60	6.85	10.7	16.0	21.8	28.3	37.1	46.5	54.5	69.2	82.7	94.0	109.
IBA	0.13	0.16	0.20	0.23	0.27	0.30	0.34	0.38	0.41	0.45	0.48	0.52	0.57	0.60	0.64
SG	19.0	28.1	32.0	34.5	35.9	37.4	37.3	38.2	38.6	39.1	39.4	40.0	40.1	41.1	40.9
Shortest Path	0.13	0.16	0.19	0.22	0.25	0.28	0.31	0.34	0.38	0.41	0.44	0.48	0.51	0.54	0.58

Figure 4.11: Average time to execute heuristics.

When more than 50 services were considered, IBA-Tabu and IBA achieved more revenue than SG. As more services were considered, the revenue achieved by IBA approached that achieved by IBA-Tabu, without the considerable time required by the tabu search.

It is observed that the revenue achievable by routing QoS-Transit Services on their shortest paths increases, relative to the other heuristics, as we consider more services. As the number of services increases, so does the number of services routed over any particular link. Optimal routing is thus more important when fewer QoS-Transit Services

are considered. If there are only a few QoS-Transit Services offered by an ISP, then the relative gain to the ISP for rerouting one may be high.

To determine the effects of network connectivity on the heuristics, we execute the heuristics on networks with varying average degrees of connectivity. Similar to Section 4.1.6, we test the heuristics on randomly created 10-node networks with average degrees of connectivity of 2, 3, 4, 5, 6, 7, 8, and 9. The number of links within each network is equal to the average degree of connectivity of the network multiplied by 5. We test the heuristics on each of the networks using 90 QoS-Transit Services with $z = 0.02C$.

As the average degree of connectivity increases, the difference in achieved revenue from routing on the shortest path and the other heuristics is greatest when the average degree of connectivity of the network is 4, as shown in Figure 4.12. If a network has a low average degree of connectivity, there is little choice in routing, and thus the routing heuristics cannot greatly increase the revenue of the ISP. Similarly, when the average degree of connectivity is great, such as for a full mesh network, all QoS-Transit Services can be routed on their shortest path, and thus the routing heuristics cannot aid much in improving the revenue of the ISP. However, between these extremes, the benefits of the routing heuristics are greatest.

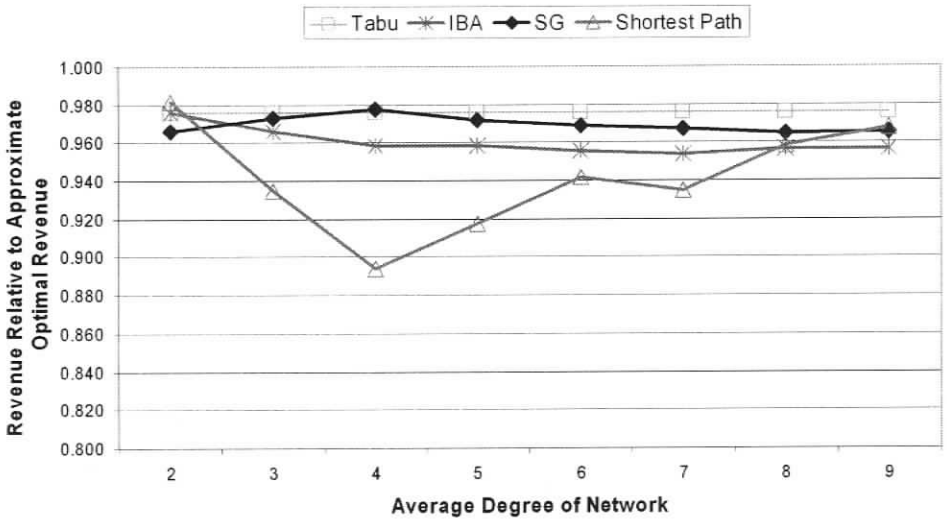
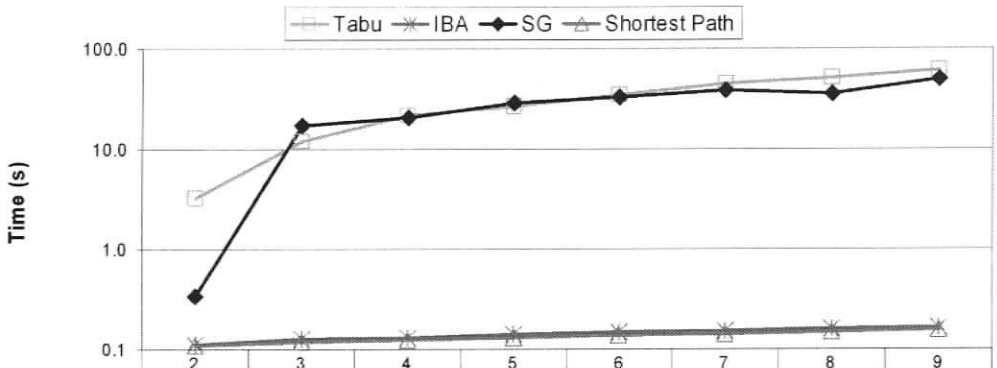


Figure 4.12: Average relative revenue of heuristics against varying network connectivity.

The time to execute each of the heuristics is shown in Figure 4.13. The running time of each of the heuristics appears to grow only linearly with respect to the number of links in a network. However, it is apparent that the time cost for IBA-Tabu and SG on networks with high average degrees of connectivity may not be worth the small revenue gains that can be achieved over IBA, or by routing QoS-Transit Services on their shortest path.



	2	3	4	5	6	7	8	9
Tabu	3.260	11.912	21.416	26.675	34.084	44.664	50.443	60.906
IBA	0.112	0.126	0.131	0.139	0.146	0.152	0.159	0.165
SG	0.341	17.178	20.729	28.628	33.032	37.799	35.851	49.674
Shortest Path	0.109	0.118	0.122	0.129	0.137	0.141	0.148	0.154

Average Degree of Network

Figure 4.13: Average time to execute heuristics against varying network connectivity.

4.2.7 Realizations

Even if a QoS-ISP can estimate demand functions, bandwidth allocations cannot simply be readjusted for its QoS-Transit Services. First, if a service is being moved to a new path where it will be allocated less bandwidth, it must have its bandwidth slowly reduced before being moved so that the advertised QoS for that service is not disrupted. Secondly, rerouting a QoS-Transit Service to a new path is plagued with difficult engineering issues. If the new path is shorter than the original path, packets may arrive out of order, and jitter constraints may be violated. Also, if many QoS-Transit Services are switched at once, the routers responsible for this transition must be synchronized to assure bandwidth is not over-consumed at any moment, adversely affecting any QoS guarantees.

Our results also do not include cross-elasticity effects, nor do they reflect that the actual performance received by a customer may, on average, be well within the performance guarantees of the QoS-Transit Services. By rerouting a QoS-Transit Service, the demand for that service may also change as its average performance characteristics also change.

4.3 Summary and Conclusions

An ISP offering dynamically priced services, such as QoS-Transit Services, can benefit greatly from allocating its network bandwidth to its services in order to maximize its revenue. We presented two heuristic solutions that solve the Optimal Bandwidth Allocation problem when the demand curves of the services are known, or can be reasonably estimated. Since a service delivers traffic for a large number of customers, and because prices may fluctuate in order to adjust demand, determining the demand function for a service may often be feasible.

To determine the optimality of our heuristic solutions, we map the Optimal Bandwidth Allocation problem to the multiple choice multidimensional knapsack problem. The optimal solution is approached as the number of items in a group, of the mapped MMKP, approaches infinity. Our first heuristic, IterLP, achieves over 99% of the optimal solution while still taking less than 2 seconds to execute for even large problem sets. By comparison, the optimal solution took about 2 minutes to solve, using a branch and bound approach, for medium sized data sets.

The second heuristic, IterGreedy, achieves approximately 90-95% of the optimal solution, but executes in even less time than IterLP. IterGreedy may be used when the optimal allocation may be sought for a large number of routing configurations.

While the QoS-Transit Services of a network may be optimally allocated on a set of predefined paths, the paths on which they are routed across may not be optimal. We developed and evaluated three heuristics to determine the optimal routing of these services: SG, IBA, and IBA-Tabu. We recommend using IBA-Tabu and/or SG to

calculate the optimal paths to route the services, unless evaluation time is important, in which case IBA should be used. IBA performs extremely well relative to the other heuristics, especially when 20 or more services are considered, achieving over 98% of the revenue of IBA-Tabu. IBA-Tabu achieved approximately 98% of the optimal revenue for small problem sizes of 12 services and less.

IBA executed in under 1 second for the test cases presented within this dissertation, whereas IBA-Tabu and SG took nearly a minute for 100 services. The short time to solve this problem allows the possibility of dynamic routing. Finally, we observed that optimal routing has a greater effect when an ISP offers a small number of QoS-Transit Services.

5 Optimal Provisioning of QoS-Transit Services without Demand Functions

In this chapter we examine the optimal provisioning of the QoS-Transit Services offered by a QoS-ISP when the demand functions of those QoS-Transit Services cannot be reasonably estimated. The optimal allocation of bandwidth, where QoS-Transit Services are routed on predetermined paths, is done using the Iterative Allocation Adjustment heuristic, which iteratively improves the solution based on the current posted prices of the QoS-Transit Services. We then demonstrate how a QoS-ISP can determine the value of links in order to identify rerouting possibilities, or to identify paths on which new services can be routed, in order to increase the ISP's revenue.

5.1 Optimal Bandwidth Allocation

This section describes an iterative process to arrive at the *optimal bandwidth allocation* for each service, without knowing the underlying demand functions of the QoS-Transit Services. The *optimal bandwidth allocation* is defined as the allocation of bandwidth to services so that, at the current posted prices, the revenue of the QoS-ISP is maximized. QoS-Transit Services are assumed to remain routed on predetermined paths.

5.1.1 Problem Definition

In Section 4.1 we examined the Optimal Bandwidth Allocation problem when the demand functions for the QoS-Transit Services could be estimated. Without knowledge of the demand functions, the only valuation we have for the services are their current posted prices. Using the implicit relationship between a pricing mechanism and a bandwidth allocation mechanism, as shown in Figure 4.1, the Iterative Allocation Adjustment heuristic considers the current posted prices to iteratively approach an optimal solution.

5.1.2 Iterative Allocation Adjustment Heuristic

The Iterative Allocation Adjustment (IAA) heuristic determines near-optimal bandwidth allocations for the offered QoS-Transit Services using a process based on tatonnement.

We assume that the marginal revenue of each service can be estimated. Marginal revenue could be estimated from the most recent price change, or it could be simply estimated as the current price per unit. Given a price vector $\overline{p}$ such that the resources of the network are not over-consumed, we can iteratively lower the price of a subset of the services and raise the price of another subset of the services, in order to increase the total revenue of the system. If the total revenue of the system cannot be raised through such price adjustments and marginal revenues can be accurately determined, then the system has an optimal allocation of resources. (Note that networks facing continuous changes in demand will not remain in a stable state of optimal allocation.)

Let v_q be the marginal revenue (the value) of QoS-Transit Service q. We define the IAA heuristic as:

1. Determine the marginal revenue for each of the QoS-Transit Services.
2. Maximize the marginal revenue gain obtained by adjusting QoS-Transit Service q's bandwidth allocation, for each QoS-Transit Service, by at most b_q Mbps, where

$$b_q = \begin{cases} b_{\max}, & \sigma\alpha_q \geq b_{\max} \\ \sigma\alpha_q, & \sigma\alpha_q < b_{\max} \end{cases}, \tag{5.1}$$

and where σ is the maximum percentage change per round for a QoS-Transit Service's allocation, up to at most $b_{\max}$ units of bandwidth. This can be solved by linear programming:

$$\text{maximize} \quad \sum_{q \in Q} v_q x_q \tag{5.2a}$$

$$\text{subject to} \quad \sum_{q \in Q} A_{q\ell} x_q \leq C_\ell - \sum_{q \in Q} A_{q\ell} \alpha_q, \forall \ell \in L, \text{ and} \tag{5.2b}$$

$$-b_q \le x_q \le b_q, \forall q \in Q, \tag{5.2c}$$

where the solved x_q is the change in bandwidth allocation to service q in order to maximize revenue, given the constraints on maximum allocation changes per round.

3. Adjust the allocation for each QoS-Transit Service q, by x_q:

$$\alpha_q \leftarrow \alpha_q + x_q, \forall q \in Q,$$

where $-b_q \le x_q \le b_q$.

4. Allow the pricing mechanisms time to adjust the demands for the QoS-Transit Services to their new allocations.
5. Repeat.

The heuristic may be terminated if the revenue gain in a round, or over a number of rounds, is below some threshold. For the results in the next section, we execute the heuristic until no better solution is found in 10 consecutive rounds.

Recall that the marginal revenues of the services are not known exactly, but are estimated. Also, the limits on allocation changes per round are to improve the stability of the system. Pricing mechanisms operate to assure that the demand for a QoS-Transit Service does not exceed its allocated bandwidth. The bandwidth allocation to a QoS-Transit Service must always exceed the bandwidth necessary to guarantees QoS given the current level of demand.

Additionally, updating rounds must be staged far enough apart in time in order to ensure that the pricing mechanisms have time to adapt demand to the new bandwidth allocations. Excess bandwidth can be maintained on each link in case excess expected demand occurs as a result of the allocation change, or due to a pricing mechanism not adapting its demand as quickly as expected.

5.1.3 Results

In this section we study the IAA heuristic. The results were compiled from a 30-node network with 54 links, each with a capacity of 100 Mbps, where $b_{max} = 2\,\text{Mbps}$. QoS-Transit Services are randomly created and bandwidth is randomly allocated, though each service is initially allocated a minimum bandwidth of

$$\frac{C}{2 \cdot |Q|}, \tag{5.3}$$

where C denotes the capacity of each network link, and $|Q|$ denotes the number of offered QoS-Transit Services.

While the demand functions are not known by IAA, they still exist, dictating how the customers react to changes in price. The underlying demand functions are created as they were in Section 4.1.6.

Figure 5.1 shows how quickly IAA approaches the optimal solution for 50 QoS-Transit Services, with $\sigma = 4\%$. The optimal solution is approximated by solving the problem with IterLP, from the previous chapter, using the underlying demand functions. 97.7% optimality was reached in 90% of test cases, as shown in Figure 5.2, and is reached after an average of 84 rounds. 90% optimality was reached by all test cases after an average of 37 rounds, and 80% optimality was reached after only 13 rounds on average. These results are as expected, with earlier gains being achieved with less work than the subsequent gains.

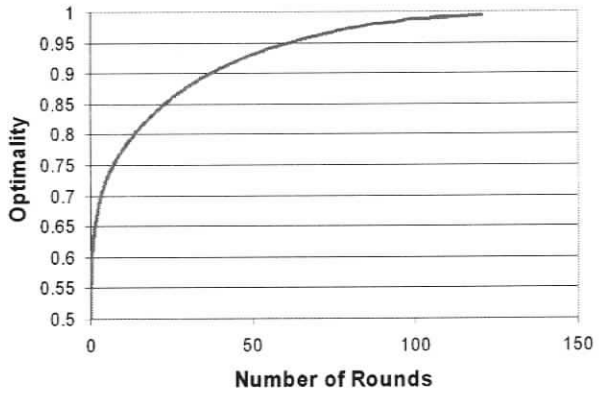


Figure 5.1: Optimality as a function of rounds for 50 services on a 30-node network, for σ =4%.

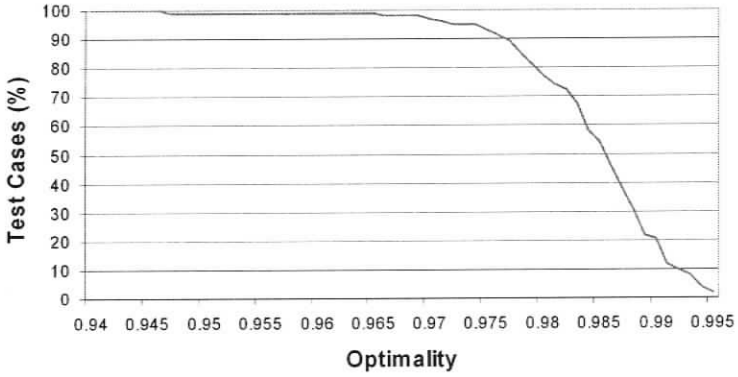


Figure 5.2: Percentage of test cases that achieve a specified optimality.

Figure 5.3 shows the average gain in optimality per additional round depending on the optimality of a given solution. As we can see, the average gain diminishes as the solution approaches optimality. The implication here is that it may be difficult for a system to achieve a near-optimal (say greater than 95% optimal) solution. There may be insufficient time to execute the number of rounds necessary to achieve a desired level of optimality before prices change, due to changes in demand resulting from external factors, thus obviating the solution. The smaller the average gain in optimality per additional round, the less the system has to dynamically change its prices (due to ongoing changes in demand) in order to offset the changes made by the optimal allocation mechanism.

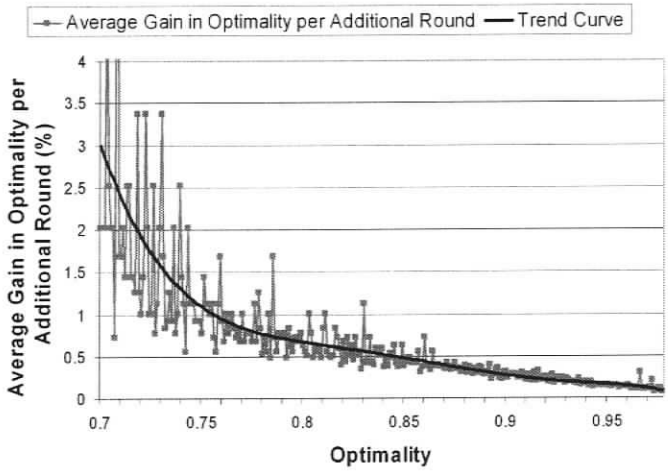


Figure 5.3: Average optimality gain per additional round for 50 services on a 30-node network, for σ =4%.

The time to calculate the allocation change for each round is negligible compared to the time required by the pricing mechanisms to react to these changes. While a round of the IAA heuristic can be performed in a few milliseconds, we estimate that the pricing mechanisms of the QoS-Transit Services should be allowed several seconds to adjust to the change. The actual time given to the pricing mechanisms to alter demand to meet the new allocations will depend on the specific pricing mechanisms implemented. Additionally, the time given to the pricing mechanisms will likely be a function of σ: the more an allocation can change, the more time a pricing mechanism needs to adjust the demand to the new allocation. Thus, if each round occurs every 10 seconds, then 90% optimality will be reached in an average of 370 seconds. This is not exceptionally fast; however, we test from a random state, and the optimal allocation mechanism can be executed often to ensure the allocations do not stray far from optimal.

While the previous results assumed 50 offered QoS-Transit Services, we now study how the number of services offered by the system affects the number of rounds required to achieve a specified level of optimality. Figure 5.4 shows the number of rounds required to achieve 90% optimality for a varying number of QoS-Transit Services. As expected, the number of rounds required to reach a specified level of optimality increases with the number of QoS-Transit Services being offered. There are two reasons for these results. First, the fewer QoS-Transit Services offered, the less likely it is that they will demand

the same resources, causing contention. The less contention there is, the closer to optimal their initial random allocation of bandwidth will be. Second, the more QoS-Transit Services offered, the less bandwidth each will be randomly assigned in our tests and the longer it will take for the higher priced services to be allocated a substantial proportion of the bandwidth.

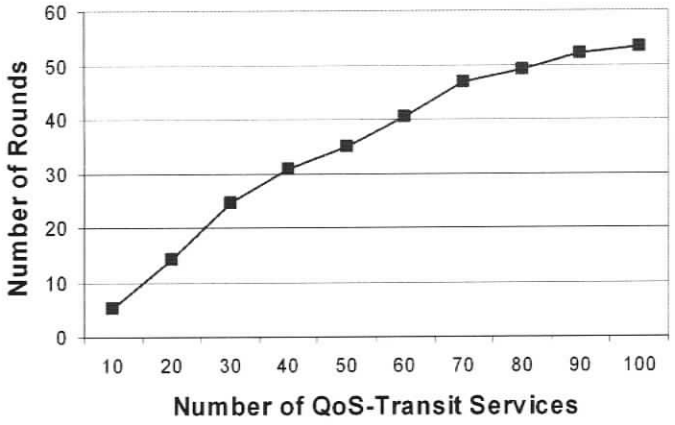


Figure 5.4: Average number of rounds to achieve 90% optimality with σ =4%, on a 30-node network.

Figure 5.5 shows the average number of rounds to achieve 90% optimality as σ is varied. Not surprisingly, as σ increases, the solution converges towards optimality more quickly. This is simply because the solution can change more in each round. However, the larger σ is, the more time a pricing mechanism may require to control demand to meet the allocation. Also, as σ increases, so does the probability of violating QoS guarantees due to a large, sudden reallocation of bandwidth from a QoS-Transit Service.

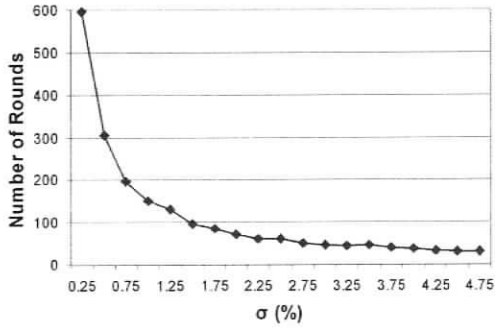


Figure 5.5: Average number of rounds to achieve 90% optimality for 50 QoS-Transit Services on a 30-node network for varying σ.

5.2 Optimal Routing

While the QoS-Transit Services of a network may be optimally (or near optimally) allocated on a set of predefined paths, the paths on which they are routed across may not be optimal. In this section we identify how a network engineer can determine which links in the network are not carrying traffic with a high value. The engineer can then reroute certain QoS-Transit Services, increasing the overall value of the traffic transiting the network, and thus increasing the overall revenue for the QoS-ISP. We assume that the flow of a QoS-Transit Service is not split among multiple paths across the network.

Given the set of QoS-Transit Services, Q, which are offered by a QoS-ISP, we can determine the value of the traffic that currently crosses each link within the QoS-ISP. Let the price, $p(q)$, for QoS-Transit Service q be normalized and reflect the price per unit of effective bandwidth. We can subsequently define a price to use a link to be $p(\ell)$, $\forall \ell \in L$, where L is the set of links within the network. As we will show, $p(q) = \sum_{\ell \in Lq} p(\ell)$ when the network bandwidth is optimally allocated, where L_q is the set of links that define the path that QoS-Transit Service q is routed on.

Consider two links (ℓ_1 and ℓ_2) of a QoS-ISP carrying three QoS-Transit Services (q_1, q_2, and q_3), as shown in Figure 5.6. $p(q_3) = p(q_1) + p(q_2)$ because we assumed that the QoS-Transit Services were optimally allocated, and therefore the services are each used to their capacity. If $p(q_3) \neq p(q_1) + p(q_2)$, then an optimal allocation cannot exist because it would be more profitable to increase either the allocation of q_3 or the allocations of q_1 and q_2. Here, since q_1 and q_2 each traverse, and only traverse, links 1 and 2, respectively, we know that $p(\ell_1) = p(q_1)$ and $p(\ell_2) = p(q_2)$.

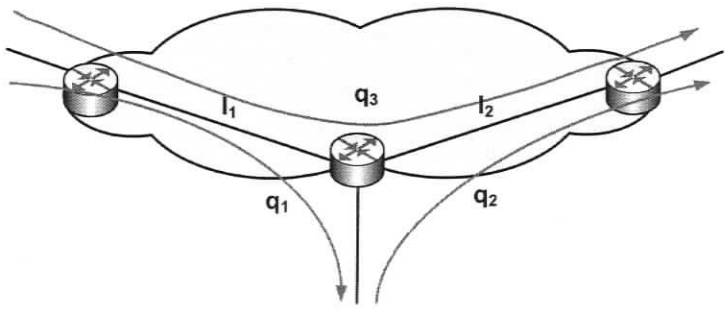


Figure 5.6: A network with 2 links carrying 3 QoS-Transit Services.

In the general case, we can define the price of each link in a network by solving the following set of linear equations:

$$\sum_{\ell \in L_q} p(\ell) = p(q), \forall q \in Q. \tag{5.4}$$

We make two provisos in order to make the set of linear equations solvable under most conditions. First, if a set of links are concatenated and they carry the same QoS-Transit Services, then those links should be considered as one for the purpose of finding the price of the links. This proviso is illustrated in Figure 5.7.

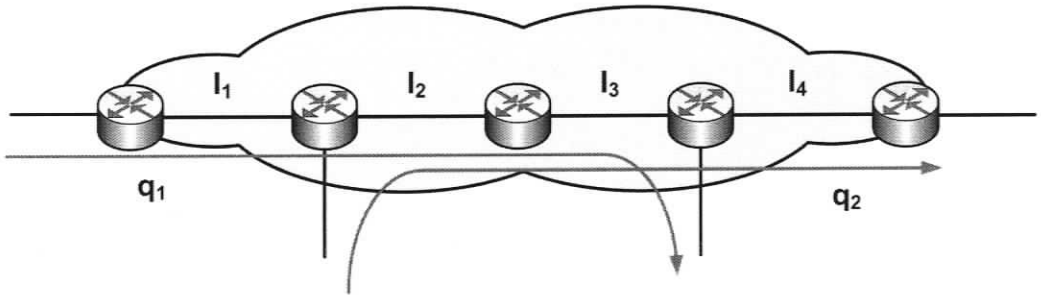


Figure 5.7: Links 2 and 3 here should be considered as a single link when finding the values of a network's links.

The second proviso is the ignoring of links (not services) not used to capacity, assuming that only positive marginal revenues can occur and that the network is optimally allocated. If remaining bandwidth exists on a link, which is not assigned to a QoS-Transit Service, then that link does not constrain the bandwidth the ISP is selling. For an example, consider the 5-node network with 4 QoS-Transit Services in Figure 5.8.

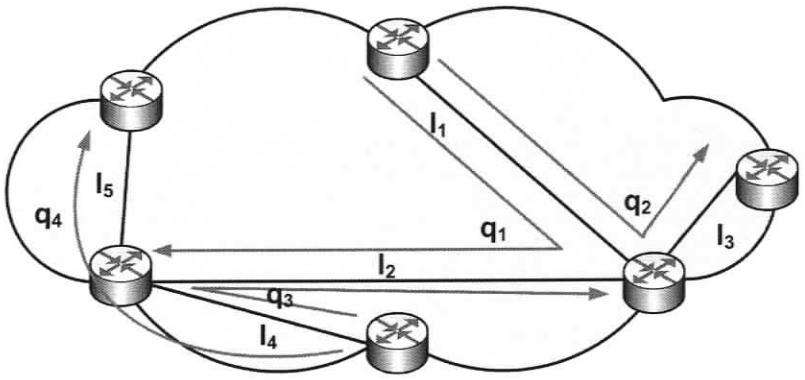


Figure 5.8: 5-link network with 3 QoS-Transit Services.

The set of linear equations resulting from this scenario is:

$$\begin{aligned} p(\ell_1)+p(\ell_2) &= p(q_1) \\ p(\ell_1)+p(\ell_3) &= p(q_2) \\ p(\ell_2)+p(\ell_4) &= p(q_3) \\ p(\ell_4)+p(\ell_5) &= p(q_4), \end{aligned} \tag{5.5}$$

where $p(q_1)$, $p(q_2)$, $p(q_3)$, and $p(q_4)$ are known. If all the links in the network have the same bandwidth, and the bandwidth on links ℓ_3 and ℓ_5 is not fully consumed, then by applying the second proviso, we set $p(\ell_3)$ and $p(\ell_5)$ to 0. We then have:

$$\begin{aligned} p(\ell_1)+p(\ell_2) &= p(q_1) \\ p(\ell_1) &= p(q_2) \\ p(\ell_2)+p(\ell_4) &= p(q_3) \\ p(\ell_4) &= p(q_4), \end{aligned} \tag{5.6}$$

and thus,

$$\begin{aligned} p(\ell_1) &= p(q_2) \\ p(\ell_2) &= p(q_1)-p(q_2) \\ p(\ell_4) &= p(q_4). \end{aligned} \tag{5.7}$$

However, if the bandwidth of links ℓ_3 and ℓ_5 is fully consumed at the optimal allocation, together with all the other links, then Equation 5.4 is unsolvable for this network. In these hopefully rare situations, we are unable to determine the value of all the links.

Using the price information for the links, a traffic engineer can decide how to best reroute traffic through the network. There is too little information to determine an optimal strategy, unless the network provider can reasonably estimate the demand functions. If so, the problem is a modified resource constrained optimization problem, as described and solved in the previous chapter.

5.3 Summary and Conclusions

If an ISP offers dynamically priced network services, such as QoS-Transit Services, to its customers, then it has an opportunity to maximize its profit by optimally allocating and routing those services. For the work in this chapter, the ISP is not assumed to know, or be able to estimate, the demand functions for its offered services. We first proposed the IAA heuristic to find the optimal allocation of bandwidth for the offered services of an ISP, so that the ISP's revenue is maximized.

Using the IAA heuristic, we estimated that 90% optimal revenue, from a random bandwidth allocation state, can be achieved in approximately 6 minutes on average, for 50 services on a 30-node network. However, if the heuristic is executed often, without allowing the current allocation to deviate far from the optimal solution, then similar results will be achieved much faster.

To best route each of the services, we proposed that a network engineer determine the values of the links. The link valuations essentially equate to the prices of the links. By using the link valuation scheme, as shown in this dissertation, a network engineer can determine which links are in high demand, and which are not. A network engineer can then route new services along less demanded routes, or perhaps reroute highly valued

services along less demanded routes, in order to increase the overall profitability of the ISP.

Up to this point within this dissertation, we have examined issues primarily related to the offering of QoS-Transit Services within a single network. QoS-Transit Services, however, were defined with the purpose of achieving end-to-end multi-network QoS. The next chapter introduces important concepts for the use of concatenated QoS-Transit Services to achieve QoS across multiple networks. The subsequent chapter then outlines a particular architecture for achieving end-to-end QoS across multiple networks.

6 Multi-Network Quality of Service

In this chapter we study problems related to achieving QoS across multiple networks, potentially from multiple network providers, using QoS-Transit Services. Having defined QoS-Transit Services, we can now consider how to allow data to be carried across multiple networks in order to provide end-to-end QoS. We discuss issues related to charging, interconnection, estimating true expected performance, and delivering QoS-Transit Services across overlay networks.

Eurescom specifies two basic interconnection models: cascade and hub [86]. In the cascade model, an ISP offers a set of services for the end-to-end delivery of its customer's traffic. A customer deals only with a single ISP. An ISP purchases the use of services from its neighbouring ISPs in order to deliver its customer's traffic, as shown in Figure 6.1. Each ISP can offer end-to-end services, but the customer is limited to only those end-to-end services offered by her ISP.

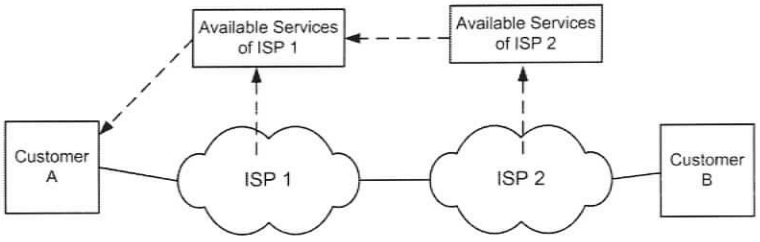


Figure 6.1: The cascade model: ISPs advertise services that may, in turn, use services of their neighbouring ISPs. Customers deal with a single ISP.

In the hub model, as shown in Figure 6.2, the services of all the ISPs are known to the customer. A customer can determine which services, from which ISPs, to use to route his data.

We begin by examining two charging models based on the cascade and hub interconnection models described above. We then examine the special cases of delivering QoS on links interconnecting two QoS-ISPs. Since QoS guarantees define lower performance bounds, we show how expected performance can be estimated for streams traversing multiple QoS-Transit Services. Further, since we cannot expect all

ISPs to offer QoS-Transit Services, the concept of *virtual network operators* offering QoS-Transit Services is introduced. Using this concept, QoS can be delivered across large geographical areas without the cooperation of existing network owners.

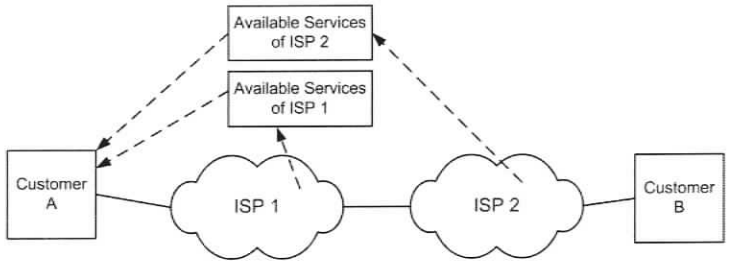


Figure 6.2: The hub model: customers can view the services of all the networks, thereby allowing a customer to choose the path for his traffic.

6.1 Hub Charging Model

Our Hub Charging Model (HCM) consists of *Pricing Gateways* and *Brokers*, as shown in Figure 6.3. *Pricing Gateways* (PGs) advertise the prices of QoS-Transit Services of one or more different networks. *Brokers* act as intermediary charging entities and determine the best routes for their customers' traffic. (Recall that we defined *QoS-ISPs* as the ISPs that provide QoS-Transit Services.)

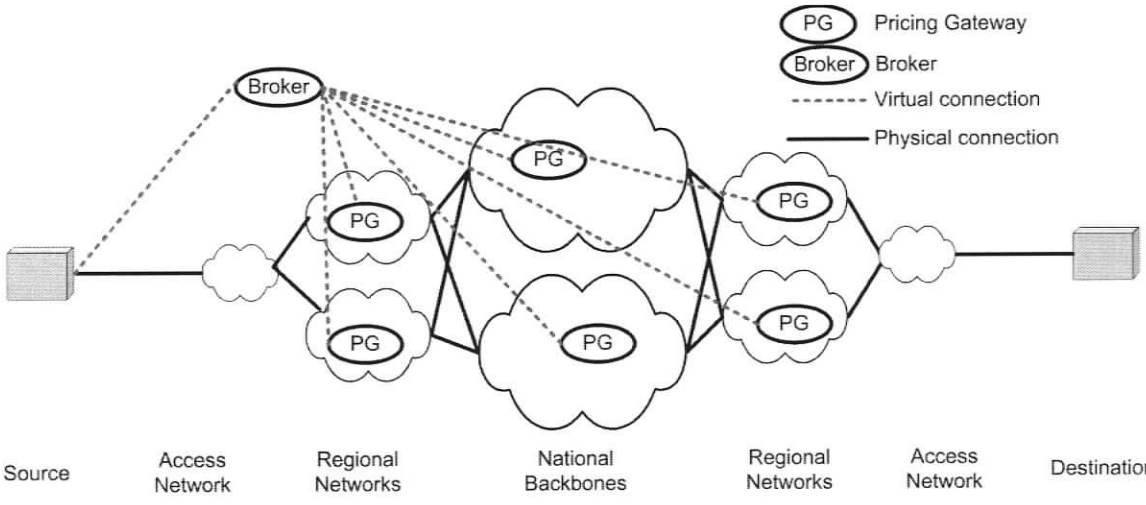


Figure 6.3: The architecture of the HCM using network dependent pricing gateways.

A connection between two customers is built from a set of links from numerous networks, and at various levels: WAN, MAN, or LAN. We assume that networks can

have their own pricing and traffic engineering policies, as long as they provide their advertised services, with associated guarantees, to incoming traffic. This black-box approach allows network providers to choose their own competitive advantages, in order to compete effectively. We therefore treat a single QoS-Transit Service as a virtual link, as the actual route across a single network is unknown. A single end-to-end connection is then made up of many virtual links, each of which may consist of numerous physical links through a single administered network.

For clarity, the interaction between the components of the HCM is shown in Figure 6.4.

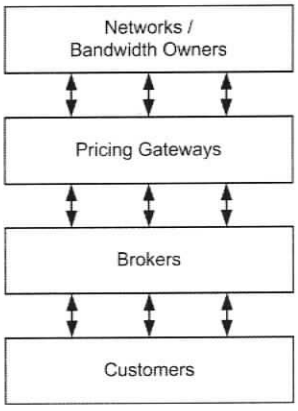


Figure 6.4: The interaction between components of the HCM.

6.1.1 Pricing Gateway

The pricing gateway (PG) contains up-to-date pricing information about network services. Pricing has long been considered as a strategic marketing tool [6(p.21),63], a main source of differentiation among producers. Pricing thus should be determined independently by each network and is based on the demand for the QoS-Transit Services in question. The spirit of this differentiated pricing must be maintained in our model. It needs to be emphasized that end-to-end paths traversing more than one network are not advertised at the PGs in our model. Instead it is the individual QoS-Transit Services that are advertised. Paths requiring the traversal of more than one network require the concatenation of two or more QoS-Transit Services.

While each network could use its own independent PG, it is advantageous to use shared PGs, where prices for multiple networks are available, as shown in Figure 6.5. Actually, the data could be replicated at several locations (i.e. one for each continent, country, or state/province). This centralization simplifies the work of the broker and should prove more scalable than independent PGs, as the number of virtual connections required when using independent PGs is

$$number\_of\ brokers \times number\_of\_QoS\text{-}ISPs,$$

whereas when using a shared PG, the number of virtual connections required is

$$number\_of\ brokers + number\_of\_QoS\text{-}ISPs.$$

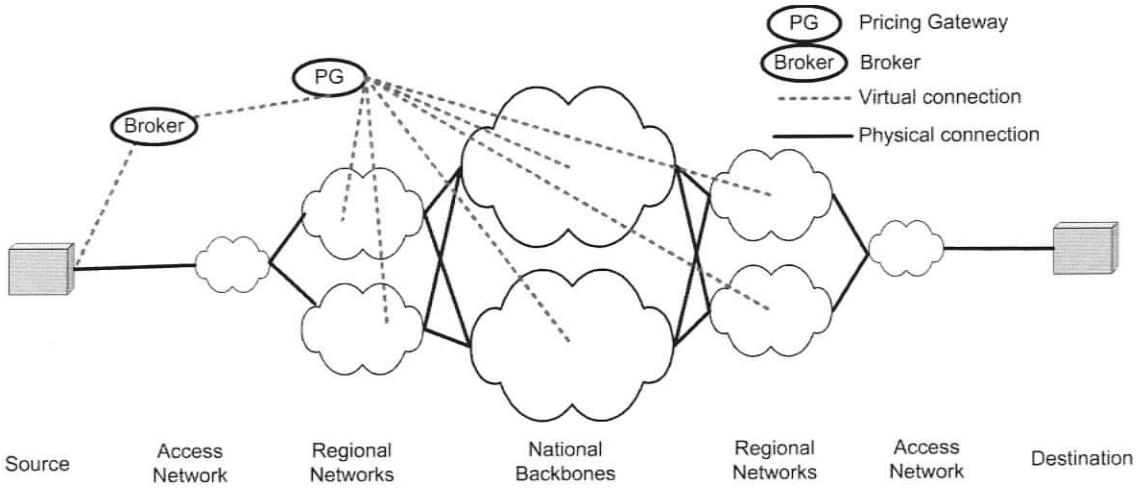


Figure 6.5: The architecture of the HCM using a pricing gateway.

The pricing gateway is host to a *pricing database*, maintaining the pricing information of the QoS-ISPs that it represents, as shown in Table 6.1. The database contains the list of QoS-ISPs, their offered QoS-Transit Services, and the current prices. The *Service Description* includes the source, destination, performance guarantees, and the tariff definition.

Table 6.1: Pricing database.

QoS-ISP	Service ID	Service Description	Current Price
1	1	source, destination, performance guarantees, tariff definition, ...	price
1	2	source, destination, performance guarantees, tariff definition, ...	price
⋮	⋮	⋮	⋮
3	1	source, destination, performance guarantees, tariff definition, ...	price
3	2	source, destination, performance guarantees, tariff definition, ...	price
⋮	⋮	⋮	⋮

6.1.2 Brokers

A broker acts as an intermediary charging entity between customers and ISPs, as well as determines the best routes for its customers' traffic.

We assume that knowledge about the best QoS-Transit Services (paths) through the networks is available, through a *market data provider* [24], or through other means. The broker could be one such market data provider.

Brokers provide many services to help a customer connect to the QoS-ISPs. Each network may employ a different charging scheme, and brokers can hide the complexity and diversity of charging schemes from customers. Hence, brokers act as a financial intermediary between networks and customers. A customer pays a broker, who in turn pays the QoS-ISPs. This relieves customers from having to pay each network provider for service, and allows brokers and QoS-ISPs to engage in more interesting, and trusting, relationships.

A customer also uses a broker to find the *best path* based on its QoS criteria. The *best path* is the cheapest path that meets the customer's performance criteria. If there exists another, cheaper, path through the network, it should be found as an arbitrage opportunity, or as an opportunity to sell the discovery of the route to interested parties. The brokers that find the cheapest paths have a competitive advantage over competing

brokers. Thus, in a competitive environment, there is pressure to find near optimal paths. Such a competitive environment would allow multi-homing, where customers can readily choose between brokers, even on a session-by-session basis.

Our framework allows for both receiver-pay and sender-pay pricing models. Receiver-pay pricing is important for offloading the cost of serving data from data sources, though it is often complicated with multicast routing [87]. Sender-pay is important for service bundling. For example, a user may pay $30 to a video-on-demand provider in order to watch a live music concert. The video-on-demand provider is then responsible for paying all network costs to deliver their video to its customers. As Hermalin and Katz remark, it may not always be economically optimal to have one of the parties bear all the costs [88], which is also allowed in our charging framework.

6.1.3 Customers and Users

Recall that we differentiate between customers and users. Customers are the entities that purchase QoS-Transit Services. Users are the individuals or entities that generate traffic. A customer purchases QoS-Transit Services for an aggregate of users. We do not explicitly define the relationship between customers and users, as it is largely beyond the scope of this dissertation. We will, however, give an overview of how these relationships may occur.

A customer may be an ISP that purchases QoS-Transit Services from QoS-ISPs for its own customers. The ISP can use the performance guarantees for certain users, or for certain traffic (e.g. multimedia traffic). The agreements between the ISP and its clients (the users) may be static or dynamic in nature.

Similarly, consider an application layer multicast service provider that sells the use of multicast trees for the delivery of multimedia traffic. The multicast service provider may purchase QoS-Transit Services to guarantee the delivery of traffic across the multicast tree. The multicast service provider can then engage in a multitude of possible arrangements with its client (the users), splitting costs among the users, charging only the

provider of a stream, or charging only the receiver of a stream. Additionally, these charges may be dynamic in nature, reflecting the dynamic costs of the QoS-Transit Services, or they may be static. If the multicast service provider charges static prices to its clients, then it must decide when the dynamic prices of the QoS-Transit Services are too high to continue serving its clients.

6.2 Interconnection Agreement Charging Model

The Interconnection Agreement Charging Model (IACM) considers the cascade model of interconnection currently used by ISPs. In this model, a QoS-ISP advertises guaranteed services that may terminate outside its network. A QoS-ISP reads the pricing tables of its neighbouring QoS-ISPs and advertises the product of its neighbours' advertised services and its own QoS-Transit Services. Figure 6.6 represents how services are advertised from the view of a single QoS-ISP. A QoS-ISP can price services however it wants, but must realize that it is also a customer of any QoS-ISPs that it sends traffic to.

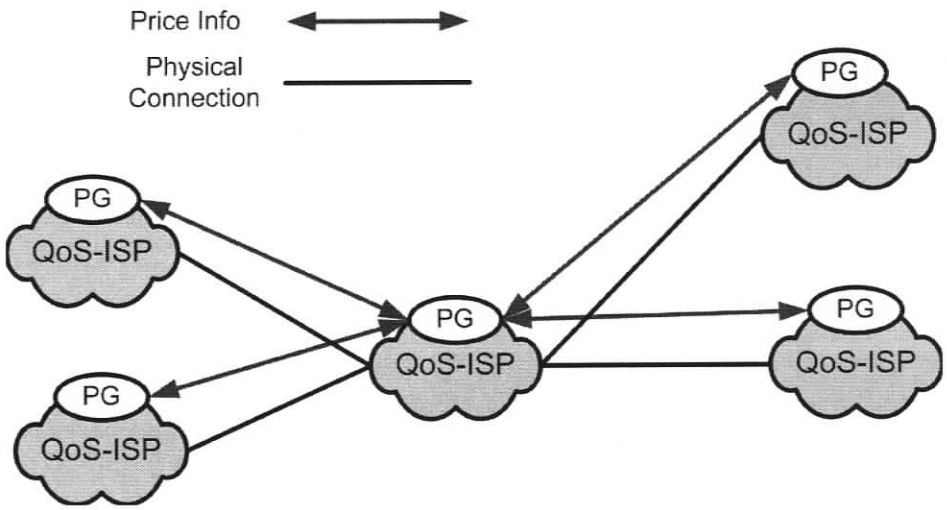


Figure 6.6: View of data flow among QoS-ISPs. Charging and offered services from one QoS-ISP is determined from the charging and service offerings from neighbouring QoS-ISPs.

Figure 6.7 demonstrates the information flow through multiple networks. The source starts by sending traffic to ISP_1 and is charged by a rate determined by ISP_1. ISP_1 then forwards the traffic to either ISP_2 or ISP_3, depending on which one will deliver the traffic to the destination at the cheapest rate, while still respecting any performance guarantees.

Payments are passed down the transmission chain. For example, in Figure 6.7, the customer pays ISP_1, ISP_1 pays ISP_2, and ISP_2 pays ISP_4. Each ISP is unaware of the entire topology of the Internet; it is only aware of its neighbouring peers, and the services that those peers provide. Consider the propagation of service information from ISP_4 to ISP_1; ISP_4 advertises its services to $ISPs_2$ and ISP_3. ISP_2 (and ISP_3) in turn advertises services that originate in its network and terminate in either its network, or the network of ISP_4.

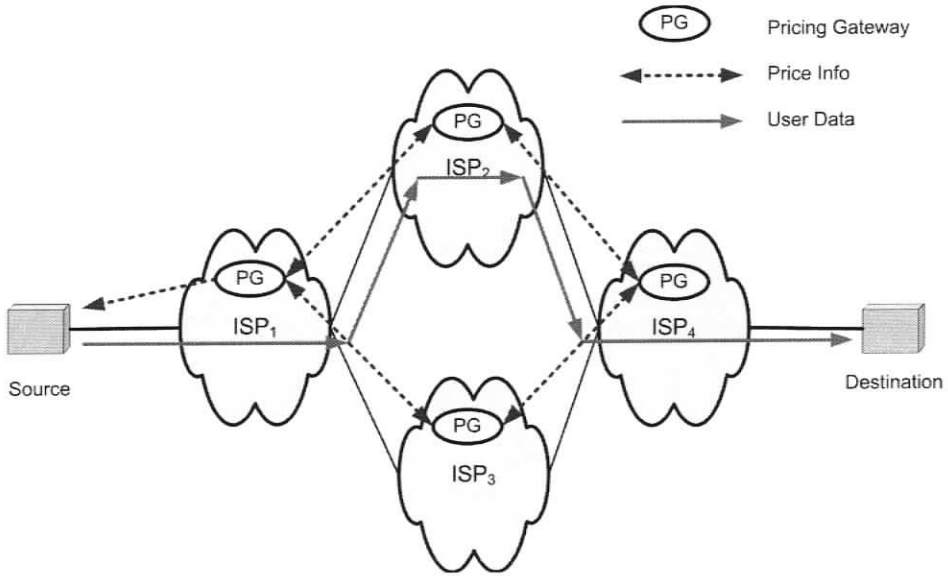


Figure 6.7: The ISPs report pricing information for paths that originate in their networks.

The main disadvantages of the IACM versus the HCM is that pricing information may take longer to propagate to the customer in the IACM, and each QoS-ISP must store pricing information to every destination. As a result, QoS may be harder to guarantee against short, rapid increases in demand, because a single QoS-ISP is at the mercy of all other QoS-ISPs to guarantee its QoS. Additionally, the routing tables could get very large.

The services advertised at each QoS-ISP must be such that any destination is reachable. Therefore the number of services is of the order of the number of destinations. While the aggregation of destinations is possible, such as with CIDR (classless inter-domain

routing) notation [89] (e.g. 142.104.0.0/16 or 142.104.0.0/255.255.0.0), it hinders reacting to destination-specific congestion.

Due to the potential size of the routing tables, we do not expect the IACM to be used end-to-end, but possibly as a means for a small number of QoS-ISPs to cooperate in providing this form of *edge pricing* among themselves. *Edge pricing* [63] is a pricing scheme where charges are calculated locally, at the edge of a provider's network, regardless of the number of networks a connection may be routed.

Consequently, due to the disadvantages of the IACM, in the rest of this dissertation we focus on the HCM, to the exclusion of the IACM.

6.3 QoS between Interconnected QoS-ISPs

So far, we have ignored achieving QoS on links interconnecting two QoS-ISPs. Such a link may be a direct peering link or an exchange point peering link. We note that an IX can operate as a QoS-ISP, just as a regular ISP can. QoS-Transit Services provide guarantees between the sources and destinations of a network, and an IX is a network.

Previously, we assumed that the source and destination of a QoS-Transit Service existed within a single QoS-ISP. However, there are two special cases that result when considering the interconnection of QoS-ISPs. First, the destination of a QoS-Transit Service may be the ingress node of an adjacent network, and second, the egress node of a network may also be the source of a QoS-Transit Service for an adjacent network.

The link interconnecting two networks may be owned by one of the network owners, or the costs may be shared between them. Similarly, control and revenue of routing QoS-Transit Services may be split between the owners, or retained by just one of the owners.

Figure 6.8 illustrates an example interconnection of two QoS-ISPs to an IX, which is represented as a third QoS-ISP. Consider the link between routers A and B in Figure 6.8, and assume QoS-ISP$_1$ owns and controls the associated link between the two routers. So

far, we have considered that QoS-ISPs offer QoS-Transit Services across their own networks. In this example, router B is outside QoS-ISP$_1$'s network. For a QoS-Transit Service from QoS-ISP$_1$ with a destination of router B, QoS-ISP$_1$ can manage the service as usual. This is because the routing of packets along the interconnecting link, to router B, remains in QoS-ISP$_1$'s direct control, and pricing determination can be performed by routers within its domain as well.

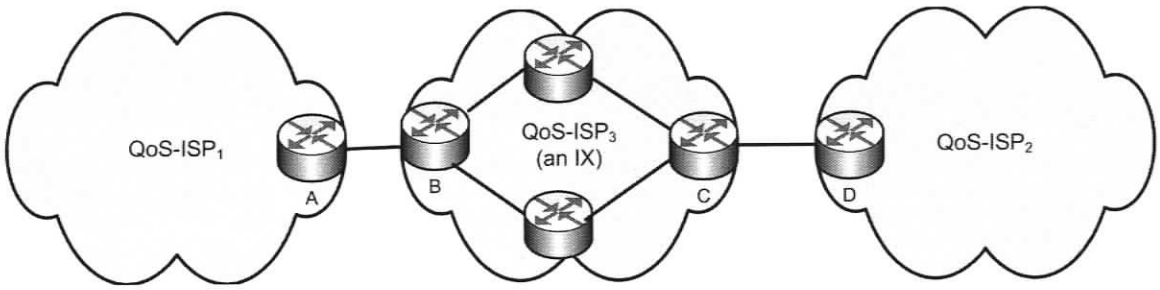


Figure 6.8: Interconnection between 3 QoS-ISPs.

However, if QoS-ISP$_1$ wishes to advertise a service beginning at router B, and terminating within QoS-ISP$_1$'s network, then the ISP must use the destination node of the service to monitor traffic and update the pricing information for the link. This requires only minor changes to the algorithms described in this dissertation, but is worth noting. Additionally, QoS-ISP$_1$ and QoS-ISP$_3$ may need to negotiate routing information, as is discussed in Section 7.2. That is, QoS-ISP$_3$ may need information from QoS-ISP$_1$ in order to properly mark or transmit packets from router B to router A.

6.4 Estimating Performance across Multiple QoS-Transit Services

The performance guarantees resulting from routing traffic across multiple QoS-Transit Services in sequence are a function of the individual QoS-Transit Service guarantees.

Consider a flow, z, that uses a set of QoS-Transit Services, Q_z. The maximum packet loss guaranteed to flow z is:

$$1-\zeta_z = \prod_{q \in Q_z} (1-\zeta_q), \tag{6.1}$$

where ζ_q and ζ_z are the maximum packet losses of QoS-Transit Service q and flow z, respectively.

The guaranteed maximum delay is:

$$l_z = \sum_{q \in Q_z} l_q, \tag{6.2}$$

where l_q and l_z are the latencies of QoS-Transit Service q and flow z, respectively.

While these equations define the guarantees provided by the QoS-Transit Services, a service may often deliver better-than-guaranteed performance. If the expected performance of a QoS-Transit Service is advertised by a broker, in addition to the guaranteed performance, then a more realistic estimate of the expected end-to-end performance can be calculated.

Let us assume that performance statistics are kept for each QoS-Transit Service. Specifically, assume maximum packet loss, average delay, and maximum delay, are determined over defined periods of time (e.g. 200 ms). We can then determine the probability distributions for these performance statistics to determine the expected performance for the flows that use multiple QoS-Transit Services.

Let β be the length of the probability intervals. [For delay statistics, β may be 1ms, while for packet loss, β may be 0.01 (i.e. 1%).] We can now construct a cumulative distribution function for each of the probability distributions.

Let $lossprob(q,\tau_q)$ be the probability that packet loss on service q is within the probability interval $(\tau_q - \beta, \tau_q]$, where $0 < \beta < 1$ and q is the q^{th} QoS-Transit Service that flow z is routed over. Additionally, let $S = \{0, \beta, 2\beta, ...1.0\}$ be the set of values representing the upper bounds for each of the probability intervals. The probability that the packet loss experienced by flow z will be less than ζ_z is defined by the following cumulative distribution function:

$$P(X \le \zeta_z) = \sum_{\tau_1 \in S} \sum_{\tau_2 \in S} \cdots \sum_{\tau_n \in S} \left[\prod_{q=1}^{n} (1 - lossprob(q, \tau_q)) \right], \tag{6.3a}$$

$$\forall \tau_1, \tau_2, ...\tau_n \text{ such that } \prod_{q=1}^{n} (1 - \tau_q) \le 1 - \zeta_z, \tag{6.3b}$$

where the random variable X is the observed packet loss and n is the number of QoS-Transit Services that flow z is routed over.

The cumulative distribution function for the delay statistics is computed in a similar manner. Let $S = \{\beta, 2\beta, 3\beta ...\}$ be the set of values representing the upper bounds for each of the probability intervals, where $\beta \ge 0$. The probability that the maximum packet loss will be less than l_z is defined as follows:

$$P(X \le l_z) = \sum_{\tau_1 \in S} \sum_{\tau_2 \in S} \cdots \sum_{\tau_n \in S} \left[\prod_{q=1}^{n} delayprob(q, \tau_q) \right], \tag{6.4a}$$

$$\forall \tau_1, \tau_2, ...\tau_n \text{ such that } \sum_{q=1}^{n} \tau_q \le l_z, \tag{6.4b}$$

where the random variable X is the observed delay and $delayprob(q, \tau_q)$ is the probability that the maximum (or average) packet delay on QoS-Transit Service q is within the interval $(\tau_q - \beta, \tau_q]$.

We demonstrate the above with an example. Consider a flow z that traverses three QoS-Transit Services. The maximum delays of the services are measured in specified intervals (e.g. 200 ms). Consider the example probability distributions of the maximum delays of the three services, as shown in Figure 6.9, as reported by a broker. The measured maximum delays are grouped within 5 ms intervals (i.e. $\beta = 5$ ms).

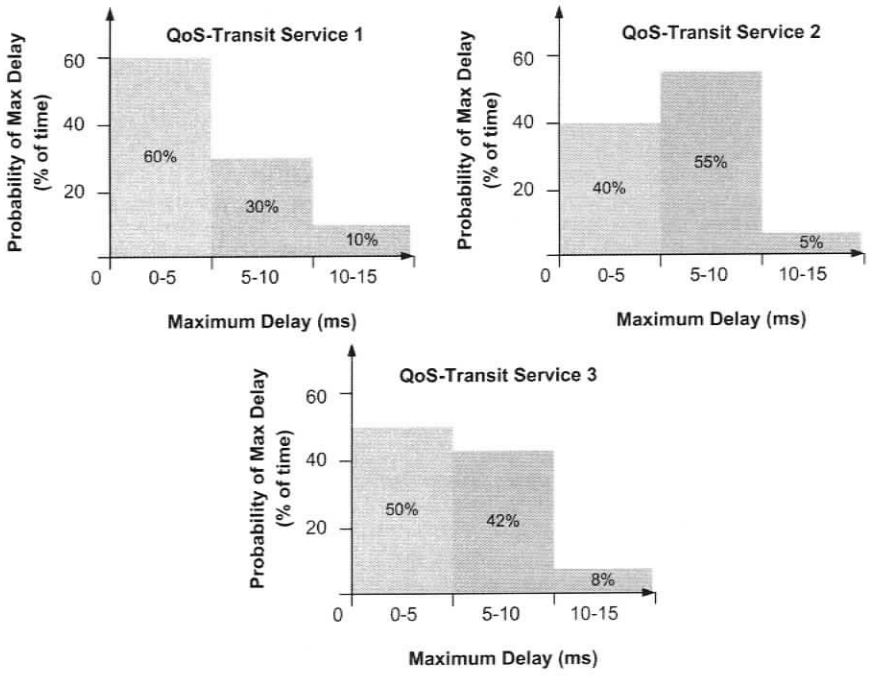


Figure 6.9: Example probability distributions for three QoS-Transit Services.

To calculate the probability that the maximum delay in a single time interval is less than or equal to 20 ms, we calculate the following:

$$P(X \le 20) = \sum_{\tau_1 \in S} \sum_{\tau_2 \in S} \sum_{\tau_3 \in S} \left[\prod_{q=1}^{3} delayprob(q, \tau_q) \right], \tag{6.5a}$$

$$\forall \tau_1, \tau_2, \tau_3 \text{ such that } \sum_{q=1}^{3} \tau_q \le 20, \tag{6.5b}$$

where $S = \{5,10,15,20\}$.

Additionally, by expanding the above equation:

$$P(X \le 20) = \begin{cases} [\text{delayprob}(q_1,5) \cdot \text{delayprob}(q_2,5) \cdot \text{delayprob}(q_3,5)] + \\ [\text{delayprob}(q_1,5) \cdot \text{delayprob}(q_2,5) \cdot \text{delayprob}(q_3,10)] + \\ [\text{delayprob}(q_1,5) \cdot \text{delayprob}(q_2,10) \cdot \text{delayprob}(q_3,5)] + \\ [\text{delayprob}(q_1,10) \cdot \text{delayprob}(q_2,5) \cdot \text{delayprob}(q_3,5)] \end{cases}, \quad (6.6)$$

$$P(X \le 20) = \begin{cases} [0.60 \cdot 0.40 \cdot 0.50] + [0.60 \cdot 0.40 \cdot 0.42] + \\ [0.60 \cdot 0.55 \cdot 0.50] + [0.30 \cdot 0.40 \cdot 0.50] \end{cases}, \quad (6.7)$$

$$P(X \le 20) = 0.12 + 0.1008 + 0.165 + 0.06 = 0.4458 = 44.58\% . \quad (6.8)$$

The cumulative distribution function $P(X \le x)$ can be plotted upon calculating for other values of x, as shown in Figure 6.10.

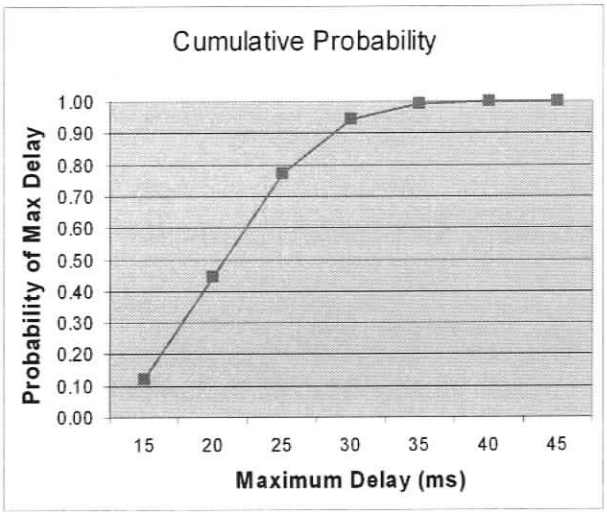


Figure 6.10: Cumulative distribution function of maximum delays for the example flow z using three QoS-Transit Services.

As the example demonstrates, the usual performance received by an application may be much higher than that obtained by simply considering the guarantees of the individual QoS-Transit Services.

6.5 Overlay ISPs

QoS-Transit Services are interesting, but to achieve end-to-end QoS we have implied that they must be universally deployed. We can avoid this implication by considering the use of *virtual network operators* to offer these services, thereby not requiring network owners from all geographic areas to implement and offer QoS-Transit Services. In this section we discuss the possibility of using application layer routing within the Internet to deliver QoS-Transit Services.

We first outline the concept of an *Overlay ISP*, and then discuss how this kind of ISP prices its services based on transit charges from other ISPs. We then discuss how Overlay ISPs can provide guarantees while using the network services offered by other ISPs. Finally we examine the optimal provisioning of an Overlay ISP's links.

6.5.1 Overview

We propose the use of *Overlay ISPs* (O-ISPs), a type of *virtual network operator*, which provides network services without actually owning the links connecting its facilities and routers. An O-ISP operates hosted routers (or servers acting as routers) at strategic locations, within collocation facilities. These routers are used together with leased link bandwidth and/or currently available transit IP services to provide QoS-Transit Services. Figure 6.11 shows how the routers of an O-ISP may be placed between members of a set of ISPs.

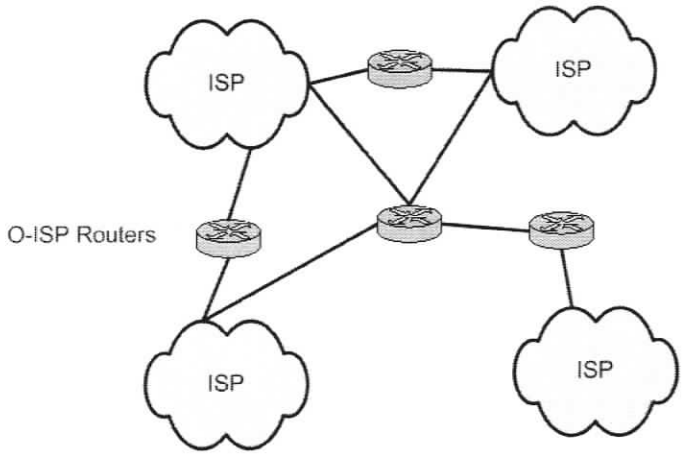


Figure 6.11: An example placement of an O-ISP's routers, at collocation facilities, between members of a set of ISPs.

O-ISPs are interesting for several reasons. First, we can continue to use the basic infrastructure of the Internet to provide end-to-end QoS. Additionally, with O-ISPs we are not at the mercy of ISPs to provide QoS-Transit Services in order to deliver end-to-end QoS.

While current transit agreements provide guarantees across the backbone of a customer's ISP, an O-ISP can provide QoS across a larger geographic area, without large capital outlays, relying on traditional transit agreements from other ISPs for the transport of data between its nodes. Also, as customers are enabled to direct traffic across numerous O-ISPs in sequence, they can further increase the geographical area in which QoS is guaranteed.

We suggest the use of application layer routing within the network to support QoS. The VORAL system [90], for example, uses relay servers within the network to route voice calls through a network. Skype [91] is a peer-to-peer telephony application that uses customers' computers to build an overlay network in order to route calls. The value of such application layer routing is in part due to the fact that Internet routing does not always choose the best route. According to Savage et al., 30 to 80 percent of network routing paths have at least one alternative path that has lower round-trip delay, less packet loss, or more bandwidth [92].

One could build an overlay network using the routers of an O-ISP, as shown in Figure 6.12. We abstract each ISP backbone as a router, with a queuing delay equal to the sum of the queuing and link latencies along the path chosen by the ISP's routing algorithm through its network. (This abstract queuing delay cannot be determined precisely, but it can be estimated by measuring observed delays.) The speed and cost of the links between the abstracted routers will depend on the transit agreements with the attached ISPs. Figure 6.12 indicates the maximum queuing delays for the O-ISP Router (OIR) and the two abstract routers. (Instead of maximum queuing delays, it is equally plausible to indicate 95th percentile queuing delays, where 95 percent of the packets experience a delay of at most the indicated queuing delay.)

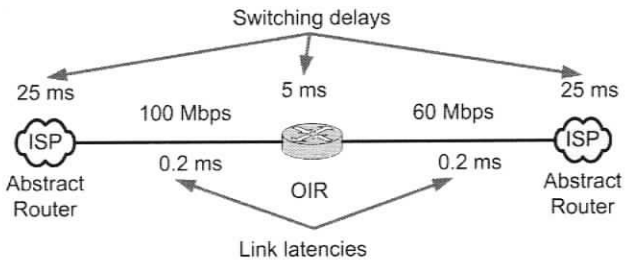


Figure 6.12: Overlay concepts with each ISP's network abstracted to a router with a queuing delay.

Like any ISP's customers, the customers of an O-ISP may include another ISP, a directly attached customer (such as an enterprise), or a hosted customer (such as a content service provider).

The framework as proposed so far has not explicitly addressed bottlenecks within peering links or within IXs [51,52]. The simplest way to avoid these bottlenecks is to place the routers of O-ISPs in suitable locations. For example, IXs based on shared Ethernet may suffer congestion that is beyond the control of the O-ISP. However, IXs based on ATM, with properly provisioned circuits between transit providers and hosted entities, should not have congestion problems, unless caused by the O-ISP itself. Similarly, a small percentage of routes are responsible for the majority of network instability [93]. Avoiding such routes, and their fail-over routes, may help to improve the stability of overlay networks.

An O-ISP is a form of virtual network operator, and thereby suffers their limitations. Since a virtual network operator does not own its network, it has no direct control over failures in its network [94]. However, the O-ISP can subscribe to bandwidth from more than one network, through leasing or transit agreements, to provide fault tolerance.

6.5.2 Determining Guarantees

Since packets sent between O-ISP routers (OIRs) are often carried on the network of another ISP, much of the direct control over performance is lost to the O-ISP. In this section we explore how an O-ISP can determine performance criteria based on its transit agreements.

ISPs' SLAs often state guarantees for latency, packet loss, and jitter; however, these are often only monthly averages [95,96]. These SLAs do not help O-ISPs to provide shorter guarantees for QoS-Transit services. The OIRs are thus required to measure the performance among themselves. We propose that each OIR monitor the connections with its adjacent OIRs. We assume each OIR transmits to its adjacent OIRs using UDP/IP – although TCP could be used to reduce packet loss for a service, with the expectation of higher delay.

We assume that bandwidth itself is not a limiting resource on an ISP's network, as an O-ISP will be bandwidth limited by its connection to the ISP. We thus assume the ISP will not sell more bandwidth to the O-ISP than it can deliver.

One-way performance measurements can be either active or passive. With active measurements, probe packets are periodically injected into the network to determine the performance of the network. With passive measurements, existing data packets within the network are monitored to determine the performance they are enjoying.

The standardization of the One-Way Active Measurement Protocol (OWAMP) [97] is an example of using active measurements. Passive measurements can be performed through the synchronization of clocks between the sender and receiver of a packet. Clocks may

be synchronized through GPS (Global Position System) and/or CDMA (Code Division Multiple Access) signals. Then, through time-stamping packets, delay and jitter can be determined. Current systems exist for passively measuring networks, such as Endace's DAGMON series of network monitoring systems [98].

6.5.3 Provisioning Links

We assume that QoS-ISPs apply uniform, nondiscriminatory pricing, and that bandwidth is optimally allocated to the QoS-Transit Services, as defined in Chapter 4. There, we examined how to allocate link bandwidth to a set of QoS-Transit Services, and now we must additionally consider when to allocate additional bandwidth to a link. We do not discuss deallocation of bandwidth from a link, as we assume bandwidth is provisioned for a fixed period of time, based on an agreement at the time of purchase. Thus, bandwidth deallocation is determined by the contract duration at the time of bandwidth allocation.

Determining how an O-ISP can optimally provision links is difficult because the decision to provision is based on forecasts of future demand and on the prices of QoS-Transit Services, and because the cost (per unit) of transit services or leased lines is often reduced by committing to longer term agreements.

There are two scenarios we need to consider: 1) transmission of packets using multiple connections (and thus agreements), and 2) purchasing (additional) bandwidth for connections. We assume an O-ISP is charged for both incoming and outgoing traffic.

For the transmission of packets using multiple agreements, we need to consider the current revenue generated by the QoS-Transit Services, and the costs of the network. Recall that costs may be dynamic, especially for transit agreements. For example, with volume pricing, once the initial purchased volume has been used, the transmission of additional traffic will cost more, per unit, than the initial allocation. Similarly, with percentile pricing, the choice to transmit at a higher rate may have expensive consequences, especially if subsequent demand for the QoS-Transit Service is reduced.

In this situation, an O-ISP does not need to explicitly purchase additional bandwidth, but the ISP must determine at what rate data should be transmitted across existing connections. Forecasting demand is thus extremely important for O-ISP profitability. This is also seen when determining if additional bandwidth needs to be provisioned.

Additional bandwidth should be provisioned (purchased) when the forecast marginal revenue from the additional bandwidth is greater than the cost of purchasing it. If we consider agreements of varying length, such that the unit price ($/Mbps/month) increases for shorter term contracts, then we can say that bandwidth should be provisioned when

$$\exists \bar{t}, \bar{\ell}, \bar{c} \text{ such that } MR(\bar{t}, \bar{\ell}, \bar{c}) \geq MC(\bar{t}, \bar{\ell}, \bar{c}), \tag{6.9}$$

where $\bar{\ell}$ is a vector of links being upgraded, $\bar{c}$ is a vector of bandwidths for which the links are being upgraded by, $\bar{t}$ is a vector of contract lengths, MR is the forecasted marginal revenue, and MC is the marginal cost. The optimal allocations are thus the set of allocations that maximize the increase in profit, calculated by subtracting the marginal costs from the forecast marginal revenue:

$$\max_{\bar{t}, \bar{\ell}, \bar{c}} \left[MR(\bar{t}, \bar{\ell}, \bar{c}) - MC(\bar{t}, \bar{\ell}, \bar{c}) \right]. \tag{6.10}$$

6.6 Summary and Conclusions

In this chapter we have expanded on the concept of QoS-Transit Services for guaranteed performance across a single domain to include the concept of end-to-end QoS using multiple QoS-Transit Services.

We first presented two charging models: the HCM, based on a hub model, and the IACM, based on a cascade model. The IACM provides a simple pricing interface for attached networks; however, pricing signals may take a long time to propagate through the networks, and there are scalability issues as a QoS-ISP may need to store pricing information for every destination. The IACM thus may be best to administer a small

number of cooperating QoS-ISPs. Conversely, the HCM model not only provides a structure for the development of higher-level services, but is also conducive to competitive and complex relationships. By combining central pricing repositories (pricing gateways) with the concept of brokers, pricing information can be rapidly disseminated, and the complexities of the networks are hidden from customers.

While QoS-ISPs provide guarantees across their networks, we also addressed how the links interconnecting two QoS-ISPs are managed to assure end-to-end guarantees. This was also important in describing the role of IXs as QoS-ISPs.

Although QoS-Transit Services provide statistical guarantees of performance, the actual performance received from a service may be much better than the bounds provided by the guarantees. For a flow routed along multiple QoS-Transit Services, the performance expected can be estimated, based on past measurements of the individual services. Such estimation allows customers to be better informed as to which set of QoS-Transit Services will allow them to achieve their performance goals.

As it is unreasonable to expect a large number of ISPs to suddenly offer QoS-Transit Services, we developed the idea of an Overlay-ISP: a virtual network operator that operates as a QoS-ISP without owning the underlying network. Using this approach, the Overlay-ISP can provide QoS-Transit Services across large geographic areas, and does not require cooperation from the underlying ISPs. The ability of an Overlay-ISP to maximize its revenue depends on its ability to forecast demand. For example, transmitting additional traffic across a network that charges based on 95^{th} percentile pricing can be an expensive decision if future demand drops.

7 A Possible Architecture

In this chapter we propose a possible architecture for deploying QoS-Transit Services using existing networking protocols. We first give an overview of the architecture, including how the HCM of the previous chapter works within our proposed architecture. Second, we illustrate how customers can route their traffic across a predetermined set of QoS-Transit Services. We then show how a billing party can be designated within IPv4 and IPv6 packets, and how customer traffic flows can be metered and policed. Finally, we briefly address potential security problems and outline potential solutions for them.

7.1 Overview

We propose an architecture for deploying QoS-Transit Services based on the HCM (Hub Charging Model) of Section 6.1, using existing networking protocols. Specialized protocols may be devised to overcome certain disadvantages of our proposal.

A customer is connected to a *Broker-ISP* (B-ISP) in order to gain access to the QoS-ISPs, as shown in Figure 7.1. Recall that customers may not be end users, but rather ISPs or enterprises that aggregate user traffic. A B-ISP performs the functionality of the broker in the HCM, finding the best QoS-Transit Services on which to route its customers' traffic, as well as acting as a financial intermediary between its customers and the QoS-ISPs that they use. Besides its role as a broker, the B-ISP aids significantly in routing, and may additionally aid in metering and security, as is explained below.

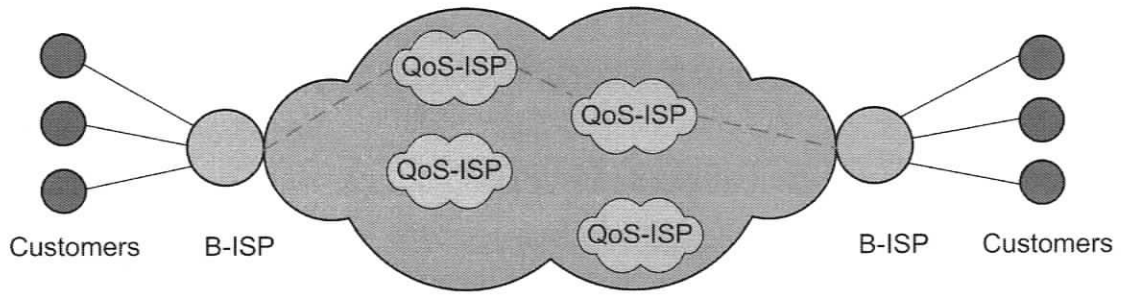


Figure 7.1: The proposed architecture.

The destination is not required to be attached to a B-ISP, as the final QoS-ISP in a path can simply forward the traffic towards the destination, using traditional IP routing.

We do not address QoS between customers and their B-ISP, or between customers and users, as this is technology dependent. The customer is responsible for controlling QoS in his own network.

7.2 Routing Framework

While various protocols may be developed for routing traffic between the QoS-ISPs, to ease adoption we suggest the use of MPLS. As previously mentioned, QoS-ISPs advertise their offered QoS-Transit Services at a PG (Pricing Gateway). Along with each QoS-Transit Service, a QoS-ISP advertises an associated MPLS label, perhaps in place of the Service ID in the pricing database, as shown in Table 7.1. All traffic routed across a QoS-ISP using one of its advertised labels enjoys the stated advertised performance associated with that label. An MPLS label thus represents a QoS-Transit Service within this framework.

Table 7.1: Pricing database with MPLS labels to indicate the QoS-Transit Service.

QoS-ISP	Service ID / MPLS Label	Service Description	Current Price
1	1	source, destination, performance guarantees, ...	price
1	2	source, destination, performance guarantees, ...	price
⋮	⋮	⋮	⋮
3	1	source, destination, performance guarantees, ...	price
3	2	source, destination, performance guarantees, ...	price
⋮	⋮	⋮	⋮

The MPLS header [99] is shown in Figure 7.2. *Label* is a 20-bit field identifying the MPLS label, *Exp* is a 3-bit field reserved for experimental use, *S* is a 1-bit flag indicating the bottom of the stack, and *TTL* is an 8-bit field indicating the time-to-live. Either the *Exp* or the *Label* field can be used to determine the scheduling behaviour of the packets.

(DiffServ over MPLS can use the *Exp* field for holding the DiffServ CodePoint, as explained in [100].)

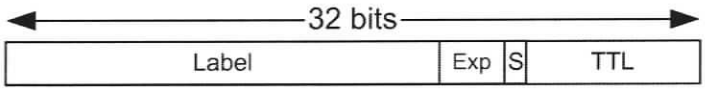


Figure 7.2: MPLS header.

We now describe how to use MPLS to route traffic across multiple networks. This is simple because MPLS allows label stacking, whereby a series of MPLS headers within a single packet can define a concatenated series of label-switched paths (LSPs) that the packet will navigate.

For the use of a single QoS-Transit Service, one or more MPLS labels may be stacked, as shown in Figure 7.3, where the service identification value is represented in the top label, and additional parameters are stacked beneath it. A parameter may be any information required by the tariff definition, such as a promised peak rate or expected data rate. A pricing mechanism could then charge for violations to these parameters or charge a fee when these parameters change. A QoS-ISP's ingress router pops the top label, and if further parameters are required for that QoS-Transit Service, further labels are popped. Forwarding decisions should be based on the top label, while charging and policing decisions may be based on the subsequent labels. To simplify our discussion, we will assume services do not require parameters.

Service	Exp	S	TTL
Parameter (peak bandwidth)	Exp	S	TTL

Figure 7.3: Stacked MPLS packets representing a QoS-Transit Service and a peak bandwidth parameter.

For an end-to-end path requiring the use of multiple QoS-Transit Services, the MPLS labels for the services may be stacked, providing a type of source-routing using MPLS. Figure 7.4 illustrates how label stacking can route packets across multiple networks. If a customer wants to use service 8 and service 12, respectively, of two adjoining QoS-ISPs (assuming service 8 of the first QoS-ISP routes the traffic to the next QoS-ISP), the

necessary MPLS labels must be present in her packet. The labels may be inserted by the customer, or may be inserted by the customer's B-ISP, forwarding the traffic transparently through the QoS-ISPs. The first network will route according to the top label. Upon exiting the first network, the egress LSR will pop the top label and forward the packet. The next network of the sequence will then route based on the new top label.

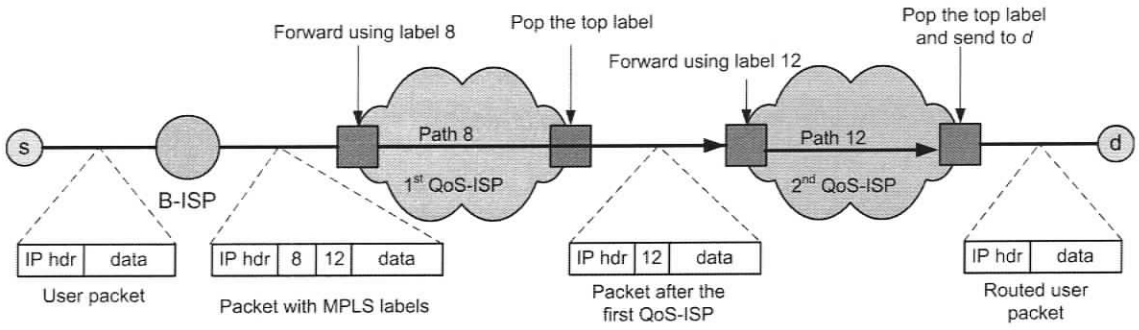


Figure 7.4: Label stacking through 2 networks.

The above situation is complicated if the fibre between the networks is owned, and controlled, by the downstream QoS-ISP. Consider the network in Figure 7.5, where the link connecting the two networks is owned and controlled by QoS-ISP$_2$. Additionally, consider a QoS-Transit Service with a source of router A and destination of router B.

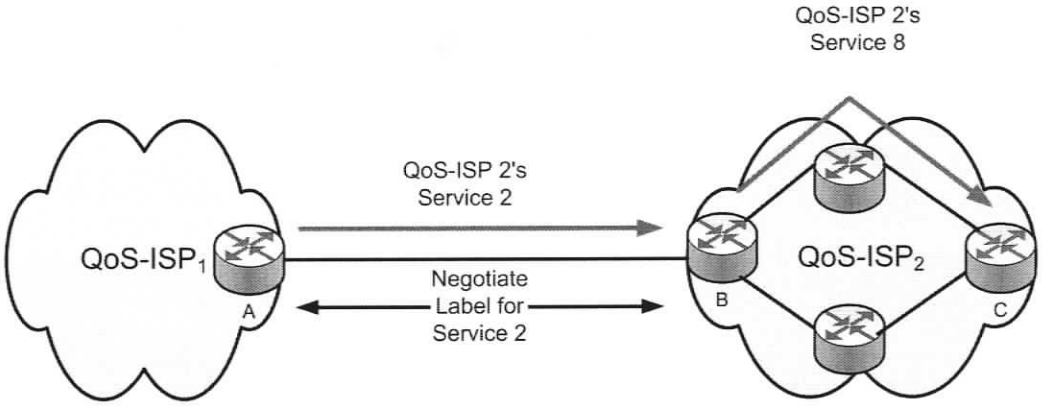


Figure 7.5: Routing across links between two connected providers.

As discussed in Section 6.3, the ingress router at the downstream router (router B) can perform the dynamic pricing function. Routing is performed as usual, with routers simply forwarding based on the top MPLS label of the label stack. Note that the label for routing through the upstream QoS-ISP will have already been popped at the penultimate

hop within the QoS-ISP. This means that, while the QoS-Transit Services offered across the interconnecting link belong to QoS-ISP_2, traffic must be routed onto the link by QoS-ISP_1. To accomplish this, adjacent routers between QoS-ISPs, such as routers A and B in Figure 7.5, must negotiate the use of an MPLS label across the link. More advanced relationships may see the negotiation of the choice of queuing and scheduling algorithms to use, in order to improve the offered QoS across such a link.

To simplify the knowledge required of the upstream egress router, the interconnecting link may support just a single QoS-Transit Service. However, there is no reason that only a single QoS-Transit Service can be purchased from each QoS-ISP. IP packets can use the QoS-Transit Service of the interconnecting link, and then another QoS-Transit Service to cross the network, as shown in Figure 7.5. Packets may include MPLS labels indicating the use of services 2, and then 8, of QoS-ISP_2's network.

7.3 Billing

A major requirement of any pricing implementation is the ability to bill the appropriate party. Recall that QoS-ISPs charge the B-ISPs, who in turn charge their customers. A B-ISP must identify each customer's flows in order to bill the appropriate party. A B-ISP can identify flows by examining the packet headers, or by associating customers with specific router interfaces. Each flow may be billed to a different entity, though we do not address how these associations are determined. The bill for a flow may be sent to the sender, receiver, a third party, or a combination of individuals. For example, in a multicast stream it is possible that the receivers may split the cost of the transmission. This is all handled easily because a B-ISP can bill the entities responsible for a flow based on a prior agreement.

A QoS-ISP's ingress router must be able to identify the B-ISP responsible for each packet. This is not necessarily for metering, but for policing, as we will explain later.

We propose that the B-ISP be identified in the IP header. There are two options for this: 1) identify the source address, or 2) use an IP Options field to define a unique Billing ID.

A B-ISP can tunnel the customers' data across the networks of the QoS-ISPs, making the source address in the outermost IP header the address of an egress router's interface of the B-ISP. A QoS-ISP can therefore determine which B-ISP is responsible for each packet. The limitation of this method is that each QoS-ISP must know the IP addresses of each of the egress routers of all the B-ISPs. This can be solved with the use of a single registry, accessible by all QoS-ISPs, containing the IP addresses of the B-ISPs along with any information required to bill them.

Alternatively, the B-ISP can append a *Billing ID* to the IP header of each of the packets. The *Billing ID* may be a governed global identification, similar to autonomous system numbers [46], or may simply be a selected IP address of the responsible B-ISP.

The IP version 4 (IPv4) header allows the inclusion of extra information through the optional *Options* fields [101]. Non-conforming routers will simply ignore the field [102]. However, the inclusion of an IPv4 Options field has been known to induce slight increases in delay and jitter, as well as severe increases in loss rate, in some router implementations [103]. Figure 7.6 illustrates the IPv4 header with the inclusion of the Billing ID. In IPv4 the overhead for the Billing ID is 16-bits plus the length of the Billing ID.

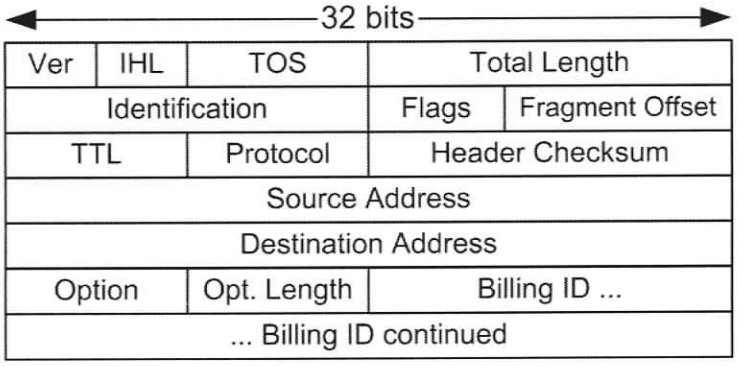


Figure 7.6: The IPv4 header with an option for indicating the Billing ID.

In IP version 6 (IPv6) [104] the Billing ID is added to the Hop-by-Hop extension header, which is examined and processed by all routers en route to the destination. The IPv6 header, including the Billing ID option in the Hop-by-Hop extension header, can be seen

in Figure 7.7. The overhead for the Hop-by-Hop extension header and the Billing ID in IPv6 is 32-bits plus the length of the Billing ID.

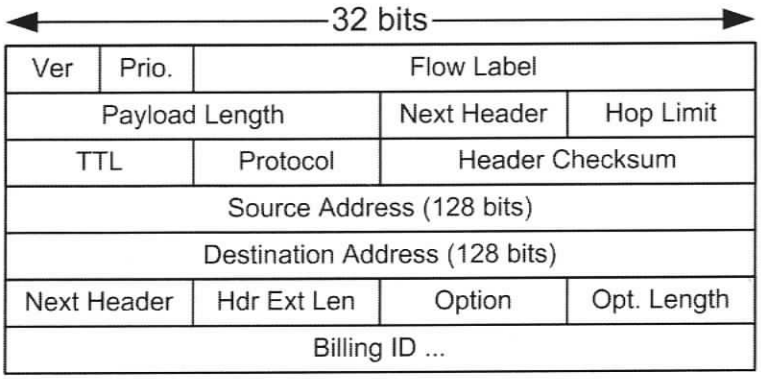


Figure 7.7: The IPv6 header with an option for indicating the Billing ID.

The Billing ID does provide some simplicity in the numbering, by allowing a single corporate entity to be represented by a single Billing ID, regardless of the number of egress interfaces that interacts with the QoS-ISPs. It does, however, result in an unnecessary overhead. Using the IP address of the egress router's interface makes use of already existing identification; however, it requires each QoS-ISP to recognize more addresses for billing or policing purposes.

7.4 Traffic Metering and Policing

We refer to *traffic metering* as the ability to determine how much traffic was transmitted by each entity for charging purposes. It is feasible for some networks to count every packet, or at least approximate this statistically by counting a percentage of all traffic, to reduce metering overheads. However, it is potentially very expensive to count packets entering a large network.

We propose that B-ISPs meter traffic flows because they are on the edge of the network, and because user traffic will exist as an electrical signal (i.e. non-optical) at this location in the network for some time in the future. A B-ISP may even perform probabilistic billing, by counting only a percentage of all traffic to reduce metering overheads. We contend that the B-ISP can therefore report to the QoS-ISPs how much data is being

transmitted onto the QoS-ISPs' networks, for billing purposes. The trustworthiness of this relationship will be discussed in the next section.

While QoS-ISPs depend on the B-ISPs to meter the traffic, QoS-ISPs may each want to perform their own policing functions to verify the B-ISPs' reported data. A QoS-ISP may meter only a percentage of traffic, or meter a specific subset of the traffic (both performable with NetFlow in Cisco's current Internetworking Operating System [105,106]). The QoS-ISPs results can be compared to the results reported by the B-ISPs. Any differences in the results could trigger legal or other corporate action.

7.5 Security Considerations

There are several potential security issues regarding the architecture defined in this chapter. We identify two such issues: *customer fraud* and *B-ISP fraud*. In *customer fraud*, a customer may fraudulently forge the source address in their packet headers to divert fees to another customer. In *B-ISP fraud*, a B-ISP may use another B-ISP's billing ID in packet headers to avoid being charged. Alternatively, a B-ISP may misreport metering information to the QoS-ISPs, either not reporting any information, or reporting less data than was actually transmitted.

We consider customer fraud of importance to the B-ISPs and B-ISP fraud of importance to the QoS-ISPs. There are four approaches we consider to these security problems: 1) legal action, 2) cascading trust, 3) policing, and 4) authentication. These approaches are not mutually exclusive and so may be used in any combination.

7.5.1 Legal Action

If we assume that it is an act of fraud to forge IP headers, or to use Billing IDs without permission, to avoid payment of a service, then most entities may not attempt to do so. The effectiveness of this method for preventing customers from illegally using the system is beyond the scope of this dissertation.

7.5.2 Cascading Trust

In cascading trust, each network does not accept traffic from untrustworthy networks. Each network, therefore, does not want to propagate traffic from a distrusted source, because the upstream network may block the current network as a result. It is thus the edge networks that are ultimately responsible for verifying that traffic is valid and from trustworthy sources. This is not to imply that non-edge networks will not perform policing functions, but they will police a lower percentage of traffic, partly because they assume their attached neighbours have already properly verified customer traffic, and partly because large amounts of traffic cross these core networks. Consequently, a B-ISP, at the edge, will not want to represent customers from untrustworthy networks, as it will fear action from QoS-ISPs. Each network thus has incentive to police their customers and/or users to assure they do not misbehave.

7.5.3 Policing

While relying on trust and legal measures may work in some situations, we also consider technological solutions to security. In this section we consider how policing can be implemented other than in the QoS-ISPs.

We define a *policing node* as a network entity that can properly police the transmitted traffic. *Traffic policing* is "the process of discarding packets... within a traffic stream" [35]. Packets may be discarded if a customer has been deemed untrustworthy or if he transmits higher than an agreed upon peak data rate.

Here we assume that policing nodes are trusted and secure. We suggest that the policing nodes be situated between the B-ISPs and the QoS-IPSs, as shown in Figure 7.8. These nodes can meter and police traffic, assuring no customer or B-ISP fraud occurs.

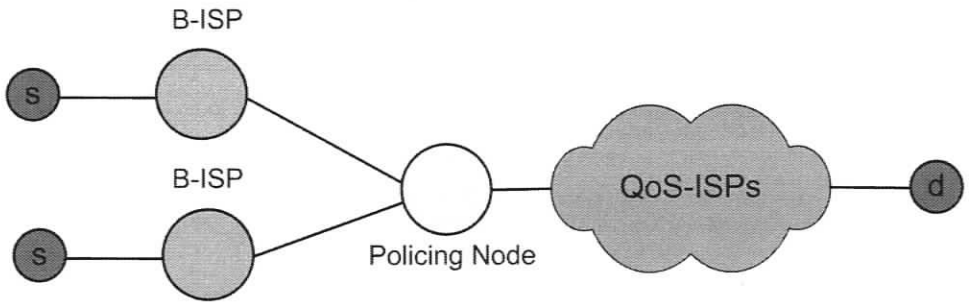


Figure 7.8: A policing node can be situated between one or more B-ISPs and their connection to the QoS-ISPs.

7.5.4 Authentication

We can also employ Internet security protocols [110,111,112] to assure that customers do not cheat. To prevent customer fraud, a secure tunnel can be created between a customer and a B-ISP. Preventing B-ISP fraud is a little more difficult. If all QoS-ISPs trust each other, and a B-ISP is directly connected to QoS-ISPs, then the ingress router for the first QoS-ISP must ensure that the Billing ID (or source address) matches the B-ISP from which it is receiving data – assuming data from different B-ISPs arrive on different router interfaces.

If the QoS-ISPs do not trust each other, then we propose a modification to the existing IPsec AH (Authentication Header) [111] protocol. Let a B-ISP distribute session keys to each of the QoS-ISPs for which a session flows. QoS-ISPs can authenticate packets and assure that the B-ISP has the right to use the embedded Billing ID by authenticating the packet against the distributed key. This differs from the usual AH protocol because the AH protocol specifies authentication to be done by end-systems, not by intermediate routers. The view of the IPsec tunnel in this scenario is shown in Figure 7.9 and the placement of the AH header in an IP packet when using tunnel mode is shown in Figure 7.10.

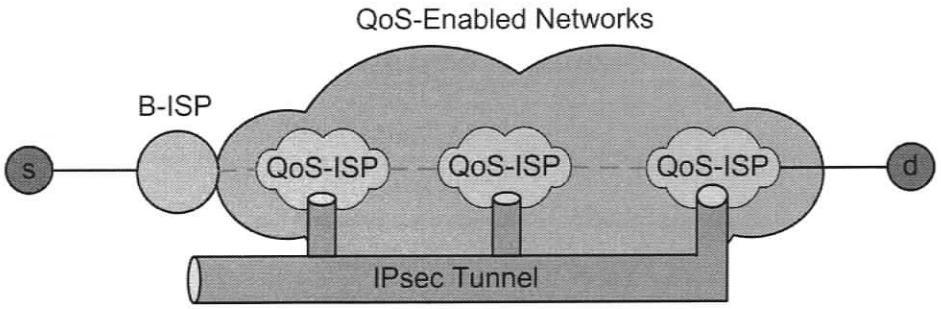


Figure 7.9: IPsec tunnel between Access Nodes, with ISP allowable authentication.

New IP header	AH	Original IP header	Payload (including other headers)

Figure 7.10: The AH protocol in tunnel mode.

There is a tradeoff of security for speed for the use of authentication. For high speed networks it may not be expected that every packet be authenticated. To improve throughput, not every packet should be authenticated, but only a subset. For example, every n^{th} packet could be authenticated, determined randomly, or all packets coming from untrustworthy customers.

One important note is that *IP Options* fields are not copied when using an IPsec tunnel. If a secure tunnel is used across the QoS-ISPs, it must be ensured that the outer IP header contains the *Billing ID*. Since we propose that the secure tunnel be established from the B-ISP, it is this entity that must assure the Billing ID exists in the outer header.

7.6 Summary

In this chapter we expanded on the concepts given in this dissertation by proposing a specific architecture that can be used to support the delivery of QoS-Transit Services, using existing network technologies.

We considered the deployment of QoS-Transit Services using the HCM (Hub Charging Model). A B-ISP is a core entity in our architecture, performing key roles in charging, as well as in routing. The B-ISP performs the broker functionality of the HCM, as well as tunnels the appropriate traffic across the QoS-Transit Services.

Routing in our sample architecture is achieved through MPLS label stacking. For each connection, a B-ISP tunnels traffic across a set of QoS-Transit Services between itself and the egress router of the last QoS-ISP the connection crosses. To specify the QoS-Transit Services the connection is routed across, an MPLS header is included with a stack of labels indicating the exact set of QoS-Transit Services. Each addressable node within the QoS-ISPs is assumed to be MPLS-capable and forwards traffic based on the top label.

According to the HCM, the QoS-ISPs charge brokers, which are the B-ISPs in this architecture. To identify the B-ISPs to charge, QoS-ISPs can either examine the source address of a tunneled packet, or require the B-ISPs to include a *Billing ID* within each packet.

QoS-ISPs may meter all incoming traffic, or they may perform statistical metering, considering every n^{th} packet. However, for very high speed networks this may not be achievable. Alternatively, the B-ISPs, which are on the edge of the network, can perform metering of their traffic and subsequently inform each of the QoS-ISPs the amount of traffic they are responsible for, and thus, the amount they should be charged themselves.

Since the presented architecture deals with money exchange, it is important to discuss security considerations. We presented four ways security can be realized: legal consequences, cascading trust, policing, and authentication. These security measures are not mutually exclusive. For example, cascading trust can be used, but a QoS-ISP may police the traffic from questionable B-ISPs to ensure that the B-ISP does not cheat the QoS-ISP.

8 Conclusion

This dissertation has demonstrated how policy control can be implemented end-to-end, to statistically guarantee QoS, by using dynamic pricing.

8.1 Contributions

The work within this dissertation focused on the definition, design, simulation, and optimal use of QoS-Transit Services. Our major contributions can be summarized as follows:

1. We developed the notion of QoS-Transit Services: a set of primitive services offered by an ISP in order to deliver packets with statistical performance guarantees within its network, using dynamic pricing. Further, we examined two pricing mechanisms and demonstrated, through simulation, that even a simple dynamic pricing mechanism can guarantee the QoS of QoS-Transit Services.
2. Since QoS-Transit Services are dynamically priced, we determine how QoS-Transit Services can be allocated network bandwidth in order to maximize the ISP's revenue. We first defined the *Optimal Bandwidth Allocation* problem as the problem of determining the allocation of bandwidth to QoS-Transit Services, routed on predefined paths, in order to maximize an ISP's revenue. We then defined the *Optimal Routing* problem as the problem of determining the paths to route each QoS-Transit Service, and the allocation of bandwidth to each QoS-Transit Service, in order to maximize the ISP's revenue. Additionally, each of these problems can be solved depending whether or not the demand functions of the QoS-Transit Services can be reasonably estimated.
    a. We define the Iterative Linear Programming (IterLP) and the Iterative Greedy (IterGreedy) heuristics to solve the Optimal Bandwidth Allocation problem when the demand functions of the QoS-Transit Services can be determined. IterLP achieves over 99% of an optimal solution, while taking

less than 2 seconds to execute for large data sets. IterGreedy achieves approximately 90-95% of the optimal solution, but executes much quicker than IterLP, making it useful for Optimal Routing heuristics that need to determine the optimal allocation of bandwidth for a large number of routing configurations.

b. We define the Service Grouping (SG), Iterative Bottleneck Avoidance (IBA), and Iterative Bottleneck Avoidance with Tabu (IBA-Tabu) heuristics to solve the Optimal Routing problem when the demand functions of the QoS-Transit Services can be determined. IBA executes extremely quick (under 1 second for many problem sizes), while achieving approximately 98% of the optimal solution when 20 or more QoS-Transit Services were considered (on a 30-node network). IBA-Tabu achieves greater optimality than IBA, but executes slower. The advantage of IBA-Tabu is clear when considering less than 40 services, achieving high revenue with moderate execution times. SG achieved the highest revenues of all the other heuristics for 40 services and less, but was also the slowest.

c. We define the Iterative Allocation Adjustment Heuristic (IAA) to solve the Optimal Bandwidth Allocation problem when no demand functions are assumed to be known. IAA was shown to achieve 97.7% optimality in 90% of test cases.

d. We propose the use of price information for QoS-Transit Services to value the links in the network in order to aid a traffic engineer to increase the revenue of an ISP. Using price information, as proposed within this dissertation, a traffic engineer can identify links in high (or low) demand, to reroute existing services, or to introduce new services across links in low demand.

3. We examined the issues relevant to providing end-to-end multi-network QoS with QoS-Transit Services. We proposed the use of a Hub Charging Model (HCM) to provide scalable charging between the purchasers and sellers of QoS-Transit Services. We also proposed using overlay networks to offer QoS-Transit Services without requiring the direct cooperation of the underlying network carriers.

4. Finally, we proposed a feasible architecture that can deliver QoS-Transit Services across multiple networks using existing protocol standards. We focused on the use of MPLS for routing, how billing can be achieved, how to meter and police traffic, as well as an overview of potential security considerations.

8.2 Future Work

The following are a few problems and issues that require future work:

- *The implementation of QoS-Transit Services for use with real traffic.* Traffic patterns may change with pricing [113], and we need to understand these patterns. The offering of QoS-Transit Services could be offered across an overlay network. This itself would lead to a large array of issues to study, including:
    - The *elasticity of demand of customers using QoS-Transit Services.* While we have estimated the elasticity of demand of customers, we do not know how customers would react to pricing changes for QoS-Transit Services.
    - *The implementation of customer agents.* Customers may want or need agents to manage the purchasing of QoS-Transit Services. The implementation of agents needs to consider not only the dynamic pricing of QoS-Transit Services, but also the relationship between customers and their users. What traffic should an agent route across QoS-Transit Services? When should an agent cease purchasing QoS-Transit Services for a particular flow?
    - *The effectiveness of various pricing mechanisms.* When dealing with real customers, are there certain properties of a pricing mechanism that are ideal in controlling traffic and maximizing revenue?
    - *The stability of prices with real customer behaviour.* Can a pricing mechanism maintain relatively stable prices in the short-term when faced with real customer behaviour? Constantly fluctuating prices may not be valued by customers.

- *The ability to guarantee QoS when different networks use different pricing mechanisms.* The demand for a given QoS-Transit Service may be dependent on the demand for other complement or substitute QoS-Transit Services. Is demand relatively stable, or does it oscillate between a set of competing services? Can QoS guarantees be maintained in the event of such oscillating demand? If such oscillations occur, can they be dampened through policies or by changing the tariff definition of the QoS-Transit Services?
- *The determination of the paths across the QoS-Transit Services.* We have assumed that a broker, or other market data provider, can determine which QoS-Transit Services can be used to route packets to the destination of a particular connection. Processes need to be defined to demonstrate how this can be achieved on-line, as new connections are created.

- *Further study of implementing security and its effect on architectures.* While security was only naively approached within this dissertation, an in-depth study needs to be done on how to implement security while considering the effects of any security measures on deployable architectures.

References

[1] A.M. Odlyzko, "Internet Traffic Growth: Sources and Implications," *Optical Transmission Systems and Equipment for WDM Networking II* (Editors B.B. Dingel, W. Weiershausen, A.K. Dutta, and K.-I. Sato), Proc. SPIE, vol. 5247, 2003, pp. 1-15.

[2] S. Iyer, S. Bhattacharyya, N. Taft, and C. Diot, "An Approach to Alleviate Link Overload as Observed on an IP Backbone," *Proceedings of IEEE INFOCOM 2003*, San Francisco, CA, April 2003, pp. 406-416.

[3] K. Mitchell, "The Future of Core Routing," Online Presentation, Light Reading Inc, www.lightreading.com, September 8, 2004.

[4] K. Cho, K. Fukuka, H. Esaki, and A. Kato, "The Impact of Residential Broadband Traffic on Japanese ISP Backbones," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 1, January 2005, pp. 15-25.

[5] Hybrid Ethernet/SDH Transport Networks, White Paper, Lucent Technologies, February 13, 2004

[6] C. Courcoubetis and R. Weber, *Pricing Communication Networks: Economics, Technology, and Modelling*, England: John Wiley & Sons Ltd, 2003.

[7] G. Hardin, "Tragedy of the Commons," *Science*, vol. 162, December 13, 1968, pp. 1243-1248.

[8] M. Falkner, M. Devetsikiotis, and I. Lambadaris, "An Overview of Pricing Concepts for Broadband IP Networks," *IEEE Communications Surveys*, vol. 3, no. 2, 2000.

[9] F. Kelly, A.K. Maulloo, and D.K.H. Tan, "Rate Control for Communication Networks: Shadow Prices, Proportional Fairness, and Stability," *Journal of the Operational Research Society*, vol. 49, 1998, pp. 237-252.

[10] J.K. MacKie-Mason, "A Smart Market for Resource Reservation in Multiple Quality of Service Information Network," Technical Report, Department of Economics, University of Michigan, September 1997.

[11] J.K. MacKie-Mason and H. Varian, "Some Economics of the Internet," Technical Report, University of Michigan, February 1994.

[12] C. Courcoubetis, M.P. Dramitinos, and G.D. Stamoulis, "An Auction Mechanism for Bandwidth Allocation over Paths," *Proceedings of International Teletraffic Congress ITC-17*, Salvador da Bahia, Brazil, December 2001, pp. 1163-1173.

[13] A.M. Odlyzko, "A Modest Proposal for Preventing Internet Congestion," *DIMACS Technical Report 97-68*, AT&T Labs, October 1997.

[14] D. Songhurst and F. Kelly, "Charging Schemes for Multiservice Networks," *IEE Colloquium on Charging for ATM*, London, UK, November 1996, pp. 4/1-4/8.

[15] J.K. MacKie-Mason and H. Varian, "Pricing Congestible Network Resources," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 7, September 1995, pp. 1141-1149.

[16] A. Ganesh, K. Laevens, and R. Steinberg, "Congestion Pricing and User Adaptation," *Proceedings of IEEE INFOCOM 2001*, Anchorage, AK, April 2001, pp. 959-965.

[17] X. Wang and H. Schulzrinne, "An Integrated Resource Negotiation, Pricing, and QoS Adaptation Framework for Multimedia Applications," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 12, 2000.

[18] T. Li, Y. Iraqi, and R. Boutaba, "Pricing and Admission Control for QoS-enabled Internet," *Computer Networks*, Special issue on Internet Economics, vol. 6, no. 1, September 2004, pp 87-110.

[19] H. Kneer and R. Marfurt, "A Contracting Protocol for Managing Quality of Service in a Multi-Provider Environment," *Proceedings of IADIS International Conference WWW/Internet 2002*, Lisbon, Portugal, November 2002, pp. 230-238.

[20] Terje Jensen, Irena Grgic, and Ola Espvik, "Managing Quality of Service in Multi-Provider Environment," *Proceedings of Telecom 99*, Geneva, October 1999.

[21] A. Odlyzko, "'Smart' and 'Stupid' Networks: Why the Internet is like Microsoft," *netWorker*, Vol. 2, No. 5, December 1998, pp. 38-46.

[22] A. Odlyzko, "The Stupid Network: Essential Yet Unattainable?," *netWorker*, Vol. 3, No. 4, December 1999, pp. 36-37.

[23] D. Isenberg and D. Weinberger, "The Paradox of the Best Network," www.netparadox.com, 2001.

[24] G. Cheliotis, "Structure and Dynamics of Bandwidth Markets," Ph.D. Dissertation, Department of Electrical Engineering and Computer Science, National Technical University of Athens, November 2001.

[25] P. Grant and J. Drucker, "Phone, Cable Firms Rein in Consumers' Internet Use: Big Operators See Threat to Service as Web Calls, Videos Clog Up Networks," *The Wall Street Journal*, October 21, 2005, pp. A1.

[26] P. Davidson, "Web Traffic Jams Bring Fight over Fast-Lane Fees: Phone companies plan to charge content providers," *USA Today*, February 7, 2006, pp. 2B.

[27] D. Isenberg, "The Dawn of the 'Stupid Network'," *netWorker*, vol. 2, no. 1, February/March 1998, pp. 24-31.

[28] J.K. MacKie-Mason and S. Shenker, "Service Architecture and Content Provision: The Network Provider as Editor," *Telecommunication Policy*, vol. 20, no. 3, April 1996, pp. 203-217.

[29] W. Stevens, "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," *IETF RFC 2001*, January 1997.

[30] V. Jacobson and M.J. Karels, "Congestion Avoidance and Control," *Proceedings of SIGCOMM '88*, August 1988, pp. 314-329.

[31] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet," *IETF RFC 2309*, April 1998.

[32] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, August 1993, pp. 397-413.

[33] K. Ramakrishnan and S. Floyd, "A Proposal to add Explicit Congestion Notification (ECN) to IP," *IETF RFC 2481*, January 1999.

[34] T. Kelly, "An ECN Probe-Based Connection Acceptance Control," *Computer Communication Review*, vol. 31, no 3, July 2001, pp. 14-25.

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

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

[37] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP) Version 1 Functional Specification," *IETF RFC 2205*, September 1997.

[38] S. Shenker, C. Partridge, and R. Guerin, "Specification of Guaranteed Quality of Service," *IETF RFC 2212*, September 1997.

[39] J. Wroclawski, "Specification of the Controlled-Load Network Element Service," *IETF RFC 2211*, September 1997.

[40] G. Armitage, *Quality of Service in IP Networks: Foundations for a Multi-Service Internet*, Indianapolis: Macmillan Technical Publishing, 2000.

[41] S. Khan, "Quality Adaptation in a Multisession Multimedia System: Model, Algorithms and Architecture," Ph.D. Dissertation, Department of Computer Science, University of Victoria, 1998.

[42] R.K. Watson, "Applying the Utility Model to IP Networks," M.Sc. Thesis, Department of Computer Science, University of Victoria, 2001.

[43] M.M. Akbar, E.G. Manning, R. Watson, G.C. Shoja, S. Khan, and K.F. Li, "Optimal Admission Controllers for Service Level Agreements in Enterprise Networks," *Proceedings of the 6th World Multiconference on Systemics, Cybernetics, and Informatics*, Orlando, Florida, July, 2002.

[44] M.M. Akbar, "The Distributed Utility Model Applied to Optimal Admission Control & QoS Adaptation in Multimedia Systems and Enterprise Networks," Ph.D. Dissertation, Department of Computer Science, University of Victoria, 2002.

[45] S. Shelford, M.M Akbar, E.G. Manning, and G.C. Shoja, "Distributed Optimal Admission Controllers for Service Level Agreements in Interconnected Networks," *Proceedings of the 21st IASTED International Conference on Applied Informatics*, Innsbruck, Austria, February 10-13, 2003, pp. 565-570.

[46] J. Hawkinson, and T. Bates, "Guidelines for Creation, Selection, and Registration of an Autonomous System (AS)," *IETF RFC 1930*, March 1996.

[47] B. Briscoe, A. Odlyzko and B. Tilly, "Metcalfe's Law is Wrong," *IEEE Spectrum*, July 2006, pp. 26-31.

[48] P. Ferreira, "A Model for Interconnection of IP Networks," Qualifier Paper, Carnegie Mellon University, January 6, 2003.

[49] C. Metz, "Interconnecting ISP Networks," *IEEE Internet Computing*, vol. 5, no. 2, March/April 2001, pp. 74-80.

[50] W.B. Norton, "Internet Service Providers and Peering," *Proceedings of NANOG 19*, Albuquerque, New Mexico, June 2000.

[51] P. Pan, "Scalable Resource Reservation Signaling in the Internet," Ph.D. Dissertation, Graduate School of Arts and Sciences, Columbia University, 2002.

[52] A. Akella, S. Seshan, and A. Shaikh, "An Empirical Evaluation of Wide-Area Internet Bottlenecks," *Proceedings of the 2003 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, San Diego, CA, USA, October 2003, pp. 316-317.

[53] IST Mescal project: Interdomain QoS for the Internet, "Management of End-to-end Quality of Service Across the Internet at Large (mescal)," http://www.mescal.org (Accessed 10 January 2005).

[54] P. Georgatsos, J. Spencer, D. Griffin, T. Damilatis, H. Asgari, J. Griem, G. Pavlou, and P. Morand, "Provider-Level Service Agreements for Inter-Domain QoS Delivery," Fourth International Workshop on Advanced Internet Charging and QoS Technologies (ICQT04), *Lecture Notes in Computer Science*, vol. 3266, Springer, September 2004, pp. 368-377.

[55] "Merkato Overview: A platform for real-time market-based network resource allocation," Technical Paper, InvisibleHand Networks Inc., Revision 2, 2002.

[56] "StreamingHand: An In-depth Look," Technical Paper, InvisibleHand Networks Inc., Revision 2, May 23, 2002.

[57] R.J. Edell and P. Varaiya, "Providing Internet Access: What we learn from the INDEX Trial," *INDEX Project Report 99-010W*, April 1999.

[58] J. Altmann, B. Rupp, and P. Varaiya, "The Case for Quality of Service on Demand - Empirical Evidence from the INDEX Project," *ISQE'99, Workshop on Internet Service Quality Economics*, Cambridge, MA, USA, December 1999.

[59] L. DaSilva, "Pricing for QoS-Enabled Networks: A Survey," *IEEE Communications Surveys*, vol. 3, no. 2, 2000.

[60] P. Kirkby, "Business Models and System Architecture for Future QoS Guaranteed Internet Services," *IEE Colloquium on Charging for ATM – The Reality Arrives*, November 1997, pp. 11/1-11/8.

[61] J.K. MacKie-Mason and H. Varian, "Generalized Vickrey Auctions," Working Paper, Department of Economics, University of Michigan, July 1994.

[62] T. Henderson, J. Crowcroft, and S. Bhatti, "Congestion Pricing: Paying Your Way in Communication Networks," *Internet Computing*, IEEE, vol. 5, no. 5, September 2001, pp. 85-89.

[63] S. Shenker, D. Clark, D. Estrin, and S. Horzog, "Pricing in Computer Networks: Reshaping the Research Agenda," *ACM Computer Communication Review*, vol. 26, no. 2, April 1996, pp. 19-43.

[64] F.P. Kelly, "Notes on Effective Bandwidth," In Stochastic Networks: Theory and Applications (Editors F.P. Kelly, S. Zachary and I.B. Ziedins), Royal Statistical Society Lecture Notes Series, vol. 4, Oxford University Press, 1996, pages 141-168.

[65] Robert Rosenberg, President, Insight Research Corporation, May 1, 2001.

[66] C. Lord and M. Joshi, *A Study of the History and Development of Bandwidth Commodity Trading*, Term Report, 46-875: Corporate Telecommunications Networks, Carnegie Mellon University, June 28th, 2002.

[67] S. Clavenna, J. Chacko, F. Serio, and M. Coronaro, "Building a Successful Full Service Packet Voice and IP Network," Online Presentation, Light Reading Inc, www.lightreading.com, October 1, 2003.

[68] Enron Communications, "Announces First Commodity Bandwidth Trade," *Press Release*, December 2, 1999.

[69] Platts, "Bandwidth.com aims to be the 'Travelocity of bandwidth,' keying on OTC enterprise market," *Bandwidth Market Report*, April 30, 2001.

[70] K. Henderson, "The Stage is Set: Enter Liquid Bandwidth Trading," Transport Commerce, *XChange Magazine*, January 2001.

[71] S. Moyer and A. Umar, "The Impact of Network Convergence on Telecommunications Software," *IEEE Communications Magazine*, vol. 39, no. 1, January 2001, pp. 78-84.

[72] J.L. Mindel and M.A. Sirbu, "Taxonomy of Traded Bandwidth," Working Paper, Engineering and Public Policy Department, Carnegie Mellon University, May 1, 2001.

[73] U. Black, *MPLS and Label Switching Networks*, 2nd ed., New Jersey: Prentice Hall PTR, 2002.

[74] P. Ferreira, J. Mindel, and L. McKnight, "Why Bandwidth Trading Markets Haven't Matured? Analysis of Technological and Market Issues," *International Journal of Technology, Policy, and Management (IJTPM)*, vol. 3, no. 2, 2003.

[75] S. Khan, K. Li, E.G. Manning, R. Watson, and G.C. Shoja, "Optimal Quality of Service Routing and Admission Control Using the Utility Model," *Future Generation Computer Systems*, vol. 19, no. 7, October 2003, pp. 1063-1073.

[76] J. Saltzer, D. Reed, and D. Clark, "End-To-End Arguments in System Design," *ACM Transactions on Computer Systems*, vol. 2, no. 4, November 1984, pp. 277-288.

[77] K. Fall and K. Varadhan, Eds., "The ns Manual", The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC, http://www.isi.edu/nsnam/ns/doc/ns_doc.pdf, December 13, 2003.

[78] "BT Global Services: Monthly Network Details," BT Global Services, [online] 2005, http://ippm.bt.net/month-200505/index.shtml (Accessed: 11 August 2005).

[79] R. Guerin, H. Ahmadi, and M. Naghshineh, "Equivalent Capacity and Its Application to Bandwidth Allocation in High-Speed Networks," *IEEE Journal on Selected Areas in Communication*, vol. 9, no. 7, September 1991, pp. 968-981.

[80] V. Klee and G.J. Minty, "How Good is the Simplex Algorithm?," *Inequalities 3* (Editor O. Shisha), vol. 3, New York: Academic Press, 1972, pp. 159-175.

[81] D.A. Spielman and S. Teng, "Smoothed Analysis of Algorithms: Why the Simplex Algorithm Usually Takes Polynomial Time," *Journal of the ACM* (JACM), vol. 51 , no. 3, May 2004, pp. 385-463.

[82] N. Karmarkar, "A New Polynomial-Time Algorithm for Linear Programming," Combinatorica, vol. 4, no. 4, 1984, pp. 373-395.

[83] M. Berkelaar, K. Eikland, and P. Notebaert, "Open source (mixed-integer) linear programming system," lp_solve version 5.5.0.0, code available at http://groups.yahoo.com/group/lp_solve/files/Version5.5/, May 17, 2005.

[84] A. Hertz, E. Taillard, and D. de Werra, "A Tutorial on Tabu Search," in *Proceedings of Giornate di Lavoro AIRO'95 (Enterprise Systems: Management of Technological and Organizational Changes)*, Italy, 1995, pp. 13-24.

[85] S. Shelford, "Implementation, Evaluation, and Application of Distributed Heuristics for Optimizing SLA Admission," M.Sc. Thesis, Department of Computer Science, University of Victoria, 2003.

[86] Eurescom Project P1008, "Inter-Operator Interfaces for Ensuring End-to-End IP QoS," Deliverable 2, Selected Scenarios and requirements for end-to-end IP QoS management, January 2001.

[87] S. Herzog, S. Shenker, and D. Estrin, "Sharing the 'Cost' of Multicast Trees: An Axiomatic Analysis," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, December 1997, pp 847-860.

[88] B.E. Hermalin and M.L. Katz, "Sender or Receiver: Who should pay to exchange an electronic message?," *Rand Journal of Economics*, vol. 35, no. 3, 2004, pp. 423-447.

[89] V. Fuller, T. Li, J. Yu, and K. Varadhan, "Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy," *IETF RFC 1519*, September 1993.

[90] C. Wang, C. Hsu, and Y. Huang, "VORAL: a system for voice over IP routing in application layer," *Proceedings of Seventh IEEE Real-Time Technology and Applications Symposium*, Taipei, Taiwan, May 2001, pp. 165-170.

[91] "P2P telephony explained," Skype, [online] 2004, http://www.skype.com/skype_p2pexplained.html (Accessed: 2 April 2004).

[92] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson, "The end-to-end effects of Internet path selection," *Proceedings of ACM SIGCOMM*, September 1999, pp. 289-299.

[93] C. Labovitz, A. Ahuja, and F. Jahanian, "Experimental Study of Internet Stability and Wide-Area Backbone Failures," *Technical Report CSE-TR382-98*, University of Michigan, 1998.

[94] D. Szebesta, "Virtual Reality – the Pros and Cons of the VNO Model," White Paper, COLT Telecommunications, May 2004.

[95] "Internet Dedicated, Global Transit Service Level Agreement," MCI, October 2004.

[96] "Service Level Agreement," infinity internet, online, May 13, 2005.

[97] S. Shalunov and B. Teitelbaum, "One-way Active Measurement Protocol (OWAMP) Requirements," *IETF RFC 3763*, April 2004.

[98] "OC48 – High performance rack mountable network monitoring system," Endace, [online] 2005, http://www.endace.com/oc48mon.htm (Accessed: 12 September 2005).

[99] E. Rosen, D. Tappan, G. Fedorkow, Y. Rekhter, D. Farinacci, T. Li, and A. Conta, "MPLS Label Stack Encoding," *IETF RFC 3032*, January 2001.

[100] F. Faucheur, L. Wu, B. Davie, S. Davari, P. Vaananen, R. Krishnan, P. Cheval, and J. Heinanen, "Multi-Protocol Label Switching (MPLS) Support of Differentiated Services," *IETF RFC 3270*, May 2002.

[101] J. Postel, Ed., "Internet Protocol: DARPA Internet Program Protocol Specification," *IETF RFC 791*, September 1981.

[102] F. Baker, "Requirements for IP Version 4 Routers," *IETF RFC 1812*, June 1995.

[103] P. Fransson and A. Jonsson, "End-to-End Measurements on Performance Penalties of IPv4 Options," *Proceedings of GLOBECOM 2004*, Dallas, Texas, November 29 - December 3, 2004.

[104] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," *IETF RFC 2460*, December 1998.

[105] "Cisco IOS NetFlow overview," Technical Paper, Cisco Systems, May 2, 2005.

[106] "MPLS Egress NetFlow Accounting," Cisco IOS Release 12.1(5)T, http://www.cisco.com, March 23, 2005.

[107] D. Maughan, M. Schertler, M. Schneider, and J. Turner, "Internet Security Association and Key Management Protocol (ISAKMP)," *IETF RFC 2408*, November 1998.

[108] D. Harkins and D. Carrel, "The Internet Key Exchange (IKE)," *IETF RFC 2409*, November 1998.

[109] H. Harney and C. Muckenhirn, "Group Key Management Protocol (GKMP) Specification," IETF RFC 2093, July 1997.

[110] U. Black, *Internet Security Protocols: Protecting IP Traffic*, New Jersey: Prentice Hall PTR, 2000.

[111] S. Kent and R. Atkinson, "IP Authentication Header," *IETF RFC 2402*, November 1998.

[112] S. Kent and R. Atkinson, "IP Encapsulating Security Payload (ESP)," *IETF RFC 2406*, November 1998.

[113] Z. Sahinoglu and S. Tekinay, "On Multimedia Networks: Self-Similar Traffic and Network Performance," *IEEE Communications Magazine*, vol. 37, no. 1, January 1999, pp. 48-52.

[114] M. Shreedhar and G. Varghese, "Efficient Fair Queuing using Deficit Round Robin," *Proceedings of ACM SIGCOMM '95*, September 1995, pp. 231-242.

[115] "NLANR PMA: Special Traces Archive," *NLANR PMA*, [online] 2005, http://pma.nlanr.net/Special/ (Accessed: 11 August 2005).

Appendix A: Simulation Setup

Implementation

ns-2 is a network simulator that uses modules implemented in C++, but are controllable with test scripts written in Tcl. We implemented the ULT pricing mechanism in C++, compiling it into ns-2. The Tcl scripts setup the simulated network and the flow of traffic, loading the necessary modules and connecting them together.

Within the ULT module, a periodic timer is used to determine price changes, and then delay their signaling to the customer. The added delay simulates the round trip time to the customer, which is how long the network must wait for the customer to receive the price change and for the altered traffic to reach the network. The ULT module interacts with the Tcl test script through predefined function calls, to inform the script of price updates. The Tcl script can then adjust the flow rates in response, simulating the customers' behaviour. The structure of this mechanism is shown in Figure A.1.

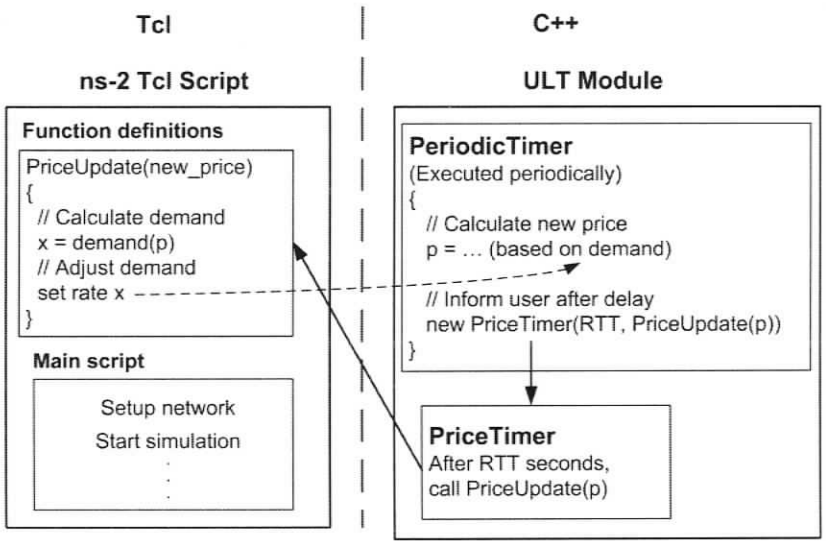


Figure A.1: Interaction between the C++ pricing module and a Tcl script.

The ULT pricing mechanism was implemented by extending the *Queue* class in ns-2. By monitoring the amount of data exiting a queue, we can determine the bandwidth used by its associated QoS-Transit Service, and thus adjust the price to use that service. For queue overflows, tail packets are simply dropped.

Each QoS-Transit Service has a separate queue. Since many QoS-Transit Services may use the same link, we use Deficit Round Robin [114] scheduling to remove packets from each QoS-Transit Service's queue, to the outgoing link. Deficit Round Robin is quite similar to Weighted Fair Queuing, but it has a faster running time. For the evaluation in this dissertation, we chose an equal weighting between the QoS-Transit Services.

The bandwidth for a period is calculated as an exponentially weighted moving average, as shown in Equation 3.3, with a weight of 0.2 for the measurement over the most recent interval:

$$x(t) = 0.2x_{measure}(t) + 0.8 \cdot x(t-1), \tag{A.1}$$

where $x(t)$ is the exponentially weighted average demand calculated for time t and $x_{measure}(t)$ is the measured (non-weighted) average demand at time t.

Prices are adjusted linearly based on how much the current demand, x, differs from the goal demand, x_{goal}. Additionally, prices are adjusted by a maximum of γ percent. The price adjustments are considered as follows:

$$p(t) = \begin{cases} \gamma \cdot p(t-1), & \dfrac{x_{goal} - x}{g} < -1 \\ -\gamma \cdot p(t-1), & \dfrac{x_{goal} - x}{g} > 1 \\ -\gamma \cdot p(t-1) \cdot (x_{goal} - x)/g, & \text{otherwise,} \end{cases} \tag{A.2}$$

where $p(t)$ is the new price, $p(t-1)$ was the previous price, and g is the price radius of ULT, which is the distance of x from x_{goal} for a maximum price change to occur.

Demand

We do not model individual users, but instead we model aggregates of users. Individuals may be hard to model accurately, as their individual strategies may be complex. To model users we make assumptions about aggregate demand functions, as well as traffic characteristics.

To model an aggregate demand function, we consider the demand functions estimated in the INDEX experiment [57]. Using their equations, we derived (see Appendix B: Demand Function) the following demand function, as illustrated in Figure A.2, which we will use as the basis for our experiments:

$$x = 10^{0.37(1-\log p)}. \tag{A.3}$$

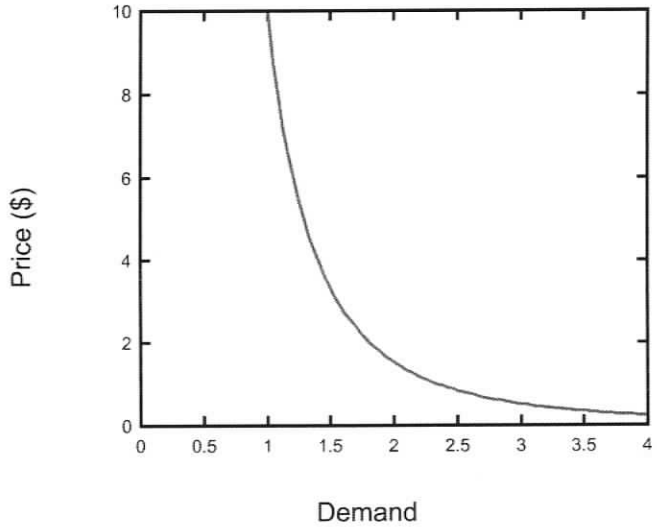


Figure A.2: The shape of the aggregate demand curve.

As our services are different from those studied in the INDEX experiment, we do not contend this to be a perfect representation of demand, but it gives us a plausible curve to use.

Notice that demand is not represented with any units. This is because we will use the demand curve for a relative measure of demand.

Simulating Traffic

We use traffic trace files to insert traffic into our simulations, simulating the aggregate traffic of customers. All packets in a trace are sent across a link as UDP packets.

The value x in Equation A.3 actually represents a probability that a packet will be sent to the network, where x is scaled appropriately for each experiment. Consequently, as x increases, more packets will be transmitted. Since the bandwidth requirements of the aggregated traffic are not constant, the traffic infused into the network is not constant. This conveniently helps reflect that a demand curve is not constant. Since demand is calculated as a percentage of packets transmitted, the load to the network is not constant over any time scale.

For $x \leq 100$, we transmit each packet in the traffic traces with a probability of x. For $x > 100$, we ensure that each packet is sent in the trace file at least once. Let $a = \lfloor x/100 \rfloor$ and $b = x \bmod 100$. For $x > 100$, we send each packet a times, and send another with a probability of b. That is, if demand is set to 410%, then we will transmit all packets in the trace file 4 times (at once), while 10% of the time transmitting the packets 5 times, thus generating 410% of demand from the trace file. However, we setup the simulations with enough traffic traces to assure $x < 100$ at most times. Additionally, for each simulation, we must choose at what price demand will equal 100% of the traffic trace.

To allow data within a traffic trace to be transmitted with a specified probability, we modified the Application/Traffic/Trace class (ns/trace/traffictrace.cc). A *percent_* field was added to allow the Tcl scripts to signal the change of demand, as calculated within the script.

Traffic Traces

The traffic in our simulations consists of traffic traces from a number of sources: movies, an access network link, and a core network link.

For the movies, we consider MPEG videos that are streamed with an average data rate of 256 kbps. Since the videos have similar characteristics with one another, and are very periodic, sending a frame every 30^{th} of a second, the data rate smoothes nicely when a large number of videos are aggregated.

The bandwidth required from a single MPEG movie over time is shown in Figure A.3, where the bandwidth is averaged over 1-second intervals. Even a single movie does not vary much in the short term, and thus the aggregation of a large number of such movies on a single link results in traffic with a near-constant demand for bandwidth.

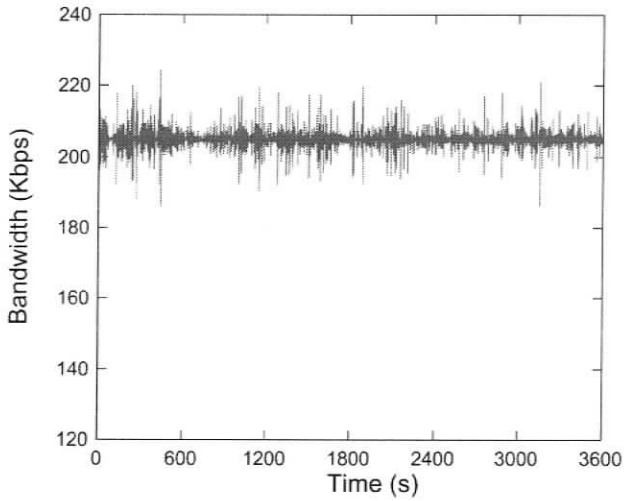


Figure A.3: Sample average bandwidth over time of a single MPEG-2 encoded movie.

To find a more bursty trace, we consider the large traffic traces in the Special Traces Archive from the National Laboratory for Applied Network Research's Passive Measurement and Analysis group [115]. A sample 4-minute period is shown in Figure A.4 for a trace from a backbone link, where the bandwidth is averaged over 1-second intervals. The trace shows short-term traffic bursts, but nothing extreme.

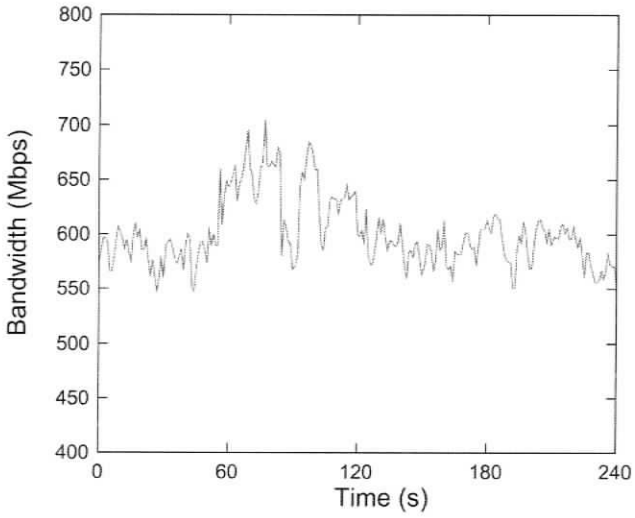


Figure A.4: Sample average bandwidth over time on a core link.

Finally, to find even more bursty traffic, we consider further traces from the National Laboratory for Applied Network Research's Passive Measurement and Analysis group [115]. Specifically, we chose a 70-minute trace of the traffic inbound to the University of Leipzig on November 23rd between 2am and 3:10am (see Figure A.5 for a 4-minute sample). The trace has many large short-term fluctuations compared to similar traces at other times. These large, short-term, demand spikes test the ability of our pricing mechanism to maintain advertised QoS in the event of such spikes in demand. The traffic over the entire period ranged from 12 to 33 Mbps.

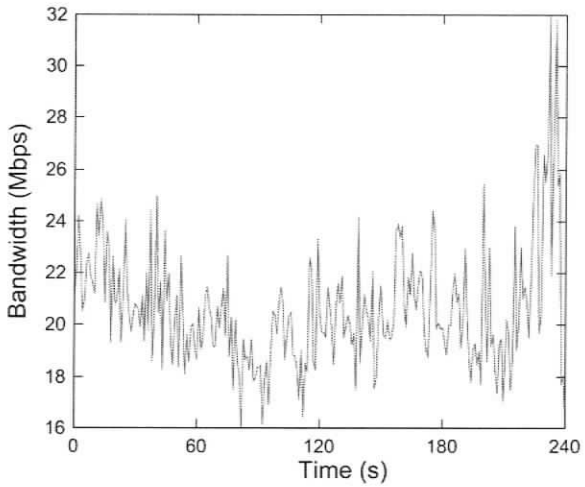


Figure A.5: Sample average bandwidth over time on an access link.

Appendix B: Demand Function

To simulate customers, we first need to model how they react to pricing changes. The only experiment we know of that has addressed this behaviour through experimentation is the INDEX experiment [57,58]. We use the results from INDEX to formulate an aggregate demand function for our work.

In [57], the authors derived demand functions based on empirical results of offering 5 services (16 kbps, 32 kbps, 64 kbps, 96 kbps, and 128 kbps) at 5 different prices (p_{16}, p_{32}, p_{64}, p_{96}, and p_{128}). The demand functions considered the prices of competing services through their *cross-elasticity of demand*. Demand generally increases/decreases when the price of competing services goes up/down, but demand increases/decreases when the price of the actual service goes down/up.

The demand functions from the INDEX experiment are as follows:

$$\begin{aligned}
\log x_{128} &= 4.10 - 1.65\log p_{128} + 0.44\log p_{96} + 0.55\log p_{64} - 0.12\log p_{32} + 0.00\log p_{16} \\
\log x_{96} &= 2.60 + 1.23\log p_{128} - 3.34\log p_{96} + 1.17\log p_{64} + 0.23\log p_{32} + 0.00\log p_{16} \\
\log x_{64} &= 2.70 + 0.08\log p_{128} + 0.84\log p_{96} - 1.71\log p_{64} + 0.47\log p_{32} + 0.55\log p_{16} \quad \text{(B.1)} \\
\log x_{32} &= 2.33 + 0.48\log p_{128} - 0.58\log p_{96} + 0.88\log p_{64} - 1.10\log p_{32} + 0.08\log p_{16} \\
\log x_{16} &= 0.52 + 0.42\log p_{128} - 0.26\log p_{96} + 0.18\log p_{64} + 0.97\log p_{32} - 1.29\log p_{16}.
\end{aligned}$$

Assume prices per kbps are constant, irrespective of the speed of the service: i.e. $p_{32} = 32p$. (It should be noted that prices were based on a per minute basis.) Then, let the above demand functions be defined by:

$$\log x_a = b + c\log p128 + d\log p96 + f\log p64 + g\log p32 + h\log p16, \qquad \text{(B.2)}$$

where a is the speed of the service in kbps (128, 96, 64, 32, or 16). We then simplify:

$$\begin{aligned}
\log x_a &= b + c\log(128p) + d(\log 96p) + f\log(64p) + g\log(32p) + h\log(16p) \\
\log x_a &= b + c(\log 128 + \log p) + d(\log 96 + \log p) + f(\log 64 + \log p) \\
&\quad + g(\log 32 + \log p) + h\log(16 + \log p) \\
\log x_a &= b + c\log 128 + d\log 96 + f\log 64 + g\log 32 + h\log 16 \\
&\quad + (c + d + f + g + h)\log p \\
x_a &= 10^{b+c\log 128+d\log 96+f\log 64+g\log 32+h\log 16+(c+d+f+g+h)\log p}.
\end{aligned} \tag{B.3}$$

Normalizing each function to a 1 kbps service results in the following:

$$\begin{aligned}
x &= 10^{b+c\log 128+d\log 96+f\log 64+g\log 32+h\log 16+(c+d+f+g+h)\log p} / a \\
x &= 10^{b+c\log 128+d\log 96+f\log 64+g\log 32+h\log 16+(c+d+f+g+h)\log p} / 10^{\log a} \\
x &= 10^{b+c\log 128+d\log 96+f\log 64+g\log 32+h\log 16+(c+d+f+g+h)\log p-\log a}.
\end{aligned} \tag{B.4}$$

The normalized demand curves are calculated as follows:

$$\begin{aligned}
x_{128} &: x = 10^{0.20087391135-0.78\log p} \\
x_{96} &: x = 10^{-0.951773824150120.71\log p} \\
x_{64} &: x = 10^{1.00862338409984+0.23\log p} \\
x_{32} &: x = 10^{0.71669649151451-0.24\log p} \\
x_{16} &: x = 10^{-0.08268921933284+0.02\log p}.
\end{aligned} \tag{B.5}$$

The normalized curves for x_{16} and x_{64} are not valid demand curves, as demand for them actually increases when prices increase. This is due to our assumption that the price per kbps is constant regardless of the service. We average the normalized curves for x_{128}, x_{96}, and x_{32}, resulting in:

$$\begin{aligned}
x &= (10^{0.20087391135-0.78\log p} + 10^{-0.951773824150120.71\log p} + 10^{0.71669649151451-0.24\log p})/3 \\
x &= (10^{0.20087391135-0.78\log p} + 10^{-0.951773824150120.71\log p} + 10^{0.71669649151451-0.24\log p})10^{-\log 3} \\
x &= 10^{0.20087391135-0.78\log p-\log 3} + 10^{-0.951773824150120.71\log p-\log 3} + 10^{0.71669649151451-0.24\log p-\log 3}
\end{aligned} \tag{B.6}$$

$$\therefore x = 10^{-0.27624734337-0.78\log p} + 10^{-1.4288950788699-0.71\log p} + 10^{0.239575236794848-0.24\log p}.$$

We then determined a simplified curve approximating the above demand curve:

$$x = 10^{0.37(1-\log p)}. \tag{B.7}$$

To determine the slope at a given point we must differentiate $x(p)$. To do so, we first simplify $x(p)$ for differentiating:

$$x(p) = 10^{0.37(1-\log(p))} = 10^{0.37}10^{-0.37\log(p)} = 10^{0.37}(10^{\log(p)})^{-0.37} = 10^{0.37}p^{-0.37}. \tag{B.8}$$

Next, we differentiate $x(p)$:

$$x'(p) = -0.37\times 10^{0.37}p^{-1.37}. \tag{B.9}$$

Appendix C: Knapsack Problems

Knapsack problems are combinatorial problems dealing with optimization under various constraints, as will be seen in the following subsections.

Classical 0-1 Knapsack Problem

In the classical, or simple, 0-1 knapsack problem, there exists a conceptual knapsack and n items. Each item has a value, $v_i \in \mathbf{R}^+$, and a specified weight, $c_i \in \mathbf{R}, c_i \geq 0$, where $i \in \mathbf{N}, 1 \leq i \leq n$. The objective is to select items to be put into the knapsack in order to maximize the value of the items within the knapsack, V, while respecting the weight constraint of the knapsack, C. Figure C.1 shows an example knapsack problem with four items of varying size.

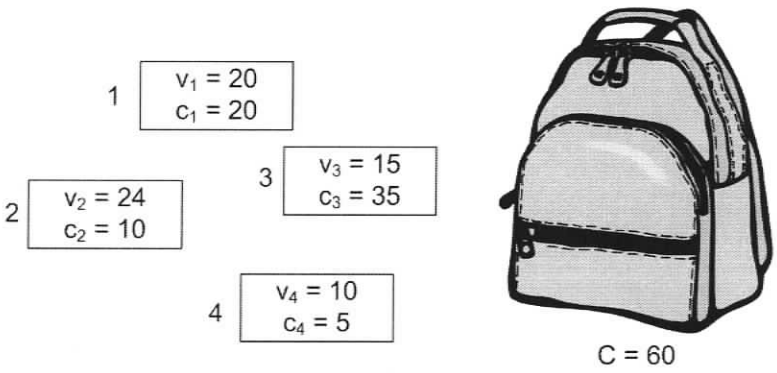


Figure C.1: 0-1 Knapsack problem with 4 items.

Let x_i be a binary-valued variable representing the decision of whether or not an item should be placed within the knapsack (0 indicates that the item is not added, while 1 indicates that the item is added), then the problem can be stated concisely as:

$$V = MAX\left(\sum_{i=1}^{n} x_i v_i\right), \text{ such that } \sum_{i=1}^{n} x_i c_i \leq C, \text{ where } x_i \in \{0,1\}. \qquad \text{(C.1)}$$

Multi-dimensional Knapsack Problem

The 0-1 multi-dimensional knapsack problem (MKP) is similar to the classical 0-1 knapsack problem, except that there are multiple resource constraints, instead of a single resource constraint. Now a knapsack can have numerous resource constraints, where items can consume some of each resource. Figure C.2 shows an example MKP with two resource constraints: C_1 and C_2.

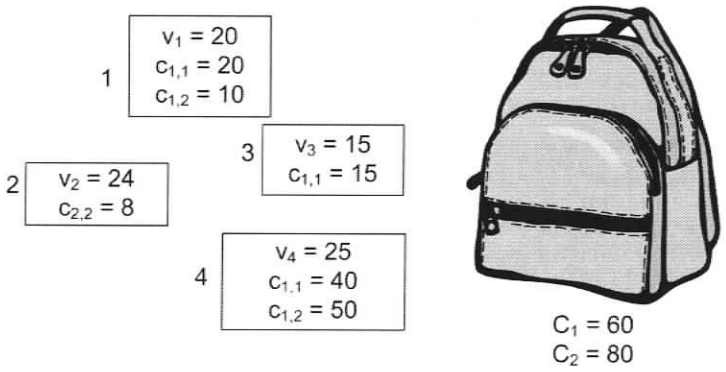


Figure C.2: Multidimensional knapsack problem with 4 items and 2 resources.

Let n be the number of items, c_{ir} be the amount of resource r item i will consume, and C_r be the total amount of resource r that the knapsack contains, then

$$V = MAX\left(\sum_{i=1}^{n} x_i v_i\right), \text{ such that } \forall r, \sum_{i=1}^{n} x_i c_{ir} \leq C_r, \text{ where } x_i \in \{0,1\}. \qquad \text{(C.2)}$$

Multiple-Choice Multi-Dimensional Knapsack Problem

The multiple-choice multidimensional knapsack problem (MMKP) is a generalization of the MKP in which items are clustered into groups, of which one item, and only one item, from each group must be added to the knapsack. For example, Figure C.3 shows three groups of items of which an item in each group has been selected to be placed into the knapsack.

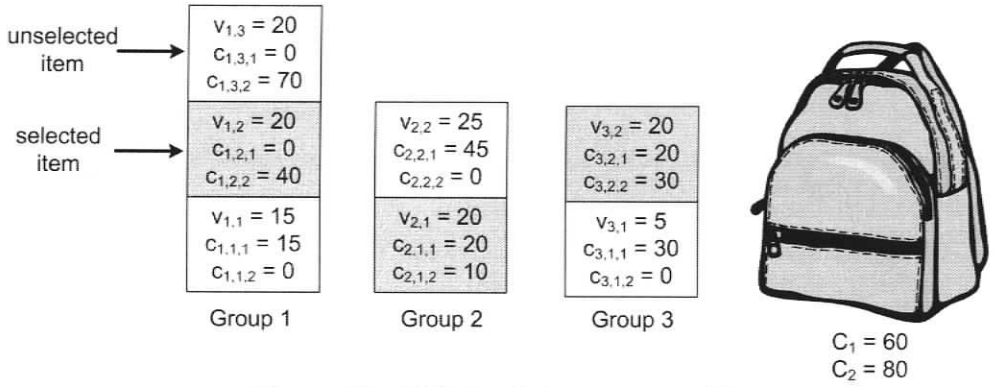


Figure C.3: MMKP with three groups of items.

Let there be n groups and l_i items in group i, where $1 \le i \le n$. Now let x_{ij} be the decision of whether or not item j of group i was picked. The MMKP can now be expressed as follows:

$$V = MAX\left(\sum_{i=1}^{n}\sum_{j=1}^{l_i} x_{ij}v_{ij}\right), \text{ such that } \forall r, \sum_{i=1}^{n}\sum_{j=1}^{l_i} x_{ij}c_{ijr} \le C_r, \tag{C.3}$$

where $x_{ij} \in \{0,1\}$ and $\forall i, \sum_{j=1}^{l_i} x_{ij} = 1$.