Pricing Network Resources in Differentiated Service Networks

A Thesis
Presented to
The Academic Faculty

by

Weilai Yang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in School of Electrical and Computer Engineering

School of Electrical and Computer Engineering
Georgia Institute of Technology
April 2004

Pricing Network Resources in Differentiated Service Networks

Approved by:

Doug Blough and Henry Owen, Adviser

Professor John Copeland

Professor George Riley

Date Approved: 8 April 2004

To my fiance Toby, my parents, brother and sister-in-law.

ACKNOWLEDGEMENTS

I would like to thank my advisors Prof. Doug Blough and Prof. Henry Owen for giving me the opportunity to pursue this work and for their teaching and research guidance throughout my study in Georgia Tech. Their vision and experience, going beyond traditional boundaries, were essential. I specially thank Prof. Dough Blough for his inspiration and support both professionally and personally.

I would also like to thank Prof. John Copeland, Prof. George Riley and Prof. Chin Lee for serving on my thesis proposal committees; as well as the defense committees. Thanks for their precious time and great advice on my research work.

I am very grateful to my peers, Rupa Para, Alessandro Brawsmen, Oliver and Dishi Sun. They provide comfort at hard times and fun at good times.

Above all, I would like to thank my fiance Toby, my parents, my brother and sister-in-law. I thank for their support, encouragement and unbounded love! Without them, I won't be able to complete this thesis.

TABLE OF CONTENTS

DEDICATION . . . . **iii**

ACKNOWLEDGEMENTS . . . . **iv**

LIST OF TABLES . . . . **viii**

LIST OF FIGURES . . . . **ix**

CHAPTER I ABSTRACT **1**

CHAPTER II ORIGIN AND HISTORY OF THE PROBLEM . . . **3**

2.1 Introduction 3

2.1.1 Quality of Service 3

2.1.2 IntServ 4

2.1.3 DiffServ 5

2.1.4 Pricing 8

CHAPTER III LITERATURE SURVEY **15**

3.1 General Survey 15

3.2 Utility and Fairness 19

3.2.1 Utility 19

3.3 Fairness 20

3.4 Summary of previous algorithm limitations and our new contributions 23

CHAPTER IV OPTIMAL SOLUTIONS FOR A SINGLE BOTTLE-NECK LINK NETWORK **26**

4.1 The Case with Base Price Fixed 26

4.1.1 Problem Formulation 27

4.1.2 Optimal Solution 28

4.1.3 Simulation and Analysis 33

4.2 Introducing the Base Price as a Variable 37

4.2.1 Problem Formulation 37

4.2.2 Optimal Solution 39

4.2.3 Simulation and Analysis 41

4.3 Problems Remaining 46

4.4 Summary 47

CHAPTER V HEURISTIC ALGORITHMS FOR MULTIPLE LINK NETWORKS 48

5.1 Problem Formulation in a Multiple Bottleneck Link Network 48

5.2 Intra-link and Inter-link Solution (IIS) 51

5.2.1 Intra-link auction 51

5.2.2 Inter-link Adjustments 53

5.3 Simulation and Analysis 55

5.3.1 Simulation Design 55

5.3.2 Simulation Results 56

5.3.3 Analysis of Results 63

5.4 One Step Solution (OSS) 68

5.5 Simulations and Analysis 70

5.6 Summary 72

CHAPTER VI AUCTIONS WITH DURATION PARAMETER INVOLVED 74

6.1 Statistics of how the model behaves 74

6.2 What happens when using IIS algorithm with duration added 77

6.2.1 Simulation Set Up 77

6.2.2 Simulation Results 78

6.2.3 Analysis 81

6.3 Problems and Possible Solutions 82

6.3.1 Problems 82

6.3.2 Prediction Function 84

6.3.3 Depreciation Function 85

6.3.4 Procedure 86

6.4 Summary 87

CHAPTER VII CONCLUSION AND FUTURE WORK **90**

7.1 Future Work 92

LIST OF TABLES

Table 1 The assumptions that are used in the simulations 42

LIST OF FIGURES

Figure 1 DiffServ Infrastructure [16] 6

Figure 2 Flows Share Links 8

Figure 3 Pricing models classification [45] 11

Figure 4 Pricing Mechanisms classification [50] 12

Figure 5 An Example of a Simple Network Sharing 21

Figure 6 One Case of Resource Assignment - maximize total flow 21

Figure 7 One Case of Resource Assignment - max-min fairness 22

Figure 8 One Case of Resource Assignment - proportional fairness 22

Figure 9 Single Bottleneck link network model 24

Figure 10 Optimization Algorithm for Single Class 31

Figure 11 Optimization Algorithm for Multiple Classes 34

Figure 12 Comparison Results 36

Figure 13 The revenue comparison (customers' mean valuation is 2.7 and 2.0 mcents) 43

Figure 14 The revenue comparison (customers' mean valuation is 4.0 and 3.0 mcents) 43

Figure 15 The revenue comparison (customers' mean valuation is 5.4 and 4.0 mcents) 44

Figure 16 A Single Link Network Example 46

Figure 17 The Revenue Comparison Between Flat Rate Pricing and Auction (with fixed flat rate and 20 nodes 200-400 customers) 57

Figure 18 The Revenue Comparison Between Flat Rate Pricing and Auction (with fixed flat rate and 200 nodes 400-1000 customers) 57

Figure 19 The Revenue Comparison Between Flat Rate Pricing and Auctions (with fixed flat rate and 1000 nodes 3500-5000 customers) 58

Figure 20 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 20 nodes) 58

Figure 21 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 200 nodes) 59

Figure 22 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 1000 nodes) 59

Figure 23 Revenue Comparison for Multiple-Link Networks (as a function of offered load) 60

Figure 24 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 20 nodes) 60

Figure 25 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 200 nodes) 61

Figure 26 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 1000 nodes) 61

Figure 27 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 20 nodes) 62

Figure 28 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 200 nodes) 62

Figure 29 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 1000 nodes) 63

Figure 30 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 30%EF and 70%AF demands) 63

Figure 31 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 50%EF and 50%AF demands) 64

Figure 32 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 70%EF and 30%AF demands) 64

Figure 33 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 50%EF and 50%AF demands) 65

Figure 34 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 50%EF and 50%AF demands) 65

Figure 35 Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 70%EF and 30%AF demands) 66

Figure 36 Revenue Comparison between OSS and Flat Rate Pricing 71

Figure 37 Revenue Comparison between OSS and IIS 71

Figure 38 Execution Time Comparison between OSS and IIS 72

Figure 39 The Interrupted Percentages for Each Auction 76

Figure 40 Revenue Comparison 79

Figure 41 Revenue Increasing Percentage 80

Figure 42 Revenue Comparison 81
Figure 43 Revenue Comparison between Auctions and Flat Rate Pricing at the End of Two Hours 82
Figure 44 Revenue Increasing Percentage for Auctions at the End of Two Hours 83
Figure 45 Revenue Comparison between Auctions and Flat Rate Pricing at the End of Two Hours 84
Figure 46 Revenue Increasing Percentage for Auctions at the End of Two Hours 85
Figure 47 Revenue Comparison between Auctions and Flat Rate Pricing at the End of Ten Hours 86
Figure 48 Revenue Increasing Percentage for Auctions at the End of Ten Hours 87
Figure 49 An Example to Show the Duration Factor 87
Figure 50 Continuous auction procedure 88

CHAPTER I

ABSTRACT

The objective of this thesis is to provide a pricing mechanism to allocate Quality of Service (QoS) capable resources while maximize a service provider's revenue. The service provider (SP) provides network resources at various levels of QoS to multiple users. A Service Level Agreement (SLA) between a client and an service provider consists of a service guarantee offered by the service provider and a promised payment by the client.

We develop a price-based resource allocation scheme for Differentiated Service (DiffServ) data networks. The DiffServ framework was proposed to provide multiple QoS classes over IP networks. Since the provider supports multiple service classes, we need a differentiated pricing scheme, as supposed to the flat-rate scheme employed by the Internet service providers of today. Charging efficiently is a big issue. By "efficient", we mean maximizing the utilities of both clients and the provider. The utility of a client correlates to the amount of bandwidth allocated. One difficulty we face is that determining the appropriate amount of bandwidth to provision and allocate is problematic due to different time scales, multiple QoS classes and the unpredictable nature of users.

To approach this problem, we designed a pricing strategy for Admission Control and bandwidth assignment. Pricing has been shown to be an effective mechanism for achieving fair allocations and good revenue generation. Despite the variety of existing strategies, the common theme is that the appropriate pricing policy rewards users for behaving in ways to improve the overall utilization and performance of the network.

Among existing schemes, we chose auction because it is scalable, and efficiently and fairly shares resources. Our pricing model takes the system's availability and each customer's requirements as inputs and outputs the set of clients who are admitted into the network and their allocated resource. Each client proposes a desired bandwidth and a price that they are willing to pay for it. The service provider collects this information and produces parameters for each class of service they provide. This information is used to decide which customers to admit. Our scheme is fair in the sense that it takes the benefits of both sides into account.

We propose an optimal solution to the problem of maximizing the provider's revenue for the special case where there is only one bottleneck link in the network. Then for the generalized network, we resort to a simple but effective heuristic method. We validate both the optimal solution and the heuristic algorithm with simulations driven by a real traffic scenario. Finally, we allow customers to bid on the duration for which the service is needed. Then we study the performance of those heuristic algorithms in this new setting and propose possible improvements.

CHAPTER II

ORIGIN AND HISTORY OF THE PROBLEM

The Internet can sustain a great range of new applications that were considered science fiction only a few years ago. Today, real time audio and video, digitized voice and fax can be transmitted over the Internet. With the growing usage of the Internet, many issues surface regarding utilizing resources effectively. Two major architectures were developed to support real-time services: Integrated Service (IntServ) and Differentiated Service (DiffServ). On top of these frameworks further management and negotiation between providers and customers are needed. Pricing is an effective methodology for providing this interaction in both IntServ and DiffServ networks. We shall summarize the contributions and limitations of the existing pricing schemes in the following sections.

2.1 Introduction

First, we introduce the concepts QoS, IntServ, DiffServ and Pricing.

2.1.1 Quality of Service

QoS (Quality of Service) is the idea that transmission rates, error rates, and other characteristics can be measured, improved, and, to some extent, guaranteed in advance. It is characterized by a service level agreement (SLA) [44] [47]. Quality of Service (QoS) involves a set of quality parameters for the transmission of data over communication networks. The set of parameters are specified according to the communication layer, but they are mainly derived from the end user's expectations and service level agreements. The failure of delivering certain QoS can have serious consequence. For example, a business website offers an online order catalog. If the site cannot provide the service with certain reliability, latency and security, it could result in the loss of thousands or even millions of dollars of sales. Being able to provide QoS dynamically can increase revenue by reducing customer inconvenience. At the network layer, QoS parameters usually define performance requirements in terms of throughput, reliability, latency, jitter, synchronization, loss packet rate and so on. Users' demands for performance, reliability and availability are increasing rapidly. To meet these requirements, a high level of QoS must be delivered.

To date, two architectures have been proposed by the Internet Engineering Task Force (IETF) to implement QoS in IP networks: Integrated Services (IntServ) with the use of RSVP and Differentiated Services (DiffServ) architecture. RSVP was proposed by the IETF working group as a method for allocating capacity at each router in a network flow's path. Within this framework, end-to-end QoS guarantees can be made in conjunction with admission control. However, RSVP is very resource intensive in terms of signaling control as well as flow based scheduling. To overcome these shortcomings, IETF proposed DiffServ, which provides reliable QoS in a scalable fashion. Within this mechanism, flows are aggregated into a fixed set of service classes. In the next sections, we review IntServ and the problems it introduces, and then the motivations for moving to DiffServ.

2.1.2 IntServ

In 1993, IETF started to develop a mechanism that would allow IP to support more than a single best-effort class of service. The goal was to provide real-time service simultaneously with traditional non-real-time services in a shared IP network. This work lead to the development of the Integrated Services (IntServ) architecture. IntServ is based on per-flow resource reservation requests from end users [52] [53].

The requests dictate the service level, i.e. bandwidth and buffer space that must be reserved along the flow path. Each router must apply admission control to the requests to ensure that they are accepted only if sufficient local resources are available.

RSVP is as one of the key components in the Integrated Services architecture. RSVP identifies a communication session by the combination of source and destination address, transport layer protocol type, and the source and destination port number. It is important to note that RSVP is not a routing protocol; it is merely used to reserve resources along the existing route computed by the underlying routing protocol. As mentioned, IntServ has some shortcomings, we list them here:

- Each router is required to maintain state information, the amount of state information increases proportionally with the number of flows. This means that the architecture does not scale well in the core network.

- The requirement on the routers is high. All participating routers must implement RSVP, admission control, classification and packet scheduling in order to obtain end-to-end QoS. It is unrealistic for all routers to enable RSVP in the current Internet infrastructure.

These problems lead to DiffServ.

2.1.3 DiffServ

From 1995-1996, researchers examined alternative approaches of supporting multiple best-effort classes of service. IntServ's inscalability attributes to the amount of per-flow state that needs to be maintained at each node in the packet-forwarding path. DiffServ takes a different strategy to provide QoS to the traffic flows. Instead of using a signaling protocol to reserve resource, DiffServ allows IP packets to be classified into a finite number of service classes. Based on a packet's service class, it receives different treatments from the routers. DiffServ does not guarantee an exact end-to-end QoS, but it does provide that a packet of a higher priority gets through with a higher probability than that of a lower priority class [5].

The complete DiffServ architecture, defined in RFC 2475, is based on a relatively simple model as shown in Figure 1. Packets are classified and then are treated accordingly. At the edges of the network, incoming packets are classified and marked into service classes according to pre-defined criteria. The Type of Service (TOS) field of the IPv4 packet headers or the traffic class field of the IPv6 (defined as DS field) is used to encode the service information (defined as DS Codepoint, DSCP) [38] [39]. When a packet enters the core of the network, each router along the transit path applies the appropriate per-hop behavior (PHB), based on DSCP carried in the packet's header. Because the packets contain information needed by buffer management and scheduling, differentiated services do not require signaling protocols in order to control mechanisms used for QoS differentiation.

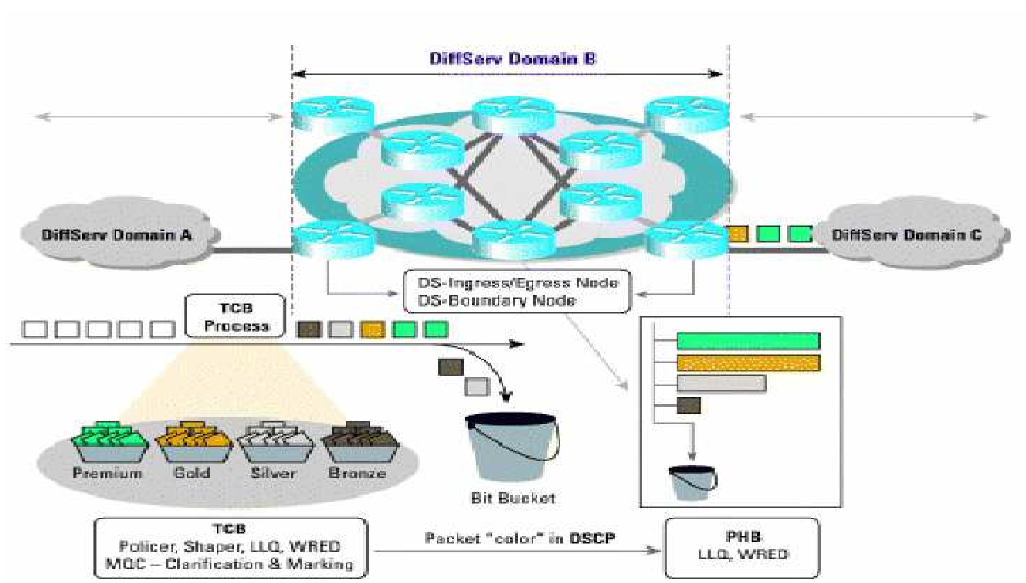


Figure 1: DiffServ Infrastructure [16]

A PHB is a description of the externally observable forwarding behavior of a differentiated services node. Each PHB is applied to a collection of packets with the same DSCP that are crossing a link in a particular direction. The DiffServ architecture supports the discrimination of packets scalably, based on this hop-by-hop resource allocation mechanism. Three PHBs have been defined.

1. Expedited Forwarding (EF) PHB is a high priority behavior typically used for network control traffic. It is ideal for supporting constant bit rate services because it maximizes the resources allotted to conforming flows by throttling the supply of bandwidth to non-conforming flows. In the process, it guarantees zero packet loss for conforming flows and 100% packet loss for non conforming flows.

2. Assured Forwarding (AF) PHB is a group of PHBs offering different levels of forwarding assurances. Four AF classes have been defined. For each AF class, packets of the class are marked with one of three possible drop precedence values and the classes is assigned to a unique flow queue with its own queue weight. Packet ordering is preserved for flows of the same class. During the scheduling process, conforming flows are usually assigned the lowest drop preference and non-conforming flows are assigned a higher drop preference.

3. Best Effort (BE) Forwarding is the default PHB for best effort traffic. This PHB does not police flows, nor does it provide guaranteed packet delivery. It merely insures that packets sent to this flow queue are not starved for bandwidth and that they receive any share of bandwidth unused by other queues.

The DiffServ domain is a class-based alternate which achieves improved scalability by aggregating data packets into a small number of service classes. It also defines router behaviors expected by packets belonging to each of these classes. Given this, next, we need a uniformed pricing model for each class.

2.1.4 Pricing

Consider a simple bandwidth contention scenario. There are four flows in the network as shown in Figure 2. Flows 1 and 2 share link 1 , flows 3 and 4 share link 2 and flows 1, 2, and 3 share link 3. Even if flow 3 knows link 2 and 3's total demand, it does not have any knowledge of the situation on link 1, which could potentially affect the bandwidth allocation of link 3. In general the users cannot be aware of the resource availability in all of the network. Thus we need a more automated network management. Prices play the role of resource allocation control signals. Whether the network is congested or not, the role is important in best utilizing the system's resources and/or maximizing the throughput from either customer's or service provider's side. In commercial networks, prices play an additional role other than controlling congestion, which is signalling the need for and financing the development of infrastructure [20].

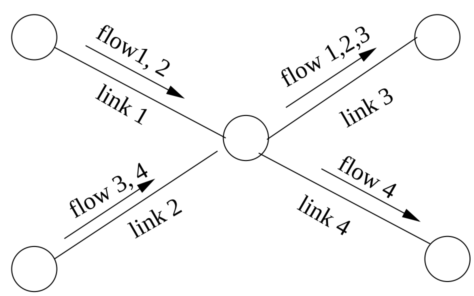


Figure 2: Flows Share Links

The telephone system and Internet represent two extremes of the trade off between utilizing resource allocation and pricing. The bandwidth allocated to a phone call is fixed and the unit price is decided by whether it is peak time or off peak time of the day, i.e. the prediction of the total demand the bandwidth. In the Internet, it's difficult to make uniform every connection's bandwidth. The predominant form of Internet retail pricing used currently is flat-rate pricing. A fee is charged for a set amount of bandwidth to access the network. The fee doesn't vary with the actual usage of the network resource or congestion situation. It decouples the allocation of resources from prices. The inevitable consequence of flat-rate scheme is the emergence of congestion externality. This nature of flat-rate pricing makes it difficult for clients to use the system's resources efficiently. Because there are no incentives for customers to limit their usages.

TCP protocol uses congestion control mechanisms. But with the growth of multimedia traffic in the networks, TCP congestion control behavior does not scale well. Under these circumstances, flat rate pricing can cause increased congestion or increased price [19]. If congestion increases, it makes more customers dissatisfied, which chases away high-value clients. On the other hand, increasing rates causes low-value clients to drop out. We can partition the network into low quality and high quality subnetworks, so that both kinds of customers can coexist in the same network. However, the partition restricts the customers from dynamically sharing all the resources and creates chances of wasting bandwidth.

Using pricing to achieve economic efficiency in computer networks has recently attracted significant attention. A number of pricing schemes were proposed [25] [31] [33] [54] [21] [12] for managing resource allocation and network congestion. In these schemes, the main idea is that the appropriate pricing policy will provide incentives for users to behave in ways that improves the overall utilization and performance.

In the survey paper [50], Stiller defined four basic pricing dimensions in terms of the classification of pricing models. Pricing models can be classified into static and dynamic. Static pricing does not change prices with respect to how the network behaves over time. An advantage of this is that the network doesn't need to manage price changes. Another is that users do not need to worry about the price changes. Flat-pricing is limited in that the network resources cannot be efficiently used in good optimality because the network treats packets identically even if they have very different values to end-users. The alternative is dynamic pricing. In this model, the frequency at which prices change is related to the predictability of the the network. The pricing model is classified according to the time scales at which they change. Long-term scale approaches update prices every month or year. This time scale coincides with the time scales of network upgrades, and makes sense for the service provider's network administration. This approach limits the cost of network over-provisioning, but it has the disadvantage that it does not consider the fluctuations in network traffic. This means that resource allocation and admission control policies have to be conservative in order to meet QoS assurances in the presence of network traffic dynamics. Short-term strategies address those issues. In these approaches, prices change in real time as transient network overloads occur. Pricing of network services can change dynamically based on the level of QoS, actual usage and congestion situation. This enables the network to be used more efficiently. Users can respond to price changes by updating their bidding price. The system communicates congestion costs to the users at fine grain in real time. The drawbacks of this approach are:

- increased computation overhead and therefore longer delays;
- clients may not be able to respond fast enough to all the changes;
- users do not know the final cost until after they have used the service

Reichl [45] categorized pricing models into three dimensions from another point of view, shown in Figure 3. They are the research, technical and economical & social dimensions. The categorization is based on service categories, charging parameters, tariff components, efficiency and research dimension [50]. The service category includes the type of traffic, as Best Effort, IntServ and DiffServ. The charging parameters are the parameters that are used to measure the performance. Common parameters are effective bandwidth, congestion cost, delay, latency *etc.* Tariff components include set up cost, access charges and usage charges. The requirements of tariff models for internet services are:

$$Tariff\ Models\ Requirement \Rightarrow \begin{cases} Efficiency \\ Feasibility \\ Fairness \\ Transparent\ predictability \\ Signalling \\ Edge\ Pricing\ paradigm \\ Dynamics \end{cases}$$

Combining the three basic tariff elements of set up cost, access charges and usage charges leads to a classification of pricing mechanisms, as shown in Figure 4 [50].

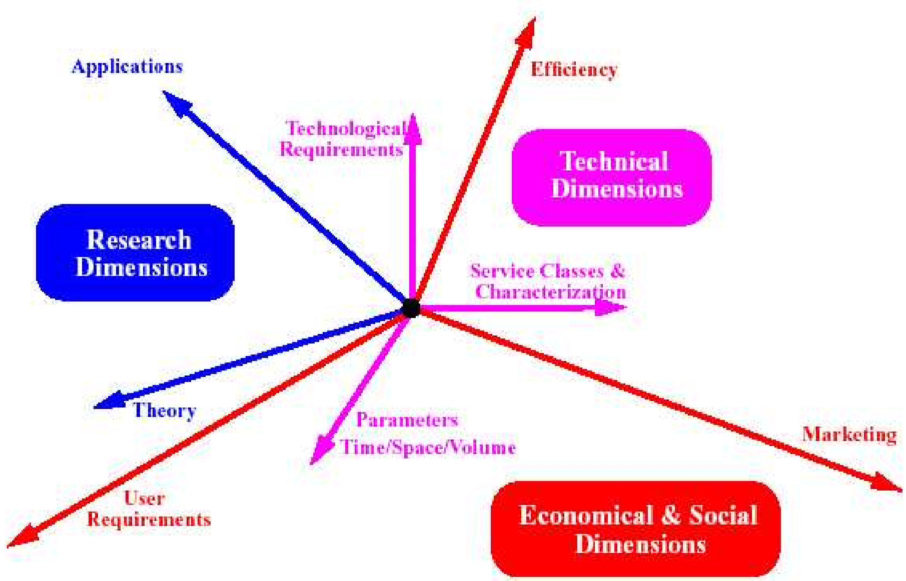


Figure 3: Pricing models classification [45]

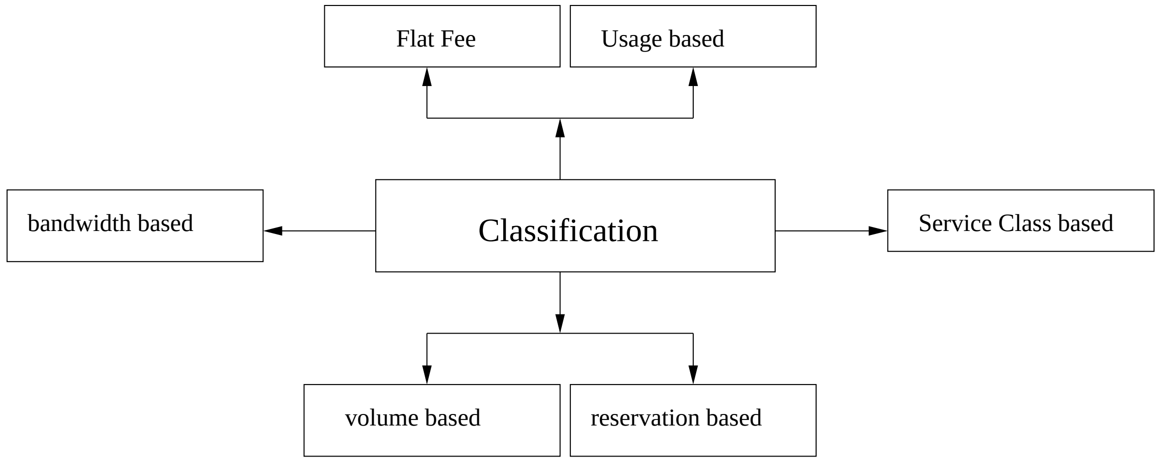


Figure 4: Pricing Mechanisms classification [50]

Pricing is also classified into edge pricing, usage based pricing, metered pricing and congestion based pricing. Edge pricing charges for usage at the edge of the network scope instead of along the routing path from source to destination [15]. Usage based pricing charges customers based on the amount of resource that they consume. Metered pricing usually measures the usage in units of time and per unit time charge is fixed or agreed on between clients and service providers. In congestion based pricing, an extra charge beyond base prices can be triggered by the network congestion status. It can efficiently control the network utilization while demanding exceeds supply. However, it doesn't give a usage for non-congestion case.

Our aim is to use pricing to achieve the best resource assignment in order to maximize the service provider's revenue. Generally speaking, most approaches use a bandwidth broker (BB) to supervise the network in a centralized manner. With this method, it is easier to control resources of the entire network, but it has an inherent scalability problem. Auction is an alternative which is decentralized, efficient and fair.

In this thesis, we study an auction based pricing method. An auction is a pricing approach with minimal a priori information needed [48]. It consists of submitted bids by clients who specify the desired amount of resources and the price they are willing to pay, and an auction step which allocates the resources to the clients. It doesn't need a *priori* information to calculate the optimum prices and the corresponding resource allocation. Auctions have been defined as a "market institution with an explicit set of rules determining resource allocation and prices on the basis of bids from the market participants" [34]. An auction is called "clear" when it commands an allocation based on the bids it has received. In Internet usage, no one person knows the true value and each individual's estimate may be highly imperfect, but the clearing price is still an accurate value estimate. The competitive process serves to consolidate the scattered information about bidders' valuations [7]. Quite a few auction schemes have been widely used.

- *English auctions*: allow renegotiation. The buyers and sellers can update their bids and sale prices. The highest bidder wins finally.

- *Dutch auctions*: the original starting price starts from a high level and declines until one buyer agrees to pay at that price.

- *Vickrey auctions*: the winner pays only the amount of the second-highest bid. It's a fair mechanism because neither the buyer nor the seller has an incentive to lie.

Most classic auction theory restricts attention to the sale of a single indivisible unit and a single winner. Auctions which allocate multiple units of a good and multiple winners have received increasing attention in the theoretical literature. Combinatorial auctions are an approach for achieving efficient allocations when bidders place bids on combinations of different goods.

In Internet usage, auctions also provide a natural and equitable incentive for customers to adapt their resource utilization to the supply and demand of the network.

Compared to fixed pricing, auctions generate more revenue for the service provider by favoring higher priced traffic classes while dropping some traffic from lower-priced classes.

CHAPTER III

LITERATURE SURVEY

3.1 General Survey

In the past few years, several economic models for resource management have been proposed by researchers. Some of them divide traffic into multiple priority classes and use fixed prices for each service class [12]. A dynamic pricing strategy, by contrast, adds congestion-dependent components into the pricing mechanism. Dynamic pricing strategies improve network efficiency and offer more competitive prices by taking network activities into account. Congestion-dependent prices are determined on a per-call basis, at the time when the client is admitted. Mason and Varian [33] introduced the idea of a “smart market”. “Smart market” is an efficient mechanism for congestion control that is usage-sensitive. Users are required to bid on each packet at each router. A packet is transmitted if the bid from the corresponding customer exceeds the current marginal cost of transportation. An advantage of this scheme is that the user usually does not pay the price that he bids, but the lower market-clearing price, i. e. the maximum bid of the users that were rejected. The bidding process takes place in a hop-by-hop manner to be determined at each hop. This requirement is unrealistic in practice because in general, customers are only concerned and would only be informed about end-to-end behavior. It is infeasible for customers to track the route that their messages are taking in being delivered. In order to be able to make an end-to-end guarantee, auctions at all hops of the path must be performed at once. This makes the system overwhelmingly complex. In addition, in the real world, the computational burden of updating prices dynamically can be quite high.

Recently there were a number of works within the IntServ framework [18] [33] [51] [45]. Reichl *et al.* [45] are trying to provide a mechanism for multiple service providers. They proposed to perform auctions continuously. The auction technique provisions new flows coming and old flows dropping to judge new rounds of bidding. A major problem with this approach is that even after a user has reserved resources with RSVP, there is a probability of him being dropped later. In addition, the system puts too much complexity into the routers because it uses the RSVP mechanism. Moreover, it is also problematic for use in a large-scale network since the recalculation and confirmation cycle would eat up all the resources in the IntServ framework.

Thomas et al. [51] assumed that the relationship between QoS and resource allocation were given, which constrained the problem to a static optimization problem. The objective of the optimization problem is to determine the amount of required resources for each type of service while maximizing the sum of the users' utilities. They proved the existence of a solution to the optimization problem. The downside of the method is that when a new user joins in, the optimization function must be recomputed to generate the new assignment and bandwidth allocation. This property makes it unscalable. The limitations of the above mentioned methods all come from the inherent constraints of IntServ networks.

After researchers realized the infeasibility IntServ, the design of pricing schemes also moved to the priority based or DiffServ domain. Marbach [32] defined a process in which clients are free to choose a desired priority. Here, different priorities represent different classes in the DiffServ domain. Customers cannot specify or know in advance the price, but are billed by the network as it sees fit. The users' service requirements are characterized by a utility function which depends on the throughput. A packet is treated based on its proposed priority. A user can have high or low priority packets. A problem with this scheme is that since priority queue scheduling is used, low priority packets may starve. The exact level of service for a given priority depends on the number of users using it, therefore, without knowing the global traffic situation in the network, customers cannot specify the exact priority level. Under such circumstances, it is beneficial to have feedback from the network, so customers can adjust to a priority which fits their needs better. Customers are allowed to specify different priorities for their applications and each priority corresponds to a transmission probability. The customers belonging to one priority comply with a probability that the application will or will not be transmitted. The mechanism allows users to determine transmission probabilities under the current aggregate allocation, but the feedback loop introduces a delay to the network and make it impractical.

Renegotiation between the service provider and the customers is a compromise between using feedback and reducing complexity. Wang [55] proposes a strategy in which price depends on the service class' average demand. The price is negotiable through a negotiation protocol. The strategy improves the performance from a stand alone feedback mechanism. However, the downside is that it requires resource reservation in the network, which leads to inefficient usage of the network resource and increases network cost.

A more scalable solution is to compute prices for customers statically but yet which is able to admit new flows without recalculating. In this way, real time traffic status would be reflected while dynamic computation complexity reduced greatly, making it feasible for being implemented in the Internet. For example, Basal [4] assumed the price per unit bandwidth was fixed for each user and that the transmission rate for each user was a function of network congestion and price per unit of bandwidth. They showed that as the number of users increases the optimal price per unit bandwidth increased as well. The pricing model does not have a base price, which is incompatible with the current Internet service model. The utility function was U= W_i $logX_i$. In this paper, W_i is the admission criteria (X_i is the allocated resources). The users with smaller W_i's are dropped out of the network. This strategy keeps admission simple but effective.

Despite the differences of approaches and objectives of solving pricing problems, a pricing model has to be designed based on network resource availability. Representing the whole network's availability and properly using it is a critical issue. Most of the research work so far deals with networks in which there is only one bottleneck link [3] [36] [41] [58], as shown in Figure 9. This is a simplified model and optimal solutions have been found. However, in a real network, there maybe multiple bottleneck links and a flow should be able to choose different routes to optimize its benefit [1].

The trade-off between engineering and economic efficiency distinguishes the pricing scheme as *model based* or *market based* [37]. Nemo states the trade-off dimensions include the following: 1)the measurement from usage to capacity pricing; 2)the granularity of differently priced service offerings; 3)the level of resource aggregation, both in time and space; and 4)the information requirement – amount of *a priori* knowledge of user behavior and preferences is required by the network in computing prices [37].

In a *model based* pricing scheme, the relationship between demand and price is assumed to follow an *a prior* relationship. The service providers adjust the *optimal* price dynamically, as shown in [2] [21] [28] [43]. Kelly designed a model where clients are charged according to a combination of declared and measured characteristics of traffic [26]. The model offers a menu of pricing plans indexed by the declared traffic to the clients. The menu takes an equivalent bandwidth model of resource utilization and assumes appropriate traffic models, which encourages users to make truthful declarations. Even though the menu does reduce the *a priori* information required, it is still not very practical. The main reason is that customers do not have a good

[1]This falls into routing optimization research areas, which is beyond our scope. We assume that routing information is given together with customers' bids.

knowledge of the price they will end up paying.

[23] summarizes pricing schemes that incorporate multiplexing gain. The summary stands at a comprehensive point of view to look at the connection establishment process between users and networks. It identifies the information required to determine the proper prices beforehand through a user-network negotiation. However, there is one point missing that without a formal mechanism to deal with the *a prior* information, complex, unpredictable and undesirable situations may occur.

The best way to avoid requiring or minimizing the *a prior* information lies in market based approaches. Since no precise model needs to be assumed, different prices may arise from the same given demand. The smart-market approach [33] falls into this category.

3.2 Utility and Fairness

3.2.1 Utility

In general, utility is a monetary evaluation of the benefit derived from a particular level of service. In cases with a real time requirement, it is useful and a common practice to consider the utility as a function of consumed bandwidth. How the utility function is defined is important.

There is quite a bit of work on maximizing service provider's revenue or user utility, using either static pricing method or dynamic computation or auction. Most of the previous work suffers from several limitations which prevent them from being used in a real network scenario:

- In order to get a nice formulation of the problem and be able to solve it analytically, researchers like to assume the function has following characteristics: continuous and strictly decreasing/increasing, differential and twice continuous differentiable [42]. Based on that, either an optimal point or a Nash Equilibrium (in game theory) is easy to locate. There are two disadvantages to these methods. First, the mathematical model is not intuitive, this means users will have to experiment with the system and become experts about it. Second, the assumptions do not provide a clear image of how the calculations are carried out.

- The previous work assumes that revenue is strictly proportional to the bandwidth allocated [21] [55]. The assumption ensures that the function is continuous and differentiable and converges quickly when used as the objective function in optimization. This is valid for some particular applications such as data transfer and email. But it does not hold for applications such as video streaming. Realtime applications need to sustain a minimum real-time network resource when running. When the assigned resource is below the required minimum, the whole application fails and the customer receives no utility. However if proportional calculation is adopted, even though the customer receives nothing, he still has to pay.

We are inspired by the function developed by Basar [4]. The utility function that they use is U=$w_i \log x_i$ (w_i is a sensitivity coefficient and x_i is the transmission rate of user i). They use w_i as the admission criteria. The users with smaller w_i's are dropped out of the network. This strategy keeps admission simple but effective. We will present in detail our revenue function in Chapter 4.

3.3 Fairness

There are various definitions of fairness. Consider a simple network topology as in figure 5. There are 3 flows in the network. Flow 1 crosses link 1 and 2; flow 2 crosses link 1 and flow 3 crosses link 2. Each link has a capacity of 6 MBPS. To make the example simple, we assume that flow 1,2 and 3 have concave utility functions. There are a few ways to assign the resource to these flows in the interest of different *fairness*.

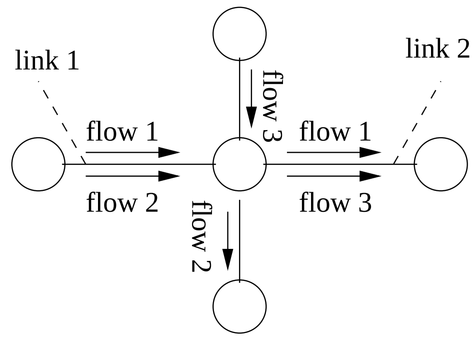


Figure 5: An Example of a Simple Network Sharing

- Maximize the total flow

  Assigning all the capacity to flow 2 and 3 reaches the goal of maximizing the total flow. The assignment figure is shown in figure 6.

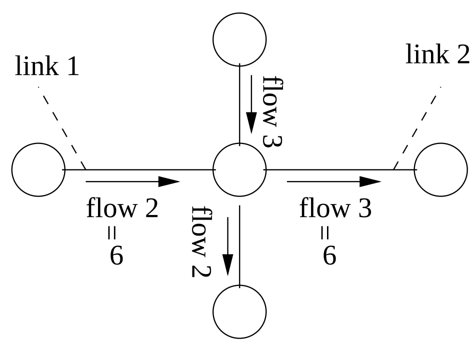


Figure 6: One Case of Resource Assignment - maximize total flow

- Max-min fairness

  An assignment is max-min fair if no flow's utility can increase without decreasing others' utility. In this case, if each flow gets 3 MBPS, it reaches max-min fairness.

- Proportional fairness

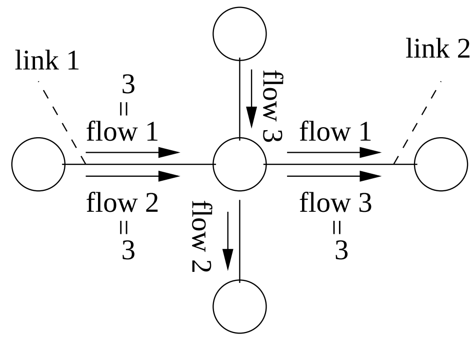


Figure 7: One Case of Resource Assignment - max-min fairness

An assignment is proportional fair if the aggregate of proportional changes of an alternative assignment is never positive [25]. In this case, flow 1 would occupy 2 MBPS and flows 2 and 3 get 4 MBPS each as shown in Figure 8.

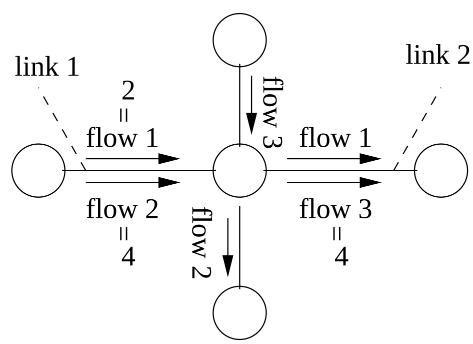


Figure 8: One Case of Resource Assignment - proportional fairness

- Optimization of total utility function

From the service provider's point of view, their revenue is the function to be maximized. In this case, since all three flows have the same utility function, they make no difference to the service provider [2]. Any assignment would give the service provider same income. Now we change the utility function so that the proportional increase to the revenue with the same resource allocation increase differs among those three flows. Suppose U_i represents the revenue from flow

[2] Here we assume the utility function also represents the revenue function.

i and B_i stands for the bandwidth allocation for flow i. The revenue function as: $U_i = i \times B_i$. Then assigning all the bandwidth to flow 2 and 3 makes the service provider gain the most. The resulting assignment satisfies a weighted proportional fairness criterion.

3.4 Summary of previous algorithm limitations and our new contributions

A fair amount of research work was done to design an effective pricing model to realize fair resource allocation, with some bias towards either customer or the service provider. However, the current limitations on previous algorithms are:

1. Customers cannot specify how much they are willing to pay.

2. The network is restrained to only have one single bottleneck link.

3. A feedback loop consisting of confirmations and renegotiations between the service provider and the customers is unrealistic for a real network.

4. The IntServ architecture is not feasible for the Internet.

5. The assumption of the utility (or revenue) function is not realistic.

To address those points, our research contributions include the following:

- We model the transaction between the customers and the service provider. The novelty of our model is that customers have the freedom to propose their required service and the price. The service provider aims to maximize his profit by deciding which customers to serve. We study resource allocation both at edge nodes and core nodes. The final solution depends not only on the service provider's revenue but also on the customer's willingness to pay for the service.

- We relax the constraints of the revenue model. The revenue function we use is simple and intuitive. It does not make any assumption of the function's characteristic (eg. differential, continuous, etc). The trade-off is that it may take more effort to find the solutions.

- Our auction models work better than flat rate pricing both in congested and in non-congested situations. The reasons are: a) In the auction model, when the network is congested, the service provider can choose which flows to pass and which not to pass in favor of maximizing his revenue; and b) when the network is over-provisioned, the service provider can reject lower class traffic if leaving them in gives the service provider lower revenue. Flat rate pricing does not have those capabilities.

- The thresholds generated by the auctions give the condition for admitting new clients.

- Flow-based input information is transformed into class-based DiffServ domain parameters. This makes the auction idea readily implementable in DiffServ.

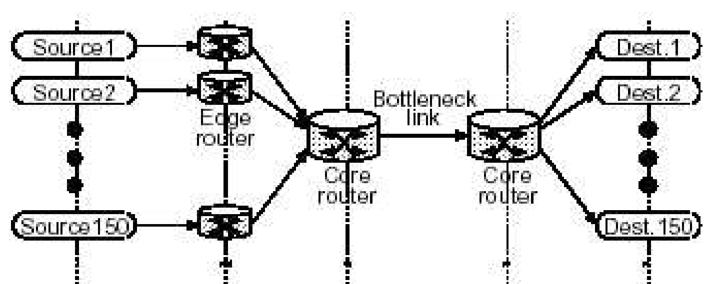


Figure 9: Single Bottleneck link network model

We give the optimal solutions for a single bottleneck link network in chapter 4. We then extend the formulations to multiple link networks and use heuristic algorithms to solve this more realistic problem. Extensive simulations were conducted to show the advantages of our algorithm. Furthermore, we examine the effects of allowing bids to either last the lifetime of an entire flow versus requiring dynamic bidding as network conditions change.

CHAPTER IV

OPTIMAL SOLUTIONS FOR A SINGLE BOTTLENECK LINK NETWORK

4.1 The Case with Base Price Fixed

The network model that we initially use makes the assumption that the network can be abstracted into a single bottleneck capacity [48](we later remove this assumption). This assumption allows us to simplify the network to a single link, furthermore, the capacity of the network is summarized into one number, namely, the capacity of the bottleneck link.

We consider a system that takes customers' bids and returns thresholds for both price and service offered for each class to maximize the SP's revenue. Flows coming and joining in later will be subjected to those thresholds during admission control. The recalculating of the thresholds occurs with some fixed frequency. The historical prices and the corresponding services offered help customers to evaluate the service. They then make bids by proposing a minimum bandwidth and the price they would be willing to pay. The provider then provides a minimum bandwidth for each service class. Using this pricing strategy, we try to maximize the service provider's profit.

We further reduce the scope of the problem by fixing the base cost. In our utility function 1, there are two parts. One is called base cost, which does not depend on the resource consumption. The other part depends on the bandwidth assigned to each client. The base cost comes with a base bandwidth. Customers may bid for extra bandwidth outside the base bandwidth. The customers need to bid for extra bandwidth which incurs an additional charge. In section 4.2, the base cost is modified from fixed to also biddable by the clients.

4.1.1 Problem Formulation

User experiments reported in [1] [29] suggest that utility functions typically follow a model of diminishing returns to scale, that is, the marginal profit as a function of bandwidth diminishes with increasing bandwidth. Hence [55] develops a general revenue function as a function of bandwidth:

$$U_{kj} = U_{0j} + W_j \log \frac{X_{kj}}{L_{kj}} \tag{1}$$

where U_{kj} stands for the revenue from client k, which belongs to class j. W_j is the sensitivity of the price to bandwidth for class j. U_{0j} is class j's monetary "opportunity" that the customer perceives at the lowest QoS level.

The base price for each class is fixed by an internet service provider. Customers bid by specifying a sensitivity coefficient and a minimum bandwidth. The objective is to maximize the service provider's revenue, subject to the constraints based on the system's available resources. The mathematical formulation is as follows.

Decision variables:

$$Z_{ij} = \begin{cases} 1; & \text{if client i is admitted to class j} \\ 0; & \text{otherwise} \end{cases}$$

X_{ij} = bandwidth obtained by client i for class j;

L_{mj} = minimum bandwidth for class j;

W_j = price for class j;

Objective function:

$$max \sum_{j=1}^{2} \sum_{i} (U_{0j} * Z_{ij} + W_j \log \frac{X_{ij}}{L_{mj}}) \tag{2}$$

Subject to:

$$
\begin{cases}
\sum_{j=1}^{2} \sum_i X_{ij} <= Q \\
X_{ij} \geq L_{mj} - (1 - Z_{ij}) * M \\
W_j \leq W_{ij} + (1 - Z_{ij}) * M \\
X_{ij} \geq V_i - (1 - Z_{ij}) * M \\
X_{ij} \geq X_j - (1 - Z_{ij}) * M \\
X_{ij} \geq 0 + Z_{ij} * M \\
X_{ij} \geq 0; L_{mj} \geq 0; W_j \geq 0 \\
X_{ij} \leq X_j
\end{cases}
$$

Parameters:

U_{0j} : base price for class j

Q : total bandwidth

V_i : minimum bandwidth required by client i

M : a very large positive number

The scenario is that each customer proposes a W_{ij} and L_{ij}. We have to decide which flows are admitted for each class such that the provider's revenue is maximized. For the flows admitted to class j, we adopt the minimum W_{ij} as W_j – the sensitivity coefficient threshold for class j and the maximum L_{ij} as L_j – the bandwidth threshold for class j. So each class has a uniform W and L value.

4.1.2 Optimal Solution

First, we introduce an optimal solution within one class. Then, we will expand that into multiple classes.

4.1.2.1 Solution in One Class

Within one class, the objective function becomes max $\sum_i (U_0 + W \log \frac{X_i}{L_i})$. Suppose C is the set of all customers who bid, M is the set of accepted customers and Q_j is the assigned bandwidth to class j. The bid values from customer i are L_i and W_i. According to our policy of choosing L_j and W_j, L_j should be the maximum value from the set M, while W_j the minimum of W_i i∈M. This rule also ensures that L∈{L_x,x∈C} and W∈{W_x, x∈C}. Therefore, L_j = maxL_i, {i∈M}, and W_j =minW_i,{i∈M}. We construct another set S with all the combinations of L_x, W_x values. S={(L_x,W_y), x∈C, y∈C}. For each combination (L_x,W_y), we define a number k as the number of flows, whose L value is less than or equal to L_x and whose W value is greater than or equal to W_y. Record the number of flows as k. k is the number of flows that can be possibly admitted if L_j= L_x and W_j=W_x.

As defined earlier, each flow shares Q_j/k amount of bandwidth. If Q_j/k < L_j, there is insufficient bandwidth to support all clients. So we have to reduce the number k until the inequality Q_j/k >L_j is valid. When we have number k for each combination, the corresponding profit $U_x = \sum_x (U_0 + W \log \frac{X_x}{L_x})$ can also be generated. So each combination has 2 values associated: k_x and U_x. Now, we need to filter out some unqualified combinations by using the value m_j, the number of flows that can be accepted in class j. Beginning with m_j=1, find all the combinations whose $k_x \geq m_j$ and choose the one with the highest U_x, recorded as U_{x_1}. Increment m until m is equal to the total number of flows n, and calculate a U_x for each m. From the set { U_{x_1}, U_{x_2},..., U_{x_n}}, the U_{x_i} with the highest profit is selected. We then can get the corresponding m_j and (L_x,W_x). Therefore, we have obtained the best values of L_i, W_i, and m_i for the class. The corresponding value U_j is the service provider's optimal profit for class j. L_j and W_j are used as the bid thresholds for the flows in class j.

To elaborate on this procedure, we give a simple example as follows. Suppose in this single class, we have the following bids:

C1:(L_1=2M,W_1=10); C2:(L_2=3M,W_2=11); C3:(L_3=2.5M,W_3=9); C4:(L_4=10M,W_4=12); C5:(L_5=8M,W_5=6).

The available bandwidth is given at 12M and the base price is given at 20. From these bids, we can build a matrix, including all the combinations of L_x and W_x.

$$\begin{bmatrix} (L_1, W_1) & (L_1, W_2) & (L_1, W_3) & (L_1, W_4) & (L_1, W_5) \\ (L_2, W_1) & (L_2, W_2) & & & \\ \cdots & & & & \\ (L_5, W_1) & (L_5, W_2) & .. & .. & .. \end{bmatrix}$$

Starting from (L_1, W_1), we look for clients whose L$\leq L_1$, W$\geq W_1$. Only client 1 itself satisfies the criteria. So, k=1. Test by dividing Q_j/k = 12M and it's greater than L_1=2M; therefore, it's a valid k value for $(L_1 = 2, W_1 = 10)$ combination. Provided L=2 and W=10, the revenue is $U = 20 + 10 log 12/2$=27.78. Proceed in this way to get all the ks and Us.

$$\begin{bmatrix} (L_1, W_1)^{k=1}_{U=27.78} & (L_1, W_2)^{k=1}_{U=28.56} & (L_1, W_3)^{k=1}_{U=27.0} & (L_1, W_4)^{k=1}_{U=29.33} & (L_1, W_5)^{k=1}_{U=24.66} \\ (L_2, W_1)^{k=1}_{U=26.02} & (L_2, W_2)^{k=1}_{U=26.62} & & & \\ \cdots & & & & \\ (L_5, W_1)^{k=2}_{U=49.54} & (L_5, W_2)^{k=1}_{U=26.62} & .. & .. & .. \end{bmatrix}$$

The next loop deals with the number of flows getting admitted. When m=1, all combinations with k$\geq$ 1 will be considered and the highest one with U value is U_1. Next, m increases to 2, and a corresponding U_2 is chosen. Then, we have {U_1, U_2, U_3, U_4, U_5}. Among them, the highest U is our final revenue. The m value associated with that U is the number of clients service provider should admit. The combination (L, W) that generated the U value is service provider's threshold. The completed solution is as shown in Figure 10.

Until now, we have generated the L and W values to maximize service provider's revenue, based on current clients' bids. Next, we introduce some properties to show how the service provider should admit new flows to maintain its maximum profit.

Property 1: If W_j and L_j are kept the same, as long as the inequality $Q_j/m_j > L_j$ is

```
Single Case(L[i], W[i], U_0, Q):
Input:Customer i's bid: L[i], W[i]; base price: U_0; network's capacity: Q
Output: L, W and X_i: assigned bandwidth to customer i.
    for i = 1 to n
        for j = 1 to n
          Combination[i][j] = L_i, W_j
        endfor
    endfor
    for i = 1 to n
        for j = 1 to n
          for t = 1 to n
            L_i, W_j <- Combination[i][j]
            if (L_t < L_i && W_t > W_j)
              k = k + 1
            fi
          endfor
          while (Q / k < L_i) k = k - 1
          k -> k[i][j]
          U[i][j] = U_0 + W_j * log(Q/(k * L_i))
        endfor
    endfor
    for m = 1 to n
        for i = 1 to n
          for j = 1 to n
            while (k[i][j] >= m)
              U_m = max U[i][j]
          endfor
        endfor
        if (U_m > U_max)
          U_max = U_m
        fi
    endfor
```

Figure 10: Optimization Algorithm for Single Class

valid, it's always true that the more flows added in, the higher value U_j is.

Proof:

The revenue function is:

$U_j = m_j * U_{0j} + m_j * W_j \log \frac{Q_j}{m_j L_j}$

Its derivative is:

$$\frac{\partial U_j}{\partial m_j} = U_{0j} + W_j \log \frac{Q_j}{m_j L_j} - W_j \tag{3}$$

Since $Q_j/m_j > L_j$, as long as U_{0j} is greater than W_j, $\frac{\partial U_j}{\partial m_j}$ is always greater than 0. That guarantees that U_j is a strictly increasing function.

Using property 1, the service provider can increase the revenue by admitting more flows with fixed W, L values. Therefore, after the bidding thresholds have been decided, property 1 allows the provider how to admit new flows to maximize the profit.

4.1.2.2 Solution in Multi-Classes

When we put the scenario in Multiple classes, the objective function and corresponding constraints formulated as a Lagrangian are

$$\begin{cases} max[\ m_1 * (U_{01} + W_1 \log \frac{Q_1}{m_1 L_1}) + m_2 * (U_{02} + W_2 \log \frac{Q_2}{m_2 L_2})]; \\ Q_1 + Q_2 = Q; \end{cases} \tag{4}$$

We have the following solution

$$\Rightarrow \begin{cases} Q_1 = (m_1 W_1)/(m_1 W_1 + m_2 W_2) * Q \\ Q_2 = (m_2 W_2)/(m_1 W_1 + m_2 W_2) * Q \end{cases}$$

We notice that the W value does not affect Q, m, and L. While Q, m and L are closely related. When value m and L are fixed, W and U are directly related. In the last section, we introduced the method of searching all (L_x, W_x) combinations in one class to get the value k. We extend last section's algorithm here.

First, start from m=1 and L_i (i=1) and check all the combinations (W_x) with L_1. From these, the effective ones are those with k≥m; then, choose the one with the highest W. Now, we have m, L, and W for class j. The same procedure applies to other classes and we have m_1, L_1, W_1, m_2, L_2, W_2. Now Q_1, Q_2 can be solved according to equation 6. Lastly, we check the feasibility of each solution by calculating Q_1/m_1,Q_2/m_2. Only if all of them are greater than L_1, L_2 respectively, do we consider this a feasible solution and record the corresponding U value. Otherwise, it is abandoned. Following the same steps by changing the value of m and L_i, we can get all the possible feasible solutions. Finally, among all the feasible solutions, the highest U is the optimal solution. The detailed procedure is shown in Figure 11.

4.1.3 Simulation and Analysis

We randomly generate a set of clients who participate in the auction. Each client is associated with a bid. The bid include the required service class, desired minimum bandwidth, and the price they are willing to pay. We use their bids and the network's capacity as inputs into our multi-class pricing model. The service provider's revenue is produced as outputs. We conduct simulations comparing the single-class and multi-class solution.

We manually assign a fixed network capacity to each class. Then, our single-class solution is used to solve the case individually within a class. The final revenue is the sum of two class revenues. The motivation of doing so is to show that the multi-class solution takes into account not only the single class, but also the competition among those classes. The process is actually a two-step auction. In the first step, clients within each class compete with each other. In the second, each class relaxes and maximizes their bandwidth usage. Finally, it comes to a converged balance point where it reaches the highest point of the SP's final revenue.

Input:Customer i's bid: L[i], W[i]; base price: U_0; network's capacity: Q
Output: L[j], W[j] for class j and X_{ij}: assigned bandwidth to customer i in class j.

```
for m = 1 to n
     for class number, j = 1 to 2
       for i = 1 to n
           for l = 1 to n
             Call Single Case(L[l], W[i], U_0, Q)
             if (k[i][l] >= m && W[i] >= W_max)
               W_max = W[i], t = i
             fi
           endfor
       endfor
             Array[j] = m[j], W[t]
         endfor
       Q_j = (m_j W_j)/(∑_{j∈N} m_j W_j) * Q,  ∀j ∈ N
       for j = 1 to 2
         for i = 1 to n
           if (Q_1/m_1 > L_X[i]  &&  Q_2/m_2 > L_X[i])
             if (temp=∑_{j=1}^{2} ∑_i (U_{0j} + W_[j] log X[ij]/L_[mj]) > max)
               max=temp;
             fi
             continue;
           fi
         endfor
       endfor
endfor
```

Figure 11: Optimization Algorithm for Multiple Classes

For our experiment, we compare the revenue generation of the multi-class algorithm and the single-class algorithm, where in the single-class algorithm, bandwidth assignment for each class is predetermined using one of three simple heuristics.

The three heuristics are described as follows:

1. Get the ratio of each class bandwidth assignment by selecting the highest bid from the desired minimum bandwidths for that class. For instance, the highest bid from class 1 is 20M, 12M from class 2. The ratio is then, 20:12. So, class 1 gets 20/(20+12) of total capacity. The same formula is used for class 2. This method is in favor of big customers from the upper class. The reasoning is that when a customer requires more resources, they might potentially be willing to pay more.

2. Use the number of customers who bid as the reference for calculating the ratio. For example, 2 customers bid for class 1, 4 for class 2. The ratio is 2:4. The amount of bandwidth is divided proportionally. This strategy is in favor of the lower class, which may have more customers. The idea is that the lower class may generate get more revenue because of its larger number of customers.

3. Calculate the total bandwidth bidded for each class if all the customers get admitted. The bandwidth allocation for classes would be proportional to the total bandwidth. For example, 2 customers in class 1 bidding for 2M and 3M; 3 customers, 6 customers in class 2 bidding for .09M, .1M, .15M, .2M, .21M, .08M. The ratio becomes: (2+3):(.09+.1+.15+.2+.21+.08).

We set the revenue results from multi-class optimization algorithm to the value one and compare the results to the three heuristics introduced above. The cases shown in Figure 12 are for six different sets of customer demands and bids. These customer demands and bids are varied to allow our algorithm to be compared to the

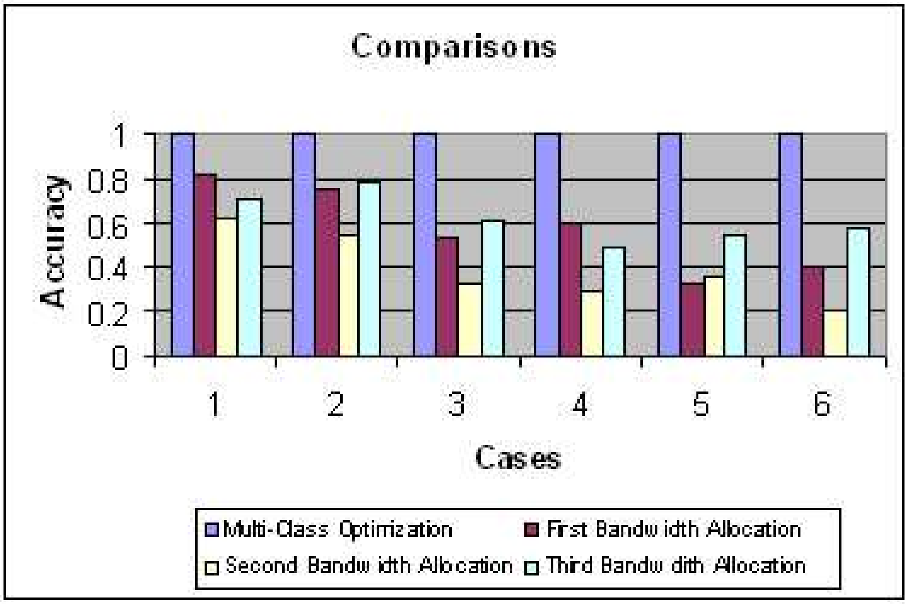


Figure 12: Comparison Results

other three simple algorithms over a variety of traffic inputs and bids. The revenue generated by our multi-class algorithm (set to unity) exceeds the revenue generated by the simpler algorithms.

The three heuristic methods have their own advantages in different situations. Method 1 and 2 are opposites of each other. From the simulation results, we can also see that when method 1 performs well, method 2 generates bad results, vice versa. Since it is hard to determine the customers' behavior, multi-class solution providers much better insight as how to efficiently utilize the network resources.

4.2 Introducing the Base Price as a Variable

4.2.1 Problem Formulation

The last section introduced the optimal bandwidth allocation to clients, based on a fixed base price. But fixed base prices are a *priori* information that a service provider must be provided. Given our argument for the difficulty of determining accurate information about the dynamics of the market, we propose to make the base price a variable in the auction algorithm. This change will also allow customers to bid on the base price, which often matters more than the sensitivity coefficient, and also makes it more similar to conventional auctions.

We still use the same utility function as in the last section:

$U_{ij} = U_{0ij} + W_{ij} \log \frac{X_{ij}}{L_{ij}}$

The auctions operate on sealed bids periodically. U_{ij} stands for the revenue from client i, who belongs to class j. U_{0ij} is client i's bid on U_0 — the base price. W_{ij} is client i's bid on the sensitivity coefficient. L_{ij} is his minimum required bandwidth. X_{ij} is the bandwidth allocated to customer i.

Customers bid for the base price, sensitivity coefficient and minimum required bandwidth. The objective is to maximize the service provider's revenue, subject to the restrictions imposed by the system's available resources.

The problem formulation is as show below:

Variables:

ϕ_j: set of clients admitted to class j;

U_{0j} : base price for class j;

U_{0mj} : bid for base price from client m (in class j);

X_{ij}: bandwidth obtained by client i for class j;

L_j : minimum bandwidth for class j;

L_{mj} : bid for minimum bandwidth from client m (in class j);

W_j : price sensitivity coefficient for class j;

W_{mj}: bid for price coefficient from client m (in class j);

Objective function:

$$max \sum_{j=1}^{2} \sum_{i \in \Phi_j} (U_{0j} + W_j \log \frac{X_{ij}}{L_j}) \tag{5}$$

Subject to:

For $\forall$ i$\in\Phi_j$

$$\begin{cases} \sum_{j=1}^{2} \sum_i X_{ij} \leq Q \\ X_{ij} \geq L_{mj} \\ W_j \leq W_{ij} \\ L_j = \max L_{ij} \\ W_j = \min W_{ij} \\ U_{0j} = \min U_{0j} \\ X_{ij} \geq 0; L_{mj} \geq 0; W_j \geq 0 \end{cases}$$

Parameters:

Q : total bandwidth

Each customer proposes his desired values of U_{0ij}, W_{ij} and L_{ij}. We have to decide which flows to admit for each class in order to maximize the service provider's revenue. On each link and in the same class, the accepted customers will pay the same price — namely the lowest bid among those accepted. Another complication arises because a client might be across multiple links and two links might have different thresholds. We resolve this by setting the client's price to the highest among those thresholds. Since the threshold for each auction equates the lowest bid among the winners, this scheme still ensures that the customer pays at most his own bid. The solution to the optimization problem formulated above provides an optimal bandwidth assignment to each individual customer as well as maximizing the service provider's revenue.

4.2.2 Optimal Solution

We notice that every flow in one class has the same threshold (U_{0j}, W_j, L_j) and flows within the same class will obtain the same bandwidth. Suppose that class j's assigned bandwidth is Q_j, each flow gets its own share of Q_j/m_j where m_j represents the number of flows admitted into class j, and generates the same revenue. We solve the problem using the objective function and corresponding constraints formulated as follows:

$$\max \quad \textstyle\sum_{j\in N} \; m_j * (U_{0j} + W_j \log \frac{Q_j}{m_j L_j}) \tag{6}$$

$$\textit{subject to}: \qquad \textstyle\sum_{j\in N} Q_j = Q \tag{7}$$

The solutions are obtained by Lagrange relaxation:

$$Q_j = (m_j W_j)/(\sum_{j\in N} m_j W_j) * Q, \; \forall j \in N. \tag{8}$$

Therefore, given (m_j, U_{0j}, W_j and L_j), we can obtain Q_j as in equation (8). According to auction policy, $U_{0j} = \min\{U_{0ij}, i \in M_j\}$, $W_j = \min\{W_{ij}, i \in M_j\}$ and $L_j = \max\{L_{ij}, i \in M_j\}$. This also implies that each combination (U_{0kj}, W_{mj} and L_{nj}), where $k, m, n \in M_j$, provides a candidate value set for class j. Therefore, based on the bids in class j, we make all combinations in terms of U_{0ij}, W_{ij} and L_{ij}. For each combination, which corresponds to one predetermined set of value (U_{0j}^*, L_j^* and W_j^*) for class j, we sort out all the flows such that $U_{0ij} \geq U_{0j}^*$, $W_{ij} \geq W_j^*$ and $L_{ij} \leq L_j^*$ $\forall i \in M_j$. Record the number as k. So each combination has a k value associated with it.

Now, for each class $j \in N$, starting from m_j=1 and L_{1j}, check all the combinations of (U_{0ij}, W_{ij}). From these, the effective ones are those with $k \geq m_j$. From our previous assumption that U_{0ij} is a dominant pricing factor, which is given the highest priority to choose the combinations with largest U values. From among the largest U value set clients, choose the one with the highest W value. Until now, we obtained values of U_{0j}, W_j and L_j for each class j. Then we have all inputs ($U_{0j}, m_j, L_j, W_j\ \forall j \in N$) for the solution of equation 6, and $Q_j\ \forall j \in N$ can be calculated as in equation 8. We have specified earlier that each admitted flow in one class shares the same amount of bandwidth and this bandwidth has to be greater than or equal to their bids. We are using that property to check the validity of each possible solution. If and only if $Q_j/m_j \geq L_j\ \forall j \in N$, the solution is a qualified candidate. If so, by using those values as well as U_{0j}, the total revenue is computed. Otherwise, this set of solution values is abandoned. Following the same steps by changing the value of m_j and L_j, we can obtain all the possible feasible solutions. Finally, the solution with the highest total revenue is optimal.

So now we have the optimal solutions for calculating the best thresholds as well as the assigned bandwidth to each class and client, in terms of maximizing service provider's revenue. The next question is how should we use the thresholds to admit new flows. We know that the auction occurs with a fixed time interval. During that interval, when new customers want to join in, they present their bids. Then, if it is possible to admit them in, they can get into the network. Otherwise, they have to wait for the next auction to take place. How does the service provider decide if letting them in is going to benefit him or not? We propose the following property to explain what procedure the service provider should follow in order to make a good judgement.

Property 1: If Q_j and (U_{0j}, W_j, L_j) are fixed, as long as $Q_j/m_j > L_j$, U_j is a strictly increasing function of m_j.

Proof:

The revenue function is:

$$U_j = m_j * U_{0j} + m_j * W_j \log \frac{Q_j}{m_j L_j} \tag{9}$$

Its derivative is:

$$\frac{\partial U_j}{\partial m_j} = U_{0j} + W_j \log \frac{Q_j}{m_j L_j} - W_j \tag{10}$$

Since $Q_j/m_j > L_j$, and U_{0j} is greater than W_j (which is our assumption), $\frac{\partial U_j}{\partial m_j}$ is always greater than 0. That guarantees that U_j is a strictly increasing function of m_j.

Using property 1, the service provider can increase the revenue by admitting any new flow i into class j as long as $L_{ij} < L_j$, $W_{ij} > W_j$ and $U_{0ij} > U_{0j}$. So, after the bidding thresholds have been decided, property 1 gives the service provider a guideline as to how to admit new flows.

4.2.3 Simulation and Analysis

We simulate the service provider and clients that use our bidding procedure and operate the resource allocation under the auction scheme. We run the same experiments in the market model with various network loads and various service provider and customers' valuations.

4.2.3.1 Simulations Set Up

As an example, we assume that the Best Effort (BE) class is charged $35 per client per month. Divide that by days and minutes, 0.00135 cents per minute needs to be paid for BE traffic. Let a mcent (or a unit) be equal to 1/1000 cents, so the charge for BE as 1.35 mcents. Based on this, we define EF traffic's price twice as much as BE's and AF's price is 1.5 times. These values are the service provider's price thresholds for each class. Any customer who bids lower than the threshold price will be rejected. The customers' valuations for EF and AF are assumed to be normally distributed.

In the flat rate scenario, a customer is admitted if and only if his valuation (which is same as the bid in the auction context) is greater or equal to the fixed price set by the service provider. The revenue is the number of customers multiplied by the price.

All the parameters are given in the Table 1 that are used to determine the revenue generated by service provider. The customers' mean valuation (MV) and standard deviation (SD) are within a fixed range instead of a fixed number. This is because we vary the MV and SD in the simulations to show that our algorithm efficiency is not affected by how the parameters are chosen. It works for all general settings.

Table 1: The assumptions that are used in the simulations

	Parameters		
	Floor Value	Customers' mean valuation	Standard deviation
EF traffic	2.7 *mcents*	[2.7 - 5.4]*mcents*	[1 - 2]*mcents*
AF traffic	2.0 *mcents*	[2.0 - 4.0]*mcents*	[1 - 2]*mcents*

4.2.3.2 Results and Analysis

To compare the optimal resource allocation results with flat rate pricing, we vary the fixed rate cost from 0.2 to 2.2 times the mean valuation of what the customer are willing to pay. We ran our algorithm with fixed set of parameters and compared with the revenue generated by fixed rate. The fixed rate for each class changes within the range of (20%–220%)* MV where MV is the customers' mean valuation. We set the fixed price range as [20%MV, 220%MV] because we notice that when the price goes beyond this range, either too low or too high, the revenue tends to drop to close to 0. We want to test and show that the algorithm's performance doesn't depend on how we set up the parameters. In another word, the auction scheme that we propose is robust.

The optimal solution is good for any network condition, either congested or overprovisioned. When it's overprovisioned, since the lowest winning bid is the threshold, not necessarily that all the customers will be admitted, if by adding low value customers brings down the service provider's revenue. Therefore, it ensures that customers benefit most by bidding the true values. The flat rate pricing has no control on the overprovisioned network. The total revenue is just the number of customers multiply the charge per head. Likewise, when the network gets congested, the auction picks up the most valuable customers while drop the others. Flat pricing can only randomly drop customers, which not only reduces the revenue income but the overall customers' satisfaction. In order to show auctions work for various load, we vary the network load from 70%, 100% to 140% as well as the customers' mean valuation.

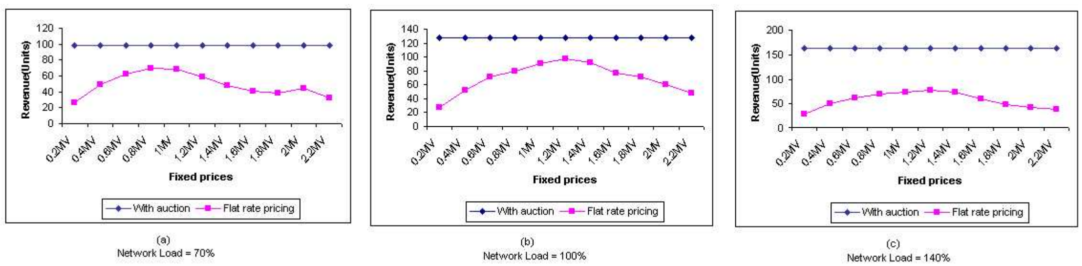


Figure 13: The revenue comparison (customers' mean valuation is 2.7 and 2.0 mcents)

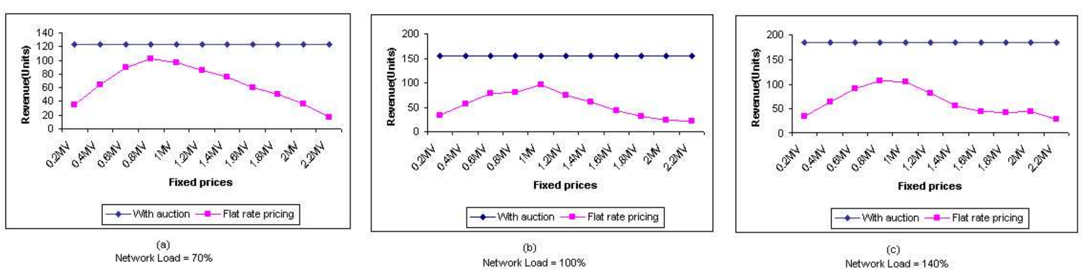


Figure 14: The revenue comparison (customers' mean valuation is 4.0 and 3.0 mcents)

Figure 13 shows the revenue comparison between our optimal algorithm and flat rate pricing when the customers' valuations are normally distributed with a mean of 2.7 mcents and a standard deviation of 1 for EF mcents and mean 2.0 mcents and standard deviation 1 mcents for AF. The subgraph (a), (b) and (c) show the results with network load at 70%, 100% and 140% respectively. The X-axis points

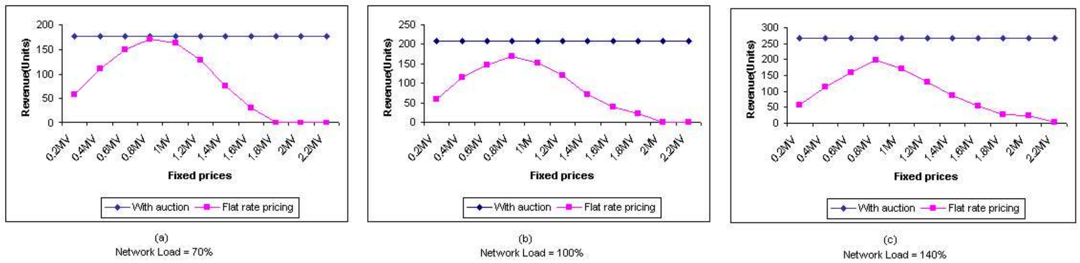


Figure 15: The revenue comparison (customers' mean valuation is 5.4 and 4.0 mcents)

the variation of flat rate prices, which go from 0.54 mcents for EF and 0.40 mcents for AF to 5.94 mcents for EF and 4.40 mcents for AF. The Y-axis is the total revenue generated by either scheme.

We then vary customers' mean valuation to 4.0 mcents for EF and 3.0 mcents for AF (standard deviation remains the same as before). The comparison results are shown in Figure 14. The flat rates vary from 0.8 mcents for EF and 0.6 mcents for AF to 8.8 mcents for EF and 6.6 mcents for AF. Again, those three subgraphs (a), (b) and (c) represents the different network load at 70%, 100% and 140% respectively. Lastly, Figure 15 shows how the network behaves when the mean valuation changes to 5.4 mcents for EF and 4.0 mcents for AF. The flat rates vary from 1.08 mcents for EF and 0.8 mcents for AF to 11.88 mcents for EF and 8.8 mcents for AF.

From all the results, we can see that the auction scheme predominantly outperform fixed pricing. In all cases, auctions generate more revenue than the fixed rates. The revenue generated by the fixed rate pricing has similar curves because when the fixed rate is very low, even when it wins all the customers, the sum of the payments is low and when the rate is high, it loses customers which also reduces the service provider's profit. In some cases, the curve fluctuates. This is because when the fixed rate increases, the number of admitted customers decreases. That causes the revenue function to not be linear. Also, it shows the same trend that as the network load increases, the gap between auctions and fixed price increases. This shows that auctions performance improves when the system gets congested. This is because auctions offer the service providers more options to choose the most valuable customers and drop others. It also causes customers have to compete for bandwidth by raising their prices.

4.3 Problems Remaining

At this point, we have provided an auction strategy on a single bottleneck link network. It generates optimal revenue for service providers, given the customers' bids. However, this network model is not a realistic model of the current Internet infrastructure. We shall now prove this assertion. If we can extract an entire network into a single bottleneck and solve the optimization problem on that link, and the solution is optimal for the entire network, then we can say that we have a final solution for all cases in the network. Let's see whether this assumption is valid.

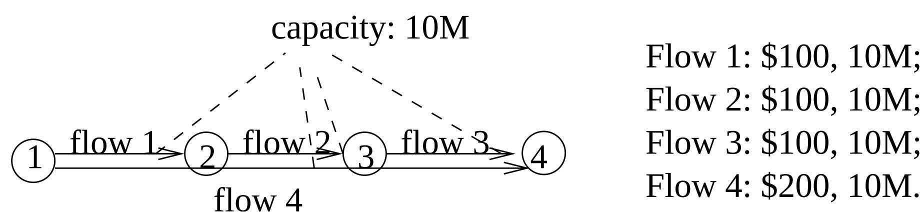


Figure 16: A Single Link Network Example

Suppose we have a network as shown in Figure 16. Flow 1 runs from node 1 to 2, flow 2 from node 2 to 3, flow 3 from node 3 to 4 and flow 4 from node 1 to 4. The link capacity for each link is 10MBPS. All flows require 10MBPS. Flow 4 bids $200 and flow 1,2, and 3 bid for $100. If we extract one bottleneck link, we pick up the link from node 1 to 2. The optimal solution on this single link gives us the result that flow 4 is admitted, while flow 1 is rejected. Since flow 4 is admitted and it occupies all of the available bandwidth, flows 2 and 3 are dropped as well. As a result, the service provider gets a revenue of $200. However, if flow 4 is dropped while the other 3 are admitted, the revenue is $300. This means that the single-link optimal solution is not optimal for the entire network. While we need to find a new scheme for the more general problem, the solution for the single-link case gives us a good foundation to build upon.

In the next chapter, we will further relax the constraint to allow more than one single bottleneck link existing in the network. Customers still bid for the base price, price sensitivity coefficient and minimum required bandwidth. However, those bids are under the network general view, which make the problem more complicated. We will also present our solutions in the next chapter.

4.4 Summary

We considered a general DiffServ network, which can be extracted into one single bottleneck link network [?] and studied the problem of maximizing the service provider's profit using pricing. We presented two novel optimal pricing strategies of maximizing the service provider's revenue based on clients' bids of price as well as desired service. In section 4.1, we examined a solution where the base price was chosen by the service provider. When the service provider chooses the base price, market dynamics are not allowed to fully function. In a fixed base price scenario, if a customer bids zero for additional bandwidth (price sensitivity coefficient), the customer is much more likely to lose the auction. Varying the base price removes this undesirable characteristic of section 4.1. The scheme proposed in section 4.2 gives customers the option of choosing how much they want to pay for their required service and this makes the base price a variable. The optimal solution provides the thresholds for each service class according to network resource availability. The thresholds can also be used as a future reference for admitting new clients. We compared the revenue generated by auction and fixed pricing. Our auction scheme generates the best result even when varying the parameters. Our results show that the auction strategy beats the fixed rate pricing scheme.

CHAPTER V

HEURISTIC ALGORITHMS FOR MULTIPLE LINK NETWORKS

In this chapter, we discuss auction set up in a multiple bottleneck link network. We assume that routing paths for each client have been given to us [1]. We have the network topology knowledge and customers' bids as well as their routing paths. The problem here is given a network and customers' bids, how does the service provider decide on who are the most valuable customers and how can the service provider maximize his revenue. The detailed problem formulations and solutions are presented below.

5.1 Problem Formulation in a Multiple Bottleneck Link Network

We adopt the same revenue function as we introduced in the single link case. However, moving from the single bottleneck link network model to the multiple bottleneck-link network model, we have to consider the flow's bandwidth assignment consistency across links along the flow's routing path. Therefore, we transform all the notation that we introduced earlier by adding a superscript ℓ to represent the value on link ℓ. A customer's bid is valid for all the links that his routing path crosses. The bandwidth capacity of link ℓ, denoted by C_ℓ (ℓ=1,2,..., L). Let ℓ_{ij} represent the set of links that flow i of class j crosses in the network. A flow's bandwidth assignment also needs to be consistent on the routing path, thus we enforce $X_{ij}^{\ell_1} = X_{ij}^{\ell_2}$ if $\ell_1, \ell_2 \in \ell_{ij}$. We introduce

[1] How to choose a best routing path for the clients is beyond the scope of this research.

following additional parameters and variables before describing the formulation.

Parameters:

M : a very large positive number

L_{ij}^{ℓ} : minimum bandwidth for client i in class j on link ℓ

U_{0ij}^{ℓ} : base price for client i in class j on link ℓ

W_{ij}^{ℓ} : price sensitivity for client i in class j on link ℓ

Decision variables:

Z_{ij} : 1 if client i in class j is admitted; 0 otherwise

X_{ij}^{ℓ} : Bandwidth obtained by client i in class j on link ℓ

Objective function:

$$\max \sum_{j=1}^{N} \sum_{i \in M_j} (U_{0j} + W_j \log \frac{X_{ij}}{L_j}) * Z_{ij} \tag{11}$$

Subject to:

$$\begin{cases} \sum_{j=1}^{N} \sum_i X_{ij}^{\ell} \leq C_\ell \ \forall \ell \\ X_{ij}^{\ell} = X_{ij} \ \forall \ell \in \ell_{ij} \\ X_{ij} \geq L_{ij}^{\ell} - (1 - Z_{ij}) * M \ \forall i, j, \ell \\ X_{ij} \leq 0 + Z_{ij} * M \quad \forall i, j \\ U_{0j} \leq U_{0ij}^{\ell} + (1 - Z_{ij}) * M, \ U_{0j} \geq 0 \ \forall i, j, \ell \\ W_j \leq W_{ij}^{\ell} + (1 - Z_{ij}) * M, \ W_j \geq 0 \ \forall i, j, \ell \\ L_j \geq L_{ij}^{\ell} + (1 - Z_{ij}) * M \ \forall i, j, \ell \\ X_{ij} \geq X_j - (1 - Z_{ij}) * M \ \forall i, j \\ X_{ij} \leq X_j \ \forall i, j \end{cases}$$

The problem formulated is a non-convex integer and nonlinear problem. To the best of our knowledge, it does not fall into any existing optimization problem category that has an absolute global optimal solution. Due to the complexity of the problem, we resort to heuristic algorithms which can generate better results in terms of accuracy, time consumed and calculation delay as opposed to strict optimization methods.

5.2 *Intra-link and Inter-link Solution (IIS)*

We approximate the optimal solution by splitting the problem into two steps. First we find an assignment for each individual link. Then we integrate the assignments into one multi-link assignment for the big picture. We first present the intra-link solution.

5.2.1 Intra-link auction

The optimal solution on one link is a special case for a network with only one link. Even though we had the optimal solution as presented in Section 4.1.2, however, the search space and time complexity is high, so it is not appropriate to adopt the optimal solution here, especially when a large amount of customers are involved. We propose a simple heuristic algorithm to approximate the optimal results.

1. Golden search with quadratic interpolation technique [49] is employed to decide each classes' bandwidth allocation. The relationship between revenue and bandwidth allocation ratio between two classes is an arbitrary function. It has some local maximal and minimal points. However, it is a continuous function and when the bandwidth assignment to each class is fixed, the revenue is also determined. We think quadratic interpolation followed by golden search can handle it in a simple but good way in terms of finding the optimal point where the bandwidth allocation ratio between two classes generates highest revenue.

   If each class' bandwidth allocation is known, step 2 is to find a good customer assignment and its corresponding revenue.

2. For each class, we perform the following operations.

   - Build a queue of the bidding clients in decreasing order of their U_{0i}/L_i such that S_1:$\{U_{01}/L_1,\ U_{02}/L_2, \ldots, U_{0n}/L_n\}$.

- Dequeue the first client, say flow i. If we can admit it, do so. Otherwise drop the client from the auction. By "can admit" we mean c/m $\geq$ max$\{L_1,..,L_m\}$, where c is the total bandwidth assigned to this class, m the number of admitted flows (including flow i) and max$\{L_1,..,L_m\}$ is the maximum L value from those flows which have been admitted. Repeat this step until the queue is empty.

- Calculate the total revenue R_1. Let the accepted clients be denoted Acc_1:{i, j,...}. U_{0i}/L_i is the ratio of quality to price. The larger U_{0i}/L_i is, the more revenue the service provider generates, thus the prioritizing of the clients with larger U_{0i}/L_i. However, there might be the case that a customer requires a huge amount of bandwidth, and is willing to pay a lot more as well — he has a high U_0 bid. In this case, U_{0i}/L_i might be low, which gives the clients a low priority. If this customer gets a good amount of bandwidth, he could generate very good revenue because of his high U_0, possibly more than the sum of the "small" clients. To address this, we create another queue to detect these valuable customers.

- Create a queue in decreasing order of U_{0i} such that S_2 :$\{U_{01}$,U_{02},..., $U_{0n}\}$.

- Apply the same procedure as applied to S_1 to queue S_2. The total revenue R_2 is calculated at the end and we denote Acc_2 the accepted clients.

- We simply take the better of these two results: $R = \max\{R_1,R_2\}$ and Acc the accepted clients corresponding to R.

- Now we look at the W value. W affects the secondary charge that the customer pays if he is allocated more bandwidth than his minimum requirement. When two customers bid the same U_0 and L, the one with a higher W should have a better chance of winning the auction. So it is reasonable to use W bids as a complimentary judgement on the admission, in addition to the procedure we introduced above. Again we form a customer queue, this time of all the clients yet to be admitted, in decreasing order of W, this is S_3:$\{W_i$, W_j,...,$W_n\}$.

- Dequeue the W-queue and check if swapping the client with the last accepted client in *Acc* yields a better revenue. If it does, perform the swap. Repeat this step until half of the clients are left on the W-queue. As W becomes smaller, it has less and less impact on the original accepted flow list and it is not necessary to adjust it any more.

- At this point we have the assignments for the accepted clients for each link and class. We let $X_{ij} = X_j$ for all i, be the bandwidth allocated to the link and class divided by the number of clients admitted. Meanwhile, we have W_j=min$\{W_{ij}$, if i gets admitted$\}$ and L_j = max $\{L_{ij}$ if i gets admitted$\}$ for the individual link. Now, it is time to put the pieces together.

5.2.2 Inter-link Adjustments

We need to perform inter-link adjustments to make the flows' assignment consistent throughout the network. We require $X_{ij}^{l_1} = X_{ij}^{l_2}$, where $l_1, l_2 \in l_{ij}$.

1. For each flow, set its assigned bandwidth to that of its bottleneck link — the link in the flow whose bandwidth per client is the lowest, i.e. the bandwidth for the relevant class in the link divided by the number of admitted clients. That can be expressed by :$X_{ij} = min\{X_{ij}^l\}$, where $l \in l_{ij}$.

   This will satisfy the consistency requirement. After this is done for all flows, there may be free bandwidth on some links for a certain class. Since the assignment was done independently on each link, it is very likely that the bottleneck links of various flows are spread out in the network, which yields a better chance for there to be extra bandwidth. In order to utilize some of this free bandwidth, the following adjustment is performed.

2. A set L is built to represent all the links which are currently under utilized. Start from the link k which has the minimum gap between the current usage and link capacity. We perform the following steps:

   - For each class c and link k that has extra bandwidth, denote the flows in class c running on link k $flow_k$.

   - Generate a set of *related* links for link k. A link l is said to be *related* to link k if k and l both belong to some flow f, or is related to any link that is related to l by the previous way.

   - If all the links related to k are under utilized, do the following.

     On each link, we impose the inequality:

     $$m * x + CurrentUsage \leq Capacity$$

     where m is the number of flows on the link from one class. x is the additional bandwidth for each flow in the class. CurrentUsage is the sum of the flows' assigned bandwidths. We solve these inequalities independently and then pick the smallest x. Now we need only to add x to the assigned bandwidth of each affected flow.

3. Update the links' current usage and delete link k from set L. Repeat the procedure above to the remaining links in the set until it is exhausted.

After the inter-link adjustments are done, we reach the local optimal solution.

5.3 *Simulation and Analysis*

We now study the performance of our algorithm in various settings in both a single bottleneck link and a multiple-bottleneck-link network. The scenario is that for auctions, each client bids for base price, minimum required bandwidth and the price sensitivity coefficient, the service provider uses the IIS algorithm to calculate the bandwidth assignment, thresholds and total revenue. We compare these results to flat rate pricing by using the same set of customers to calculate the revenue generated by flat rate pricing. In order to accurately monitor the performance, the simulations are set up as follows.

5.3.1 Simulation Design

As an example, assume that the Best Effort (BE) class is charged $35 per client per month. On average BE traffic needs to pay 0.00135 cents per minute. Let a mcent (or a unit) be equal to 1/1000 cents, so the charge for BE is 1.35 mcents. Based on this, we define EF traffic's price to be twice as much as BE's, which is 2.7 mcents per minute. Meanwhile, AF's price is 2 mcents per minute. These values are the service provider's price thresholds for each class. Any customer who bids lower than the threshold price will be rejected. The customers' valuations for EF and AF are assumed to be normally distributed.

In the flat rate scenario, a customer is admitted if and only if his valuation (which is the same as the bid in the auction context) is greater or equal to the fixed price set by the service provider. The revenue is the number of customers multiplied by the price.

In the bidding system, the valuations of the customers are used as their bids. The service provider would set the threshold values using the algorithm presented. When we have the thresholds, we also calculate the total revenue generated by the auction.

The comparison between revenue generated from the fixed price and the auction are done in various ways by varying the set of customers (and also their valuations), the network topology, the network load and the assumed fixed prices.

5.3.2 Simulation Results

Graphs are commonly used to model the topological structure of internetworks for studying problems ranging from routing to resource allocation. We use the GT Internetwork Topology Models to generate graphs that model the topological structure of the Internetwork. The details of this tool may be found in [60].

We made the same assumptions for the following grouped results labeled as Groups 1,2 and 3. The service provider's floor values are 2.7 mcents - EF and 2.0 mcents - AF. The ratio of EF and AF traffic is 30%-70%. The customers' valuations are normally distributed with a mean of 4 mcents and standard deviation of 2 mcents for EF and 3 mcents with standard deviation of 2 for AF.

Group 1(Figures 17, 18 and 19): In Group 1, we want to examine whether, in general, auction based pricing is better than flat pricing. For each subplot of simulation results, we keep the network load, customers' mean valuations and standard deviations fixed. In each case, customer bids are generated randomly with the same mean valuations and standard deviations. We also vary the number of customers in different cases in order to vary the network load while keeping other variables the same. First, we have 20 nodes and 200-400 customers. The fixed prices are 4 mcents for EF and 3 mcents for AF. The results are shown in Figure 17 (Figure 17(a)(b)(c) represent the network load at 70%, 100%, and 140%). The X-axis is the case number, representing customers' different valuations; the Y-axis is the total revenue. The cases are independent of each other.

We do the same simulations with 200 nodes, 400-1000 customers and then again with 1000 nodes and 3500-5000 customers. The results are shown in Figures 18 and

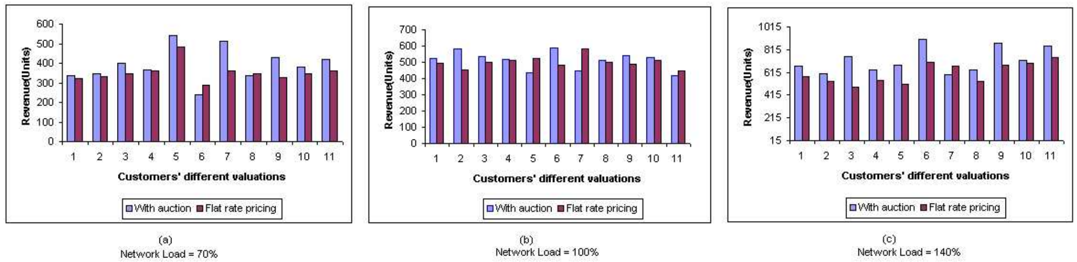


Figure 17: The Revenue Comparison Between Flat Rate Pricing and Auction (with fixed flat rate and 20 nodes 200-400 customers)

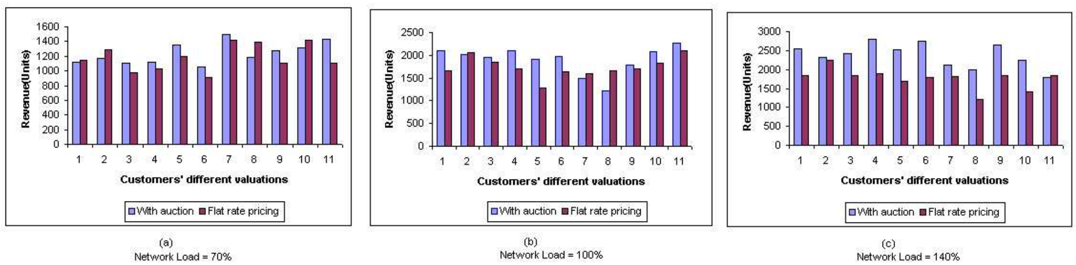


Figure 18: The Revenue Comparison Between Flat Rate Pricing and Auction (with fixed flat rate and 200 nodes 400-1000 customers)

19 respectively.

Group 2 (Figures 20, 21 and 22): In this simulation group, the flat pricing rate is varied from 0.2 to 2.2 times the mean valuation of what the customers are willing to pay. The rate for each class changes within the range of (20%–220%)* MV where MV is the customers' mean, which is 4 mcents for EF, 3 mcents for AF in this case. There are 20 nodes and 200-400 customers in the network. The network load is 70% and the customers' valuations are normally distributed with mean of 4 mcents and standard deviation 2 for EF mcents and mean 3 mcents and standard deviation 2 mcents for AF. Figure 20 shows a comparison of the bidding system and fixed pricing. In each graph, the X-axis stands for the fixed price variation, from 0.2MV to 2.2MV with 0.2MV interval. The Y-axis is the total revenue generated. Every graph denotes

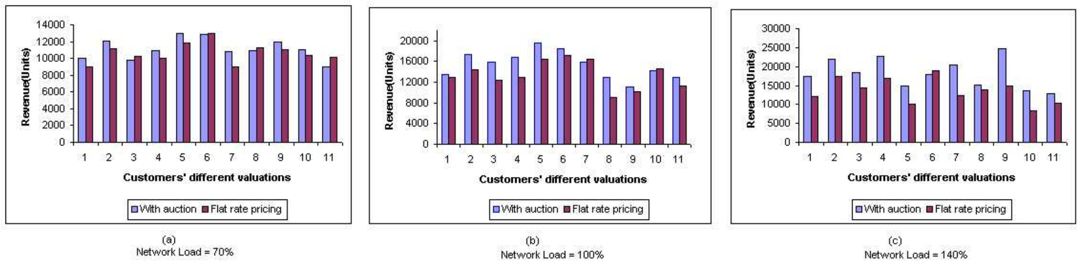


Figure 19: The Revenue Comparison Between Flat Rate Pricing and Auctions (with fixed flat rate and 1000 nodes 3500-5000 customers)

the scenario that revenue generated from one set of parameters as inputs (customers, network bandwidth) to our algorithm versus the revenue generated based on the same set of parameters but with various flat rates. Figures 20(b)(c) show how the network behaves when the network load is 100% and 140%.

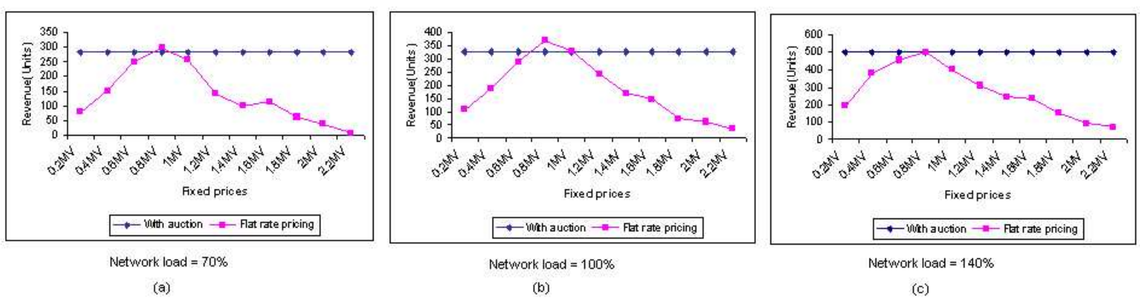


Figure 20: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 20 nodes)

Again, we run the same simulations on different network topologies, with 200 nodes and 400-1000 customers (Figure 21 with 70%, 100% and 140% network load) and 1000 nodes 3500-5000 customers (Figure 22 with 70%, 100% and 140% network load).

Group 3 (Figure 23): This group of experiments examines what happens to the revenue as the network load increases. We want network load to be the independent

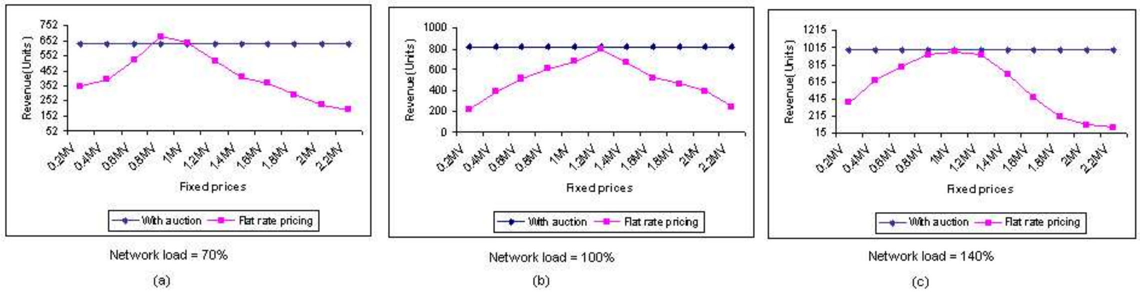


Figure 21: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 200 nodes)

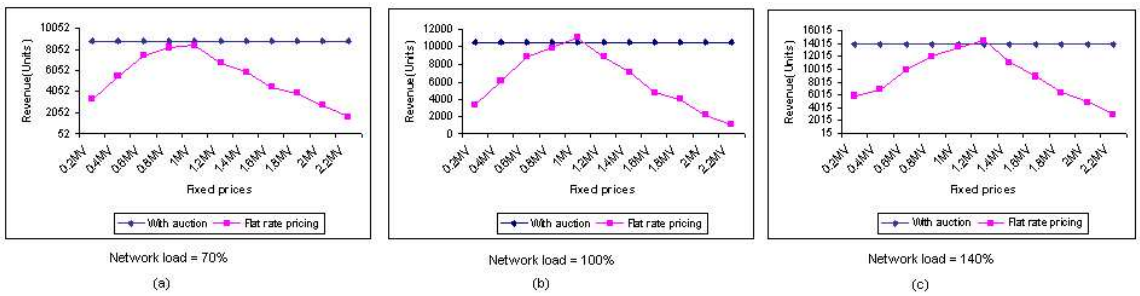
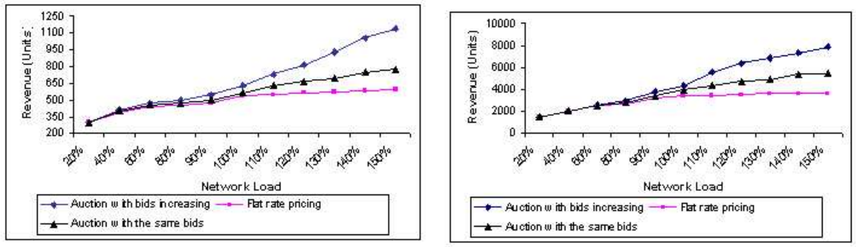


Figure 22: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 1000 nodes)

variable, but it is really dependent on the number of customers and their bandwidth demands. So we have to resort to making one of these the independent variable. We chose the number of customers because it only has one dimension. We compare three cases: 1)fixed pricing, 2) auction where customers always bid the same way, and 3) auction where customers may change their bids at each auction. The customers' valuations are normally distributed with mean of 4 mcents, standard deviation of 2 mcent for EF and mean of 3, and standard deviation of 2 for AF. The fixed price rate is 4 mcents for EF and 3 for AF and is constant for this group of experiments. We observe the revenue as the network load increases, shown in Figure 23. The network load increases along the X-axis and Y-axis records the corresponding revenue generated.



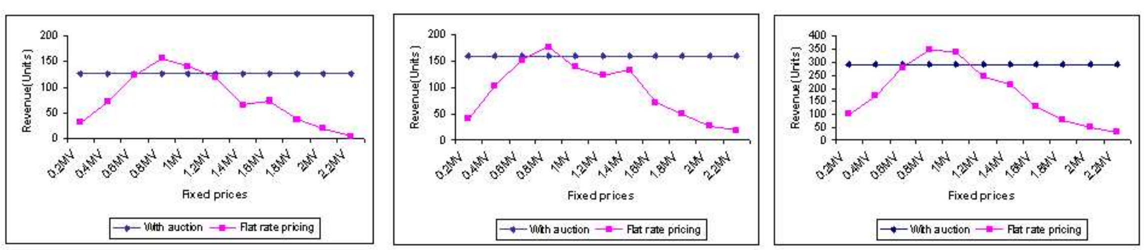
with 20 nodes in the network

with 200 nodes in the network

Figure 23: Revenue Comparison for Multiple-Link Networks (as a function of offered load)

In order to show that our algorithm does not depend on any specific settings, we vary the parameters for simulation Groups 1,2 and 3. In the following groups, we examine the different settings.

Group 4 (Figure 24, 25, 26) repeats the Group 2's procedure with customers' valuation distribution with mean value equals to the floor values. That means EF traffic has the normal distribution of 2.7 mcents mean value and 2 mcents standard deviation and AF has 2.0 mcents mean value, 1.5 mcents standard deviation. All the other parameters are same as those in group 2. We also run it on 3 different network topologies and 3 different network loads for each topology. The results are shown in Figure 24, 25 and 26. The subgraph (a)s all represent 70% network load, (b)s for 100% load and (c)s for 140%.



(a) Network Load = 70%

(b) Network Load = 100%

(c) Network Load = 140%

Figure 24: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 20 nodes)

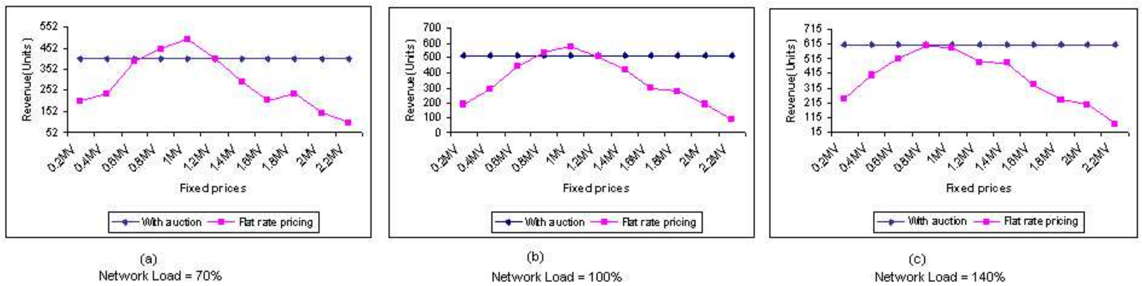


Figure 25: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 200 nodes)

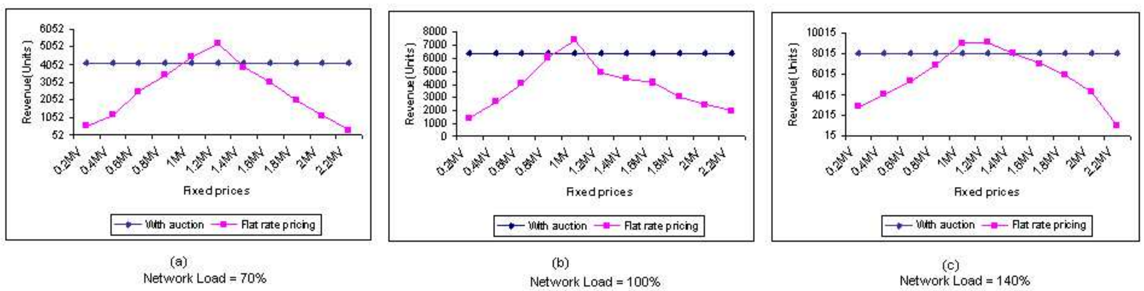
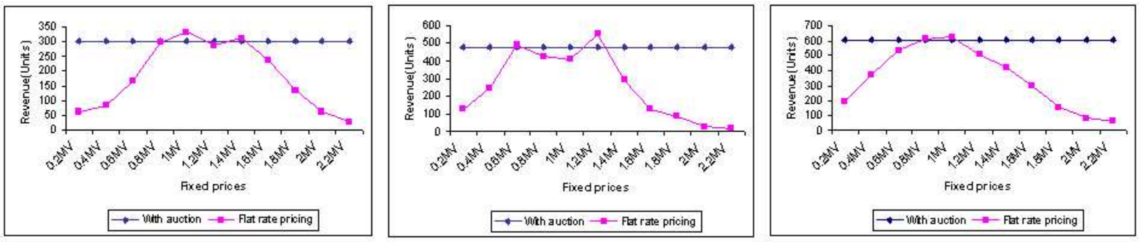


Figure 26: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 1000 nodes)

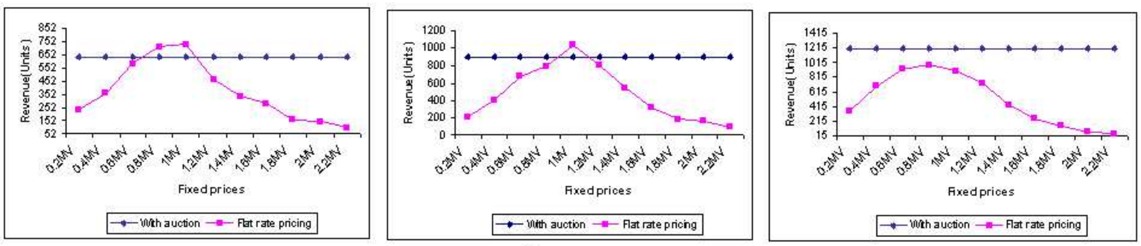
Group 5 (Figure 27, 28, 29): This group of simulations is designed to test the relativity of EF and AF price setting. We specify EF traffic costs 3 times more than BE and AF 2 times more than BE. Therefore, the floor values are changed to 4 mcents for EF and 2.7 mcents for AF. Customers' valuation is distributed as 5.4 mcents mean value, 2 mcents standard deviation for EF and 4 mcents mean and 2 mcents standard deviation for AF. Meanwhile, the demand for EF and AF is 50% - 50%.

Group 6 (Figure 30, 31, 32, 33, 34, 35): Since class EF has higher priority than AF and clients are charged more to be classified as EF, the normal demand for EF class is lower than AF. However, we cannot rule out the situation where more customers demand high quality transmission at one moment, so we also need to test how robust our algorithm is to handle different demand ratio of EF and AF. This group of simulations is designed to do auction and flat rate pricing comparison while



(a) Network Load = 70%   (b) Network Load = 100%   (c) Network Load = 140%

Figure 27: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 20 nodes)



(a) Network Load = 70%   (b) Network Load = 100%   (c) Network Load = 140%

Figure 28: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 200 nodes)

the demand for each class varies as: 30%-EF, 70%-AF (Figure 30 and Figure 33); 50%-EF, 50%-AF (Figure 31 and Figure 34); 70%-EF, 30%-AF (Figure 32 and Figure 35). The other parameters are set for Figure 30, 31 and 32 as: 1) customers' valuations are normally distributed with a mean of 4 mcents and standard deviation of 2 mcents for EF and 3 mcents with standard deviation of 2 for AF. 2)flat pricing rate is the variable within the range of (20%–220%)$\times$ MV where MV is the customers' mean valuation 3)size of the network is 200 nodes with 400-1000 customers. The different parameter settings in Figure 33, 34 and 35 are: 1) customers' valuations are normally distributed with a mean of 5.4 mcents and standard deviation of 2 mcents for EF and 4 mcents with standard deviation of 2 for AF.

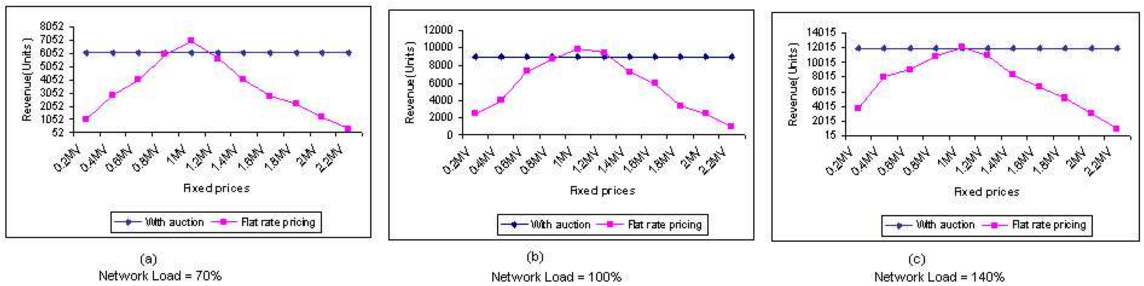


Figure 29: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 1000 nodes)

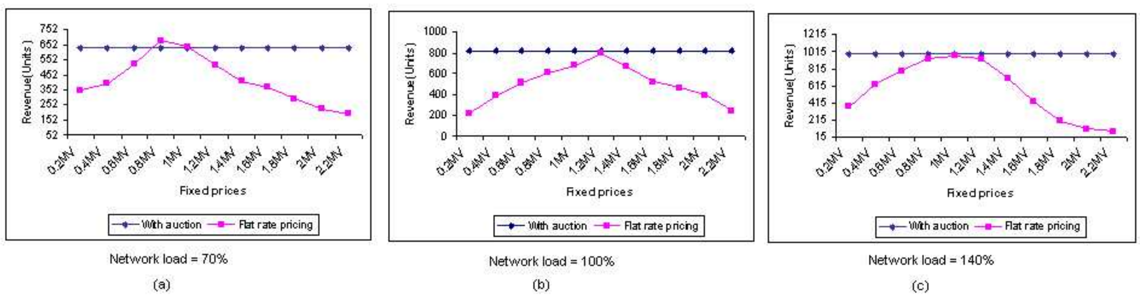


Figure 30: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 30%EF and 70%AF demands)

5.3.3 Analysis of Results

We first test auction's behavior by varying customers' valuations, while keeping the flat rate prices fixed and network load the same. When the network is not congested, there is not much difference between these two pricing approaches. The difference increases when customers really need to make sure their bids are accepted, and so they raise their bids in a non-congested network. This causes the algorithm to drop some lower bids to raise the accepting thresholds — since the thresholds come from the lowest winning bids. As the network becomes congested, auctions allow the service provider more room to make choices. As we can see from the results of Figure 17(c), 18 and 19, auctions perform better. This result gives support to the idea that auction is a valid model both for congested and non-congested networks, and it clearly beats

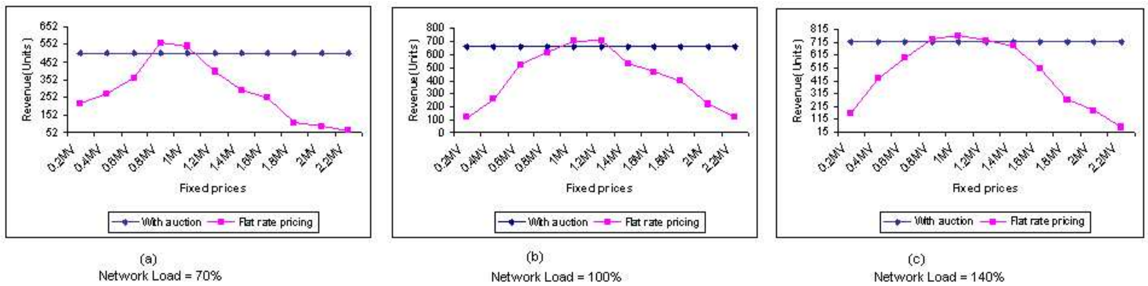


Figure 31: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 50%EF and 50%AF demands)

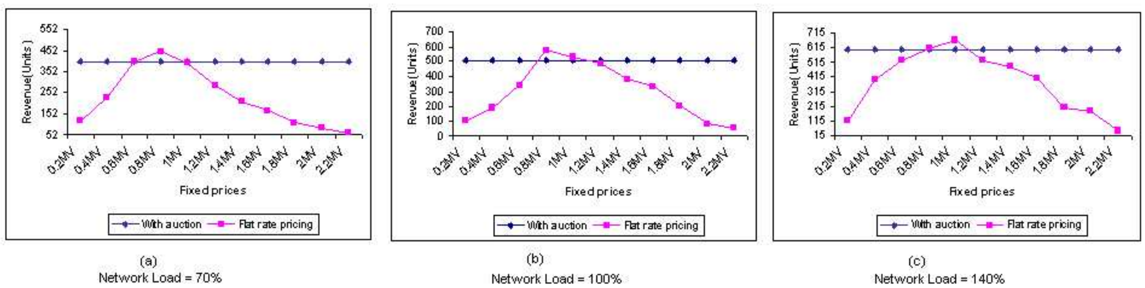


Figure 32: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 70%EF and 30%AF demands)

flat rate in the congestion case.

The second group of simulations shows that in three different load conditions, auction beats fixed pricing on average. As shown in Figure 20(b), as the fixed price rate is increasing, the revenue increases and then drops again. At the peak of this curve, flat rate can beat auction in revenue generated. Therefore, if the service provider knows the exact network load and customers' valuations, there exists an optimal price that can generate the most revenue. However, in reality because those values cannot be obtained beforehand and the time to generate a solution is limited, it is very hard to find the optimal solution in real time. Therefore, auctions are more adaptive to network conditions and generate more revenue on average. Figure 20 shows the case when the network is overprovisioned. The revenues generated by fixed price and auction are generally close to each other when the fixed price is around the mean of the

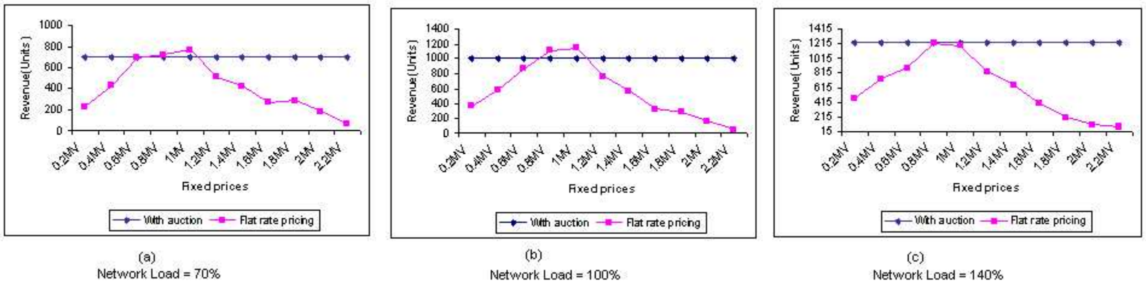


Figure 33: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 50%EF and 50%AF demands)

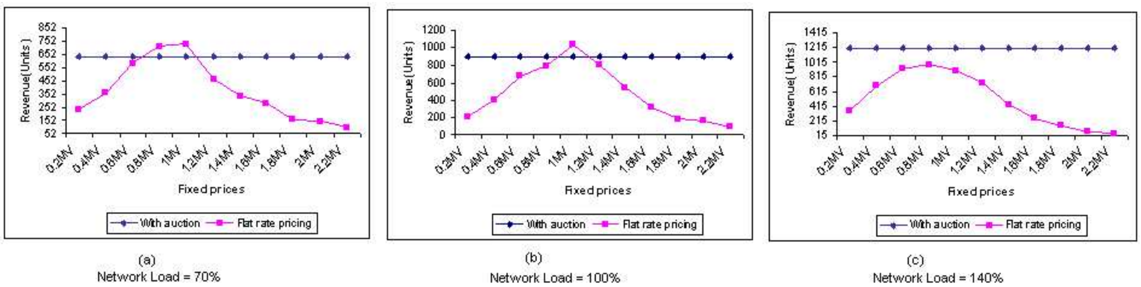


Figure 34: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 50%EF and 50%AF demands)

customers' valuation. When the fixed price rate is much lower than customers' average valuation, it loses revenue even though all clients are admitted. When the price is set much higher than the valuations, the service provider loses clients. So when the service provider does not know the customers' valuations in advance, which is usually the case, an auction provides more revenue. Figure 20(a), 21(b) and 22(b) show that as the flat rate price increases, the revenue generated by fixed price fluctuates. This is because when the fixed price increases, the number of admitted customers decreases. This is the reason that the revenue function is not linear.

The most important characteristic of the auction is that it is dynamic. So it is important to examine how a fixed price approach behaves compared to an auction when the network condition is changing dynamically. Such is the purpose of the Group 3 experiments. In these simulations, the network topology and the flat rate

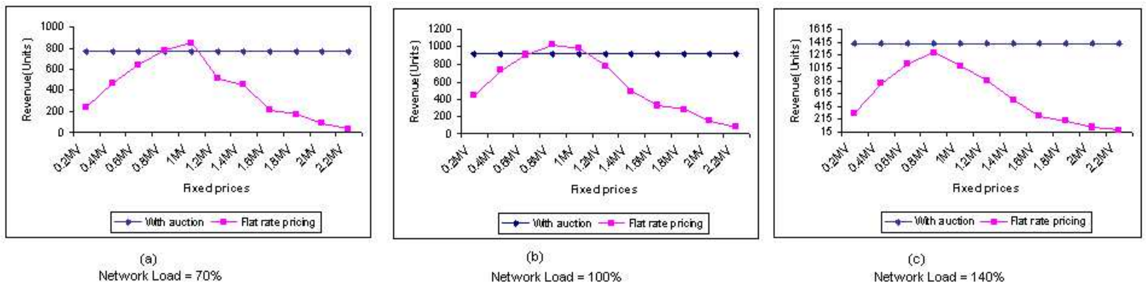


Figure 35: Revenue Comparison for Multiple-Link Networks (with fixed price as a variable and 70%EF and 30%AF demands)

prices for EF and AF are fixed, and we vary the network load. The first graph in Figure 23 shows the results for the case where the network contains 20 nodes, with hundreds of customers and the second shows the result from 200 nodes with thousands of customers in the network. It is evident that when the network is not congested, flat rate pricing and auction behave similarly. When the network is overloaded, auction performs better. Intuitively, when customers lose an auction, they may have the intention of increasing their bids in order to win in the future. To try to model this behavior, we assume that at each auction, a certain percentage of the losers intend to increase their bids in the following auction. This creates competition among customers and gives customers incentive to paying more for better services. If losers do not increase their bids, when the network load increases, they will not increase their chances to be admitted. This will also generate less revenue for the service provider. However, compared to flat rate pricing, both auction schemes give the service provider more room to make the selection in favor of more valuable customers while discarding the low bids. When there is congestion in the network, customers compete with each other for resources, which causes the price to go up an extra amount. This extra amount is what sets the auction apart from flat pricing.

The parameters we fixed for simulation Groups 1-3 are 1) floor values for each class; 2) customers' mean valuations; and 3) traffic demands for EF and AF classes.

Those are the assumptions that help to quantify the comparison between our auction algorithm and general used flat pricing. In order to show that the auction algorithm doesn't depend on those parameters, we vary them and repeat the simulations in Groups 4 and 5. Groups 4 and 5 results showed the same trends as previous ones and verify that the performance of our algorithm is not dependent on those parameters.

From the results of the experiments that we have conducted, in most cases auctions outperform the flat rate pricing in either a congested or non-congested network. Moreover, the auction algorithm works adaptively to maximizing the service provider's revenue. Meanwhile, the customers also benefit from this pricing model because they are given the freedom to express their valuation of service.

5.4 One Step Solution (OSS)

Even though IIS algorithm introduced in section 5.2 outperforms auctions in general, sometimes flat rate pricing can beat auctions when the fixed price is set at an optimal point. This means that the IIS algorithm is not an optimal solution and there is still room to improve the algorithm's performance. Next, we will present another heuristic algorithm, which can generate more revenues for the service provider.

The principle of OSS is to allocate flows into the network in order of most valuable to the worst. This algorithm differs from the IIS algorithm in that allocation is done in one step instead of two. The algorithm follows the following steps.

1. First, the algorithm assigns bandwidth to each class on every link. We again use Golden search with quadratic interpolation [49] to set the bandwidth propotion allocatd to each class. Suppose B_j^l stands for the bandwidth allocated to class j on link l.

2.
   - Build a queue S_1 for all the customers in the decreasing order of U_{ij}/L_{ij} value. S_1:$\{U_{ij}/L_{ij}, \ldots , U_{nj}/L_{nj}\}$.
   - Dequeue the first flow i in S_1.
   - Assign the flow onto each link where it crosses by the amount of L_{ij}. If there is no enough available bandwidth (i.e. $B_j^l - \sum_{k \in admitted=1}^{n} L_{kj}^l - L_{ij} < 0$, where n is the number of flows in class j which has been assigned to link l) left on any link, drop flow i. If the flow is accepted, the link's available bandwidth is reduced by L_{ij}.
   - Repeat the steps above until the queue S_1 is empty.

3. After the queue S_1 becomes empty, we calculate the bandwidth usage for all links. If any link is under utilized, we build a queue *U_link*:$\{link_i, link_j, ..link_n\}$

in the decreasing order of the gap between the bandwidth usage and actual bandwidth.

4. - Dequeue $link1$ from *U_link*.
   - On each link l, we have a queue W:$\{W_i, W_j, ..., W_n\}$ in the decreasing order of W value for the clients admitted.
   - Dequeue client i from W, if all the links that client i crosses are under utilized, increase X_{ij} (the bandwidth allocated to client i) by the minimum amount among the links' gaps (which are defined as the difference between C^l and actual allocated resource). Otherwise, repeat this step till the queue W is empty.
   - Repeat above steps until queue *U_link* is empty.

5. Calculate the revenue R_1 generated by the current assignment.

6. For the same reason that we presented in IIS algorithm, we performed the same steps as above using queue S_2 in decreasing order of U_{0ij}. Revenue R_2 is generated accordingly.

7. The assignment that generated the higher revenue of R_1 and R_2 will be chosen.

As we mentioned before, the main difference between IIS and OSS is that IIS accomplishes the assignment in two steps, while OSS does it in one step. However, the intra link part in IIS can be calculated by the links simultaneously. OSS assigns flows sequentially and is therefore slower. Suppose the number of customers is n and the number of links is m. The computation complexity for IIS is $O(m) + O(n \times m)$, while OSS runs in $O(m \times n) \times O(n)$. On another hand, because OSS adjusts all the flows' assignments while trying to allocate new clients, it is more exhaustive and usually generates better results. The intra step part in IIS provides an efficient method of allocating bandwidth on a single link. While it might not generate the best results for multiple bottleneck link network, global adjustment was designed to make the compensation. OSS does not have this concern because in each allocation, it makes inter-link adjustments.

Step 4 was designed to better utilize the network resources based on customers' bids on price sensitivity coefficient. According to our Property 1 (in Chapter 4 Section 4.1.2), because the base price predominates the price, the more customers admitted, the higher revenue the service provider receives. Therefore, steps 2 and 3 assign as many clients as is possible. Then when there is no sufficient resources for more customers, if there is still unused bandwidth we can increase revenue by assigning the rest to the clients who are willing to pay more for the extra resources allocated, i.e. customers who bid high W values. Therefore we sort the clients in the decreasing order of W in Step 4.

5.5 Simulations and Analysis

First, we want to compare OSS with flat rate pricing, as presented in section 5.3. Then we will compare IIS and OSS in terms of revenue generation.

Group 1(Figure 36): We set the parameters as following: the service provider's floor values are 2.7 mcents - EF and 2.0 mcents - AF. The ratio of EF and AF traffic is 30%-70%. The customers' valuations are normally distributed with a mean of 4 mcents and a standard deviation of 2 mcents for EF and 3 mcents with standard deviation of 2 for AF.

We perform the same simulation as in Section 5.3 Group 2, using OSS algorithm to compare revenue generation to flat rate pricing. The results are shown in Figure 36. In each graph, the X-axis stands for the fixed price variation, from 0.2MV to 2.2MV with 0.2MV interval. The Y-axis is the total revenue generated. Each graph shows the revenue generated under one set of parameters as inputs (customers, network bandwidth) using OSS compared to that generated by various flat rates.

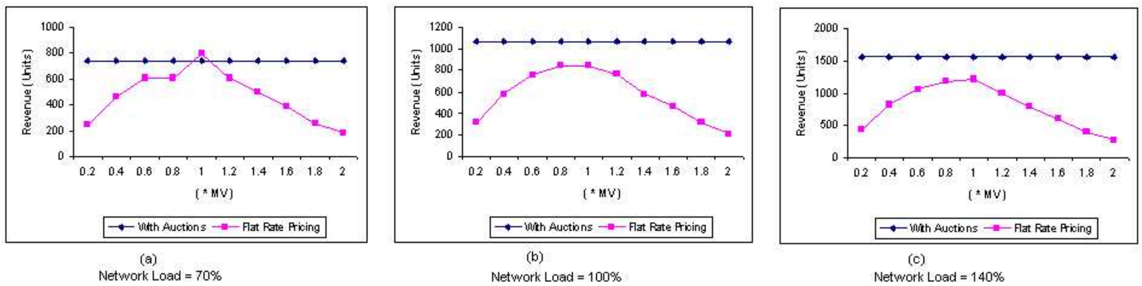


Figure 36: Revenue Comparison between OSS and Flat Rate Pricing

Group 2(Figure 37 and 38): We run the simulation using the same parameters as Group 1 by IIS and OSS. The results are shown in Figure 38, where the X-axis stands for each individual case which complies with the parameter set above and Y-axis is the total revenue generated. We also compared the execution time between IIS and OSS, as shown in Figure 38. The X-axis stands for the same cases as in Figure 37 and Y-axis gives the execution time of each algorithm.

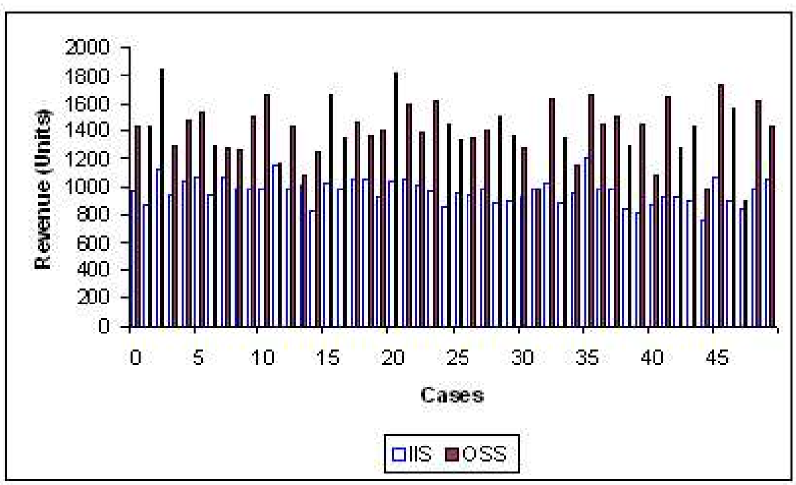


Figure 37: Revenue Comparison between OSS and IIS

The results from Group 1 show the same trend as the results from IIS algorithm, presented in section 5.3. Since the OSS algorithm is a heuristic algorithm, at certain point, flat rate pricing can generate slightly more revenue. However, OSS beats flat

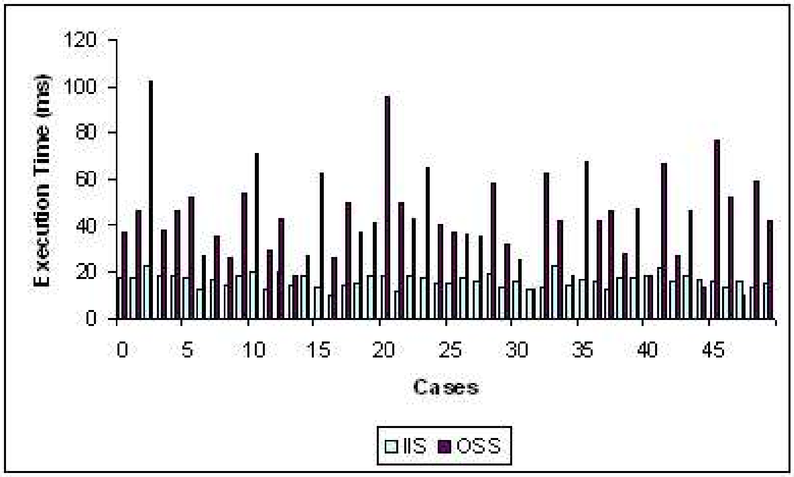


Figure 38: Execution Time Comparison between OSS and IIS

pricing at a higher percentage, compared to IIS. The outperformance of OSS to IIS can be shown clearly in Figure 37. Meanwhile, the tradeoff of using OSS is also shown in Figure 38. The service provider can choose which algorithm to use depending on the current network size and his preference. If the network size is not too large and customers' number is moderate, it might be wise to choose OSS, since it generates more revenue. But when the scale goes up, a fast calculation may be necessary, so IIS is a better choice.

5.6 Summary

Most of the work that has been done in the past only considers networks with single bottleneck link. We have considered the problem in a larger network scope, which can contain hundreds to thousands of nodes and many bottleneck links. This greatly complicates the optimization problem, which is almost certainly NP-hard.

We performed simulations of the OSS and ISS algorithms and compared them to flat pricing. The results showed that fixed pricing beats auction in revenue if the optimum price, representing the exact supply and demand balance can be selected. However, since the network changes rapidly, the best fixed price at one time could be highly sub-optimal for the next moment in time. Auctions adjust the pricing adaptively as the network changes, which makes them a better strategy in the long run. The results of our experiments show that our auction system generates more revenue than fixed pricing generated.

We also compared the performance between IIS and OSS. The results showed that IIS is more efficient in terms of time complexity but OSS generates more revenue with the tradeoff of longer execution time.

CHAPTER VI

AUCTIONS WITH DURATION PARAMETER INVOLVED

In this chapter, we introduce continuous auctions. In the previous chapters, we were only concerned with the revenue of a single maximization auction. We formulate the problem again and give possible solutions.

6.1 Statistics of how the model behaves

The theoretical solutions, algorithms and simulation results presented in previous chapters were all based on a single auction. That means each auction happens independently of any other one. At the beginning of each auction, the auctioneer does not know the results and inputs of the previous auctions. We extend the framework to include the duration of a client's application as part of a bid. Previously, we assumed that everyone was interested in the current auction and if they need more later on, they will keep bidding on it with the same bids. In the new setting, customers bid for price, resource requirement and the duration of the application. First, we assume that at the beginning of each auction, the service provider recalculates the thresholds to decide winning bids independently regardless of previous auctions. As long as the client application's duration time does not end, the client is qualified for entering new auctions, no matter if he was a winner or a loser from previous auctions. Intuitively, new auctions affect winners from previous auctions. It is likely for some clients that require duration of his application as one hour wins at the first auction, but loses the auction a minute later [1]. We want to study how often and how many clients are affected and how to solve this issue.

In order to give an overall picture on how continuous auctions behave, we conducted a series of simulations. In the following simulations, we assume the clients' mean valuation is 4 mcents for EF and 3 mcents for AF and standard deviation is 2 mcents and 1 mcents for EF and AF respectively. The duration has a mean of 1 hour and a standard deviation of 20 minutes for both EF and AF. The floor values are 2.7 mcents for EF and 2 mcents for AF. The customers' arrivals are poisson distributed. The entire simulation ran for 4 wall clock hours. An auction takes place every minute. We calculate the number of customers dropped from the network before their application completes but after winning at least one auction (denoted as *Drop*). Meanwhile we observe the number of clients who are kept in the network until they exit normally (denoted as *Normal*). We are interested in the drop rate, as *Drop/(Drop + Normal).*

Figure 39 shows the statistics of drop rate along with the time. We ran the auctions continuously for four wall clock hours. Since the auction interval is one minute, 240 auctions took place during this simulation. In Figure 39, the X-axis denotes the number of auctions. The Y-axis is the corresponding drop rate for each auction. The line shows that the drop rate fluctuates and the average rate calculated at the end of 4 hours is around 6-7%.

In our auction system, since each auction that takes place is independent of any auction that took place before, there is no guarantee that a client that was admitted in the last auction will be admitted again, if the bids submitted this round are not identical to the last round. This behavior is highly undesirable. From the results shown in Figure 39, among a thousand clients, there are on average 60-70 customers who are dropped each round after being admitted in an earlier auction. Even though

[1] We assume that the interval of auctions is one minute.

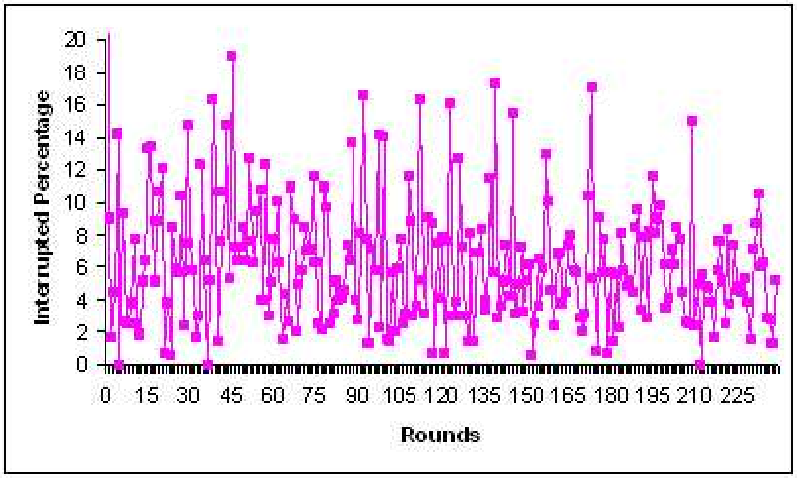


Figure 39: The Interrupted Percentages for Each Auction

the rate is not high, this statistic decreases customer satisfaction. Nobody can be assured to get admitted for any auction. Since for every auction, the service provider recalculates the bandwidth allocation based on all current customers' bids, it benefits the service provider most. The service provider does not need any traffic prediction to compute the best solution. However, consider an example where a client wants to do a video conference. Even if they have won the resource at *moment A*, they were not guaranteed anything beyond *moment A* on. If at *moment A + 1*, a new client joins the network and causes the overall price to rise. The provider may drop the video conference application and serve the new client instead. The video conference would be only up for a minute, then blackout because no more resource is allocated. This does not seem to be realistic. Therefore, we need to modify the auctioning process so that once a client wins an auction, he will be kept in the network as long as the application lasts. In general, this requirement will lower the generated revenue, but we deem it necessary in providing a reasonable quality of service.

From the literature, we found that Delta auctions [17] were designed to deal with continuous auctions. Delta auctions hold the auctions whenever a new bid arrives. This procedure reduces the end-to-end delay for reservation requests significantly.

Our continuous auction approach differs from Delta auctions in two ways. First, ours is DiffServ based where as the Delta auction is IntServ based. Second, the rate of auctions taking place would be dependent on the rate of clients attempting to join. This feature would be unscalable during peak times of network usage. Our auction models hold auctions at short constant intervals but can also accept new customers between auctions, if there is enough network resource to allocate. If the new customer is admitted, the existing customers wouldn't be affected while the service provider can still maximize his profit.

6.2 What happens when using IIS algorithm with duration added

In this section, we evaluate the efficiency of the algorithm with the change just proposed. Now clients will include a duration variable as part of their bids. The auctioneer will ensure that once a client has won an auction, he will remain in the system for the duration that the client has requested. We performed the following simulations based on this new behavior.

What interests us here is that given a time period (t_1), the difference in revenue that auctions can bring the service provider compared to the flat rate pricing. We monitor the auctions for a duration t_1. The revenue generated for the period is the sum of the revenue generated by all the auction that took place during the period.

6.2.1 Simulation Set Up

We used a similar simulation environment as we did in previous chapters. As an example, we assume that the Best Effort (BE) class is charged $35 per client per month or 0.00135 cents per minute. Let an mcent (or a unit) be equal to 1/1000 cents, so the charge for BE is 1.35 mcents per minute. Let's say EF traffic is twice as expensive as BE and AF is 1.5 times as expensive. These are the service provider's floor prices for each class. A customer who bids lower than the threshold price will be automatically rejected. The customers' valuations for EF and AF are assumed to be normally distributed. In the flat rate scenario, a customer is admitted if and only if his valuation (which is same as the bid in the auction context) is greater or equal to the fixed price set by the service provider. Revenue is the number of customers multiplied by the price.

The comparison between revenue generated from the fixed price and the auction are done by varying the set of customers (and also their valuations and durations), the network topology, the network load and the assumed fixed prices.

6.2.2 Simulation Results

We design different value patterns for the simulations, which can be classified into the following groups.

Group 1 (Figure 40 and Figure 41): In this group, the customers' valuations are distributed with a mean (MV) of 4 mcents for EF and 3 mcents for AF; standard deviation are 2 mcents both for EF and AF. Customers' application duration has a mean of 20 minutes and standard deviation of 5 minutes for both. The flat rate prices vary from .4 to 8 mcents for EF and .3 to 6 mcents for AF. Poisson arrival rate for new clients is 10 for EF and 20 for AF. Auctions are held every minute. We calculate the revenue gain at the end of one hour.

In Figure 40 the X-axis is the fixed price, from 0.1MV to 2.0MV with 0.2MV interval. The Y-axis is the total revenue generated. The graph compares the revenue generated using our IIS algorithm to those generated by various flat rates. In Figure 41, we plot the percentage increase: $(Revenue_{IIS} - Revenue_{Flat})/Revenue_{Flat} * 100\%$ against the flat price. $Revenue_{IIS}$ and $Revenue_{Flat}$ are the same as shown in Figure 40.

The overall trend is the same as appeared in section 5.3, due to the nature of

Figure 40: Revenue Comparison

the flat rate pricing. At around the clients' mean valuation range, the gap between auctions and flat pricing lessen. So we want to focus on that range and refine the flat prices. We use the formula: $U_{flat}[j] = U_{mu}[j] + factor * U_{sigma}[j]$ to vary the flat rate prices. In the equation, j stands for the traffic class and *factor* varies from -0.5 to +0.5 with 0.05 interval. $U_{flat}[j]$ is the flat rate for class j, $U_{mu}[j]$ is the clients' mean valuation in class j, and $U_{sigma}[j]$ is the clients' standard deviation for class j. The results shown in Figure 42.

Group 2 (Figure 43, Figure 44, Figure 45 and Figure 46): In this group, the customers' mean valuations are distributed with mean 5.4 mcents for EF and 4 mcents for AF; the standard deviation is 2 mcents both for EF and AF. Customers' application duration has a mean valuation of 40 minutes for and a standard deviation of 10 minutes. The auction's interval is still one minute. The revenue is calculated at the end of two hours. Figure 43 represents the scenario that revenue generated from

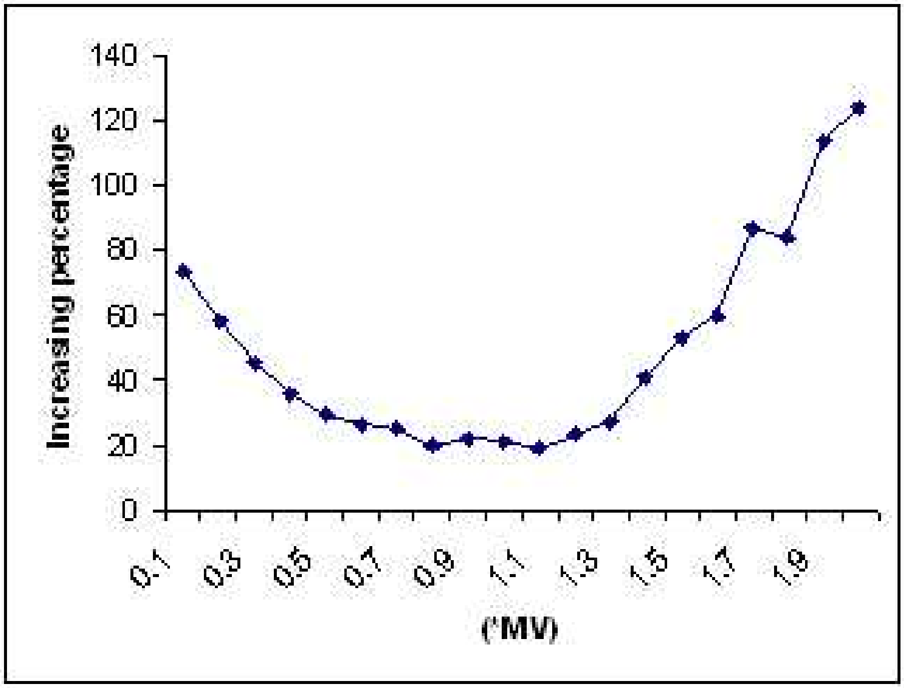


Figure 41: Revenue Increasing Percentage

one set of parameters as inputs (customers, network bandwidth) to our algorithm versus the revenue generated based on the same set of parameters but with various flat rates. We use the same formula as Group 1 to generate the flat rate prices. The factor varies between *-2* and *+1.8* in increments of *0.2*. Figure 44 gives the percentage of revenue increasing by auctions against flat rate pricing as shown in Figure 43.

Figures 45 and 46 are like Figures 43 and 44 with the difference that *factor* varies between *-1* and *0.5* with *0.02* intervals.

Group 3 (Figure 47 and Figure 48): This group has the same settings as Group 2, but the revenue is calculated at the end of 10 hours. This tests the algorithm's performance sensitivity to time duration.

The graphs are generated in the same way as Group 2.

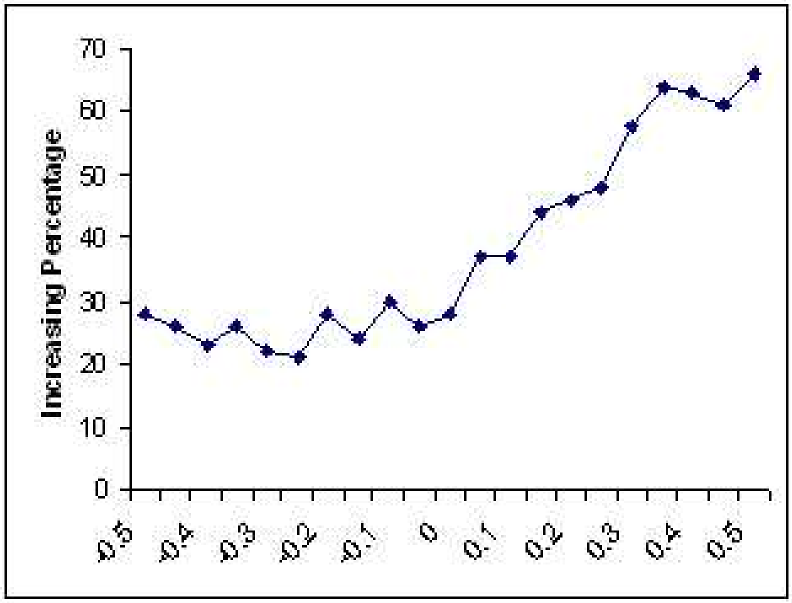


Figure 42: Revenue Comparison

6.2.3 Analysis

The results in all the groups show that our algorithm generates more revenue than flat rate. In previous chapters, when we looked at a single auction, sometimes flat rate generates more revenue than our algorithms. That's because our algorithm is not an absolute optimal solution, so when the flat prices are set at the optimal point, it can beat the heuristic algorithm. However, as we stated before, first of all it is difficult to predict and accurately calculate that perfect flat price, and secondly, when in a dynamic network, an optimal price at one moment could be a bad price at the next moment. The results above confirmed our statement. When the network is changing and the auctions are held continuously, at the end of a certain period, fixed prices always generate less revenue than our algorithm. That's because the flat rate cannot adapt to the network change. And so, over time, our adoptive algorithm always performs better.

We tested the algorithm's sensitivity to the parameters such as the customer's

Figure 43: Revenue Comparison between Auctions and Flat Rate Pricing at the End of Two Hours

mean valuations, and the auction's duration in Group 2 and 3. The results show that both groups beat flat rate cleanly.

Also the results show the same trend as what we presented previously. When the fixed prices are set too low, the service provider loses money and when it's high, customers are driven away, which brings down the revenue as well. In addition, we can see from figures that a fixed price can lose a large amount of revenue only by being different from the optimum by a small amount. This makes it even more difficult to set up a *good* fixed price.

6.3 Problems and Possible Solutions

6.3.1 Problems

From the analysis, we know that auctions always perform better than fixed prices in the long run even if we honor duration requirements. However, our IIS algorithm design didn't make use of the duration in the auction process. This shortcoming doesn't hurt the service provider's profit if the network is overprovisioned. But once the network is congested, it potentially brings down the service provider's revenue.

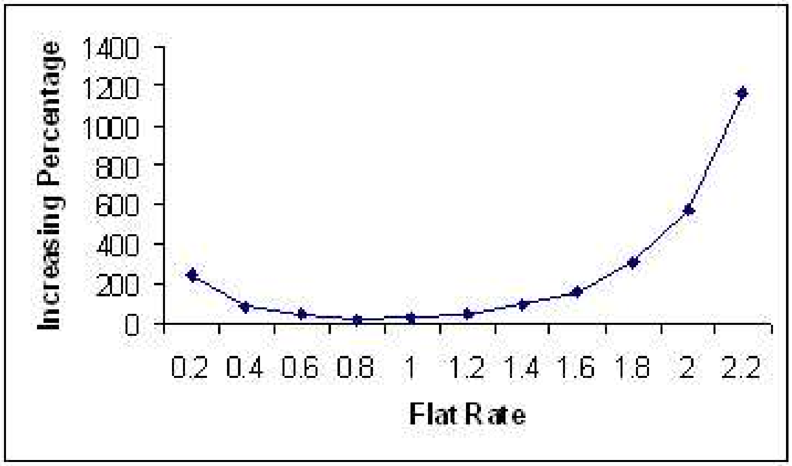


Figure 44: Revenue Increasing Percentage for Auctions at the End of Two Hours

For example, in the network shown in Figure 49. Customer 1 crosses link 1 and 2; customer 2 across link 1 and 3 and customer 3 only uses link 1. We assume that auctions are held every 10 minutes. Suppose the link capacity is 5MBPS and customer 1 requires 5MBPS for 30 minutes for \$5 per 10 minute; customer 2 bids 5MBPS for 50 minutes for \$5 per 10 minutes and customer 3 wants 5MBPS for 20 minutes for \$10 per 10 minutes. Customer 1 and 2 enter into auction at time 0 and customer 3 join in 30 minutes later. When the auction starts at time 0, customer 1 and 2 have exact the same bids other than the duration. If the auction algorithm doesn't consider duration a factor, it would assign the resource to either client 1 or 2 randomly because their bids per auction were the same. If it assigns to client 1, the service provider gets \$35 at the end, but if client 2 occupies the bandwidth, the service provider earns \$25. This shows that the duration does matter to the service provider's revenue, so we should consider it as a factor in our formulation. This also means that when the network is congested, the longer the client wants to occupy the resource, the less value the client has to the service provider.

Therefore, we need a model which depreciates the bids under congestion condition. The bidding values should not change if the network is overprovisioned. Since there

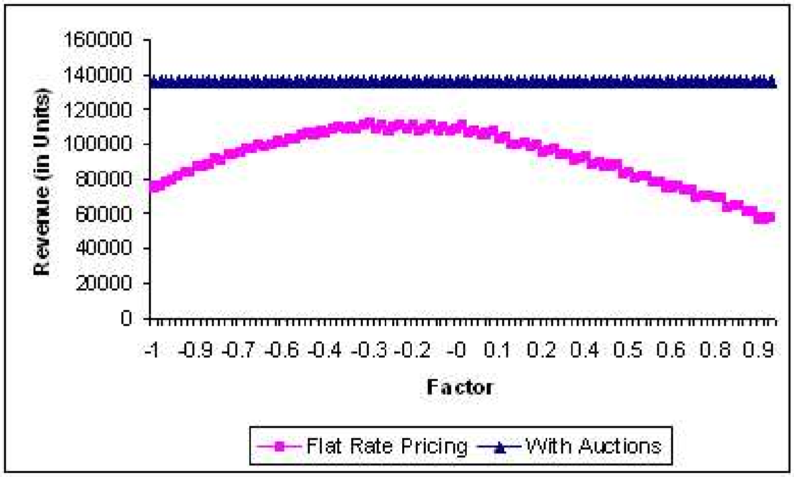


Figure 45: Revenue Comparison between Auctions and Flat Rate Pricing at the End of Two Hours

are clients who bid for more than one auction, the service provider needs to predict the traffic in the near future. We start again, with the assumption that customers' bids, including bandwidth requirements, self valuation and duration, are all distributed normally.

6.3.2 Prediction Function

Traffic predictions are traditionally performed at a single time scale using curve fitting [14] [61]. A tricky issue is choosing the number (N_h) of historical points used for current predictions: if N_h is too large, predictions tend to represent more of the past than the present, whereas if N_h is too small, predictions are made from incomplete information. Both cases lead to poor performances.

We choose an existing algorithm developed by Zhao *et al.* [61]. Zhao *et al.* presented a new approach to predicting traffic by only using traffic information in the current interval. The prediction algorithm is performed by utilizing self similarity within two adjacent intervals at sufficiently small time scales. Self similarity is measured in terms of statistical correlations between two different time scales, which is needed to predict the upper bound of future traffic volume. We use the prediction

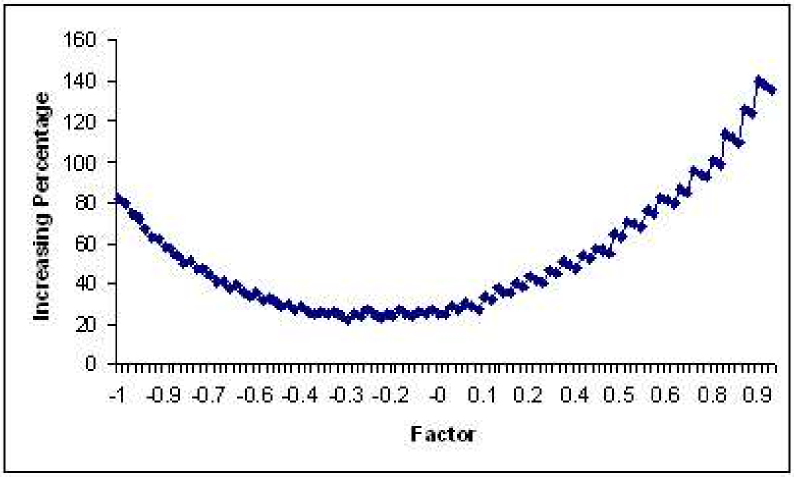


Figure 46: Revenue Increasing Percentage for Auctions at the End of Two Hours

mechanism described in [61] to predict the future traffic in the network based on current customers' bids. The prediction problem is formulated as:
Given a time scale T, the upper bound of traffic volume based on traffic information in the current interval is predicted.

6.3.3 Depreciation Function

We focus on designing an appropriated depreciation function based on the current revenue function. The depreciation should take the time duration as the parameter. The longer the required duration lasts, the more depreciated the bidding value will be. So we calculate the *real* value [2] of a client for each auction from current time on $RealValue_iAuctionj$ until the end of the application $RealValue_iAuctionm$. Add those values up and then divide by the number of auctions it will go through, which is $\sum_{n=j}^{m}(RealValue_iAuction_n)/m$. This is the actual bidding value for client i as input into our algorithm IIS.

As duration increases, customer valuation should depreciate slowly when the duration is small and faster when duration is large. The following function shows the

[2] The real value here is defined as bidding values after depreciation taken

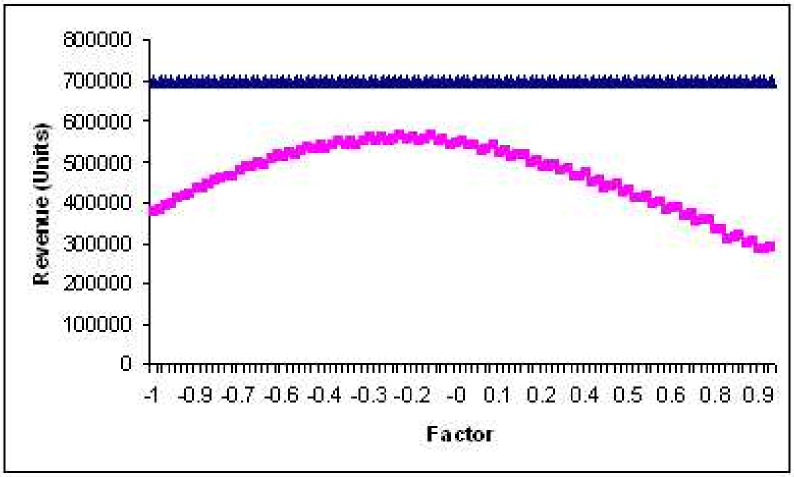


Figure 47: Revenue Comparison between Auctions and Flat Rate Pricing at the End of Ten Hours

desired characteristics.

$$U_{actual} = U_{bid}/D \times \sum_{d=1}^{D}(1/\sqrt[3]{\ln(d+2)})$$

D stands for the time duration required by the client and d the time variable.

6.3.4 Procedure

Now that we have the the prediction and depreciation function, here is the procedure for the continuous auctions, shown in Figure 50.

When the auction starts, the prediction process is triggered. The prediction algorithm tells the service provider the network congestion and chooses corresponding inputs based on the network condition. If the network is or is soon to be congested, we will apply depreciation function to customers' bids to get the real values as input to our algorithm IIS. If the network is overprovisioned, the customers' bids will be used as inputs to our algorithm. Auctions perform repeatedly.

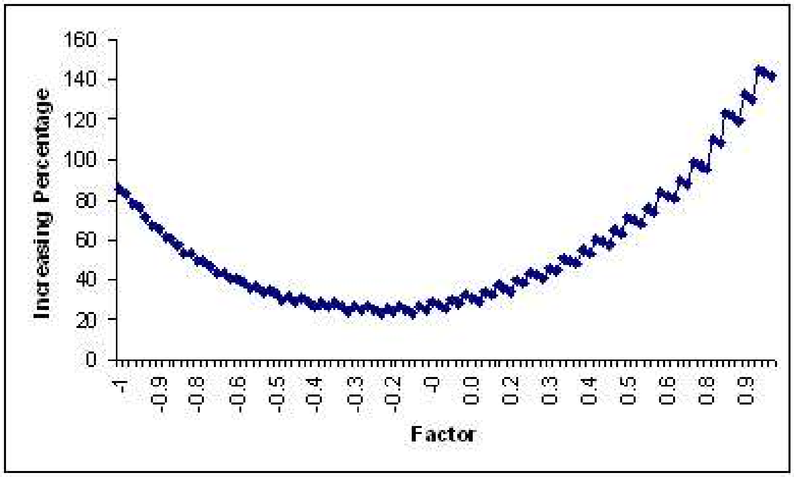


Figure 48: Revenue Increasing Percentage for Auctions at the End of Ten Hours

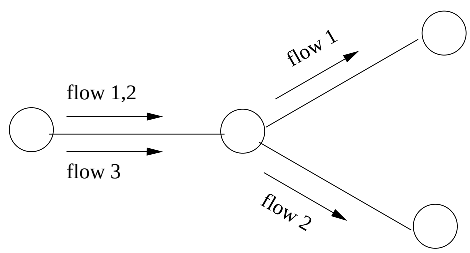


Figure 49: An Example to Show the Duration Factor

6.4 Summary

All the previous work that has been done prior this chapter was focused on individual auction and revenue maximization problem. The auctions are actually held continuously in a real scenario. We proposed three continuous auction methods in this chapter. The first one resets all the parameters at the beginning of each auction, making each auction independent of any other. The results from previous auctions are not passed to the next auction. Customers are not guaranteed anything after they are winners at one time. The second one is to keep all the winners from previous auctions until their bid on duration ends, and allocate new customers with the available resources. Both the first and second methods use the same revenue function as we

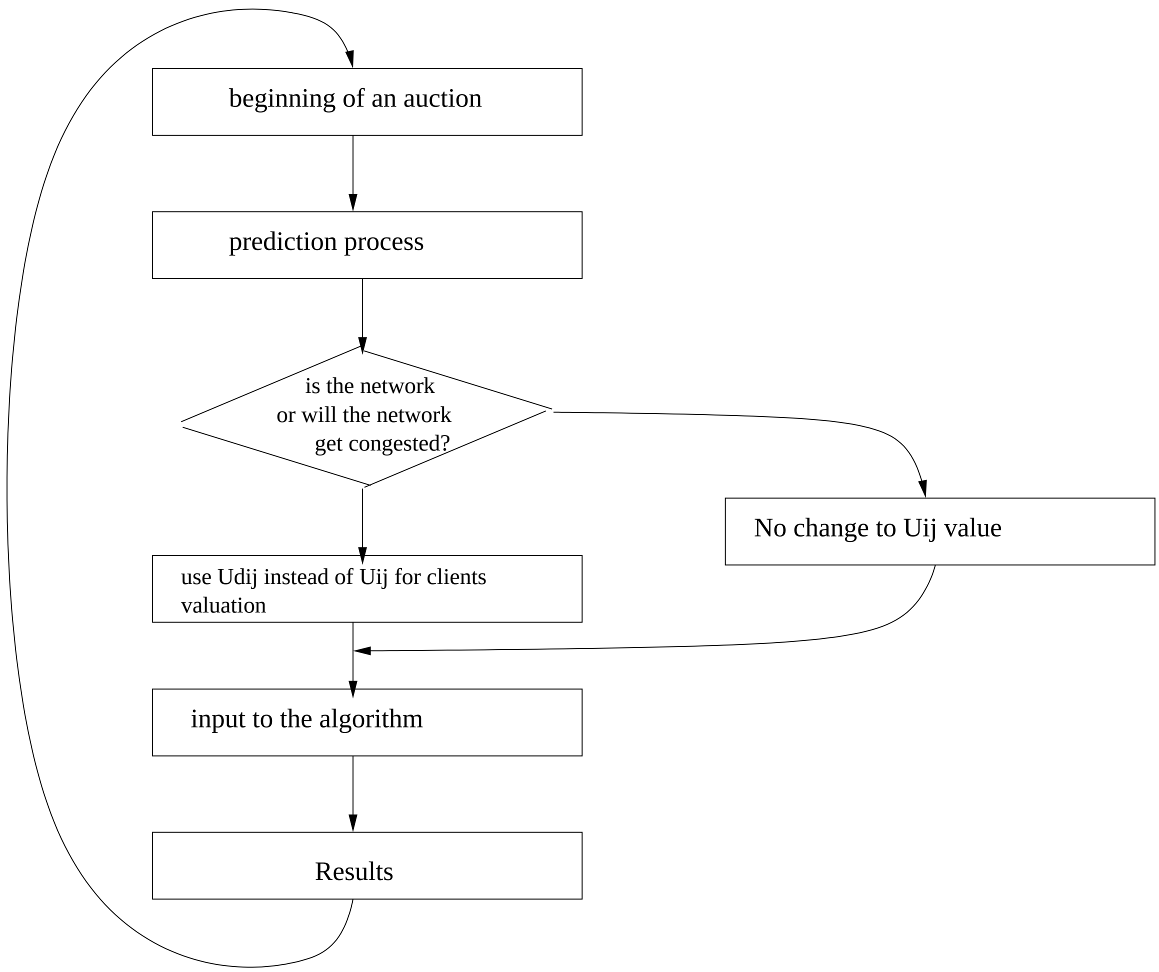


Figure 50: Continuous auction procedure

defined in Chapter 4. Customers can bid for the duration as well, but the duration bids do not alter the inputs to our auction algorithms.

The last method takes customers' bids duration as a factor in the revenue function. The new revenue function depreciates clients' values if they bid for multiple auctions, when the network is congested. The algorithms are the same as described in Chapter 5 but with the different revenue functions as inputs.

We did simulations to compare the revenue generated by continuous auction with flat rate pricing. The results shown in Chapter 5 showed that even in general our IIS algorithm beats flat rate, however, there is chance that flat rate generates more revenue when the fixed prices are set at an optimal point. Results in this chapter showed that when auctions are held continuously over time, flat rate has no chance of beating auctions. That's because auctions are more adaptive and dynamic than flat rate pricing.

Even though our algorithm outperforms flat rate, there is still room to improve the auction's performance by taking the duration bids as a factor into our revenue function. We proposed a scheme that can potentially gain more revenue for the service provider when the network is congested. The basic idea is that if the network is congested, the more time a customer requires to occupy the resources, the less value he has to the service provider, if the price bids are the same. So first we use some existing traffic prediction mechanism to tell us the traffic situation in the near future. Then based on the prediction, a depreciation factor is added into the customers' price bids. The factor can be zero, depending on the network resources usage situation.

CHAPTER VII

CONCLUSION AND FUTURE WORK

This thesis was motivated by the problem of providing differentiated QoS to clients to maximize a service provider's profit through pricing. In our model, clients are allowed to bid on the price that they are willing to pay and the service that they require. The service provider allocates the resources to each individual as well as sets thresholds for each DiffServ class based on the clients' bids, in a way that aims to maximize his revenue.

The Internet has now evolved to provide a multitude of services. The proposal known as differentiated services is a very promising architecture designed to achieve high quality of service. We use pricing as a strategy to allocate network resources in an efficient way to maximize the service provider's revenue. Among all static and dynamic pricing strategies, auctions are a widely proposed decentralized mechanism. We propose a scheme where all the clients can bid for their required bandwidth as well as the price they are willing to pay. The service provider will decide on the admission price and differentiated service provided for each class. These thresholds also provide a future reference for admitting new flows later on.

In the previous chapters, we have covered many issues regarding auctions. We dealt with the auctions in networks with a single bottleneck link in chapter 4. There are two parts of pricing bids. One is called base price which corresponds to the client's minimum bandwidth requirement. The other is price sensitivity coefficient, which measures the willingness of payment for the extra requirement beyond the minimum. In chapter 4, we fixed the base price for each service class and customers bid for the price sensitivity coefficient and minimum bandwidth requirement. An optimal solution to maximizing the service provider's revenue was presented. Then we relaxed the constraint to allow customers to bid for the base price as well. Another optimal solution was proposed to solve the service provider's revenue maximization problem. We also conducted simulations to show that auctions outperform fixed pricing.

We then further relaxed these assumptions and considered the problem in a single bottleneck link network as well as a larger network scope, which can contain hundreds to thousands of nodes and many bottleneck links. Due to the complexity of the optimization problem in a large scale network, it is certainly an NP-hard problem. In chapter 5, we proposed our heuristic algorithms and showed the results of various simulations, which were compared with the revenues generated by auction and fixed pricing. The results showed that fixed pricing beats auction in revenue if the optimum price, representing the exact supply and demand balance can be selected. However, since the network changes rapidly, the best fixed price at one time could be highly sub-optimal for the next moment in time. Auctions adjust the pricing adaptively as the network changes, which makes them a better strategy in the long run. The results of our experiments show that overall the bidding system brings the service provider more revenue than fixed pricing.

We next considered auctions held continuously instead of an individual auctions. Customers were required to bid on the duration of resource occupancy as well as the price and bandwidth requirement bids. There are two ways of handling the continuous auctions. One is to restart the calculation completely from the beginning of each auction. The other is to keep winners from one auction till the end of their applications. The first method doesn't provide any guarantees to the winning customers. The winning customers at one auction are likely to lose in the following auctions, provided the network is congested. We examined how many customers are interfered with in this case. We examined the results of how often the winners are interrupted in the first scenario. Our results showed that the percentage of interrupted customers varies as more auctions go by. Under the circumstances that we studied, the percentage range was from 2% to 18%. Even though the average percentage is not very high (at about 6-7%), the quality of the service is unacceptable.

We adopted the second method in which the winning customers are kept in the network till their applications end. Further simulations were done to compare the revenue generated by the auctions and flat rate pricing at the end of multiple auctions. The results showed that the auction algorithm always generates more revenue than flat rate pricing does in the end.

Based on what we achieved with our auction algorithms, we improved the performance further by a depreciation procedure based on application duration. The procedure improve the resource utilization efficiency by using the duration bids as factors into the revenue function.

7.1 Future Work

One of the important challenges is applying our auction approaches in a real multi-service network environment, such as the environment currently being developed to support data, video conference, and voice traffic. The approach should meet both service providers and customers demands.

The auction algorithms that we proposed have been demonstrated to outperform traditional flat rate pricing in terms of revenue generation based on bandwidth allocation. The work can be extended to include more QoS parameters in the performance evaluation such as latency and packet delay. The ability to specify additional QoS parameters gives customers added flexibility. One possible way to realize these enhancements is to add some factors in our revenue function. The general revenue function is $U_{ij} = f(desired\ price, desired\ QoS)$. Our work presented in previous chapters was based on a single bidding QoS parameter - bandwidth. Now the bidding QoS could be a list of parameters that customers would like to specify. Customers are allowed to bid on price with one or more QoS requirements.

In addition, the work can be extended to multiple service providers domain. There are network subdomains for each service provider with multiple bottleneck links. Customers can choose a service from any service provider. A network with multiple service providers introduces competition among the service providers. The providers should still be profitable, however, they will lose customers if they are not satisfied. This problem can be solved based upon our current model, with game theory as a tool.

REFERENCES

[1] W.A. and M. A. Sasse, Evaluating Audio and Video Quality in Low-Cost Multimedia Conferencing Systems *Interacting with Computers*, 8:255-275, 1996.

[2] N. Anerousis and A. A. lazar, A Framework for Pricing Virtual Circuit and Virtual Path Services in ATM Networks *ITC-15*, Dec 1997.

[3] G. de Veciana, R. Baldick, Resource allocation in multi-service networks via pricing. *Computer Networks and ISDN Systems* 30, pp951-962, 1998.

[4] T. Basar and R. Srikant, Revenue-Maximizing Pricing and Capacity Expansion in a Many-Users Regime, *Proceedings of IEEE Infocom*, 2002.

[5] Y. Bernet, J. Binder, S. Blake, M. Carlson, B. E. Carpenter, S. Keshav, E. Davies, B. Ohlman, D. Verma, Z. Wang, And W. Weiss, A Framework for Differentiated Services. IETF Internet Draft, February 1999.

[6] D. P. Bertsekas, Auction Algorithms for Network Flow Problems: A Tutorial Introduction, *Computational Optimization and Applications,* Vol. 1, pp. 7-66, 1992.

[7] Martin Bichler, The Future of e-Markets, Cambridge University Press, 2001.

[8] I.C. Paschalidis and Y. Liu, *IEEE/ACM Transactions on Networking*, Vol.10, No. 3, June 2002.

[9] D. Clark and W. Fang, Explicit Allocation of Best Effort Packet Delivery Service, *IEEE/ACM Transactions on Networking*, Vol.6, No. 4, August 1998.

[10] C.Courcoubetis, V. A. Siris, and G. D. Stamoulis, "Integration of pricing and flow control for available bit rate services in ATM networks," *Proceedings of the IEEE Globecom*, pp644-648, 1996.

[11] D. Clard, Internet Cost Allocation and Pricing *Internet Economics*, L. W. McKnight and J. P. Bailey, Eds. Cambridge, MA:MIT Press, 1997.

[12] R. Cocci, S. Shenker, D. Estrin, and L. Zhang, Pricing in Computer Networks: Motivation, Formulation, and Example. *IEEE/ACM Transactions on Networking*, 1(6):614-627, December 1993.

[13] C. Courcoubetis and V. Siris, Managing and Pricing Service Level Agreements for Differentiated Services, *Proc. of 7th IEEE/IFIP International Workshop on Quality of Service (IWQoS'99)*, Jun 1999.

[14] M. E. Crovella and A. Bestavros, Self-similarity in World Wide Web Traffic: Eveidence and Possible Causes. *IEEE/ACM Transactions on Networking*, Vol 5, No. 6, Dec 1997, pp 835-846.

[15] J. Cushnie, D. Hutchison, H. Oliver, Evolution of Charging and Billing Models for GSM and Future Mobile Internet Services, Hewlett Packard, 2000.

[16] http://www.microsoft.com/.../articulos/ 200102/art01/default.asp

[17] G. Fankhauser, B. Stiller, C. Vogtli and B. Plattner, Reservation-based Charging in an Integrated Services Network. *The Fourth Informs Telecommunications Conference*, Bocca Raton, FL USA, March 1998.

[18] X. Fu and Y. Zhang, PRM: A Resource Management Framework for Policy-driven QoS Control in Enhanced Internets. *IASTED PDCS*, 1999.

[19] J. Gong and S. Marble, Pricing Common Resources under Stochastic Demand, preprint - Bellcore, April, 1997.

[20] J. Gong and P. Sriganesh, An Economic Analysis of Network Architectures, *IEEE Network*, pp18-21, March/April 1996.

[21] A. Gupta, D. O. Stahl, and A. B. Whinston, *Priority Pricing of Integrated Services Networks*, Eds McKnight and Bailey, MIT Press, 1997.

[22] Internet Engineering Task Force. Differentiated services working group. http://www.ietf.org/html.charters/diffserv-charter.html

[23] H. Jiang and S. Jordan, Connection Establishment in High Speed Neworks, *IEEE Journal of Selected Areas in Communication*, Vol 13, No. 7, pp1150-1161, 1995.

[24] M. Karsten,J. Schmitt,L. Wolf, R.Steinmetz, An Embedded Charging Approach for RSVP, *6th IEEE/IFIP International Workshop on QoS*, Napa, California, May 1998.

[25] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, Rate Control in Communication Networks: Shadow Prices, Proportional fairness and Stability, *Journal of Operations Research Society*, Vol. 49, pp 237-252, 1998.

[26] R. P. Kelly, Charging and Accounting for Bursty Connections. In L.W. Mcknight and J. P. Bailey, editors, *Internet Economics*, MIT Press, 1997.

[27] K. Kilkki et al, Internet Charging Reconsidered. *5th Annual Networld and Interop Engineers Conference*. Las Vegas, May 1998.

[28] Y. A. Korilis, T. A. Varvarigou, and S. R. Ahuja, Incentive-Compatible Pricing Strategies in Noncooperative Networks, *IEEE Infocom*, 1998.

[29] C. Lambrecht and O. Versheure, Perceptual Quality Measure Using a Spatio-Temporal Model of Human Visual System *Proc. of IS&T/SPIE*, Feb 1996.

[30] R.R.-F. Liao and A. T. Campbell, "Dynamic core provisioning for quantitative differentiate service," *Proceedings of the International Workshop on Quality of Service*, 2001.

[31] S. H. Low and D. E. Lapsley, "Optimization flow control - I:Basic algorithm and convergence," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 861-875, 1999.

[32] P. Marbach, "Priority Service and max-Min Fairness," *Proceedings of IEEE Infocom*, 2002.

[33] J. F. MacKie-Mason and H. Varian, Pricing the Internet. In B. Kahin and J. Keller, editors, *Public Access to the Internet*, Prentice-Hall, Englewood Cliffs, NJ, 1994.

[34] McAfee, R. and McMillan, P.J. Auctions and bidding. *Journal of Ecnomic Literature*, 25, pp699-738, 1987.

[35] M. Morrow and K. Vijayananda, Developing IP-Based Services, Morgan Kaufmann Publishers, 2003.

[36] Murphy, L., J. Murphy, Bandwidth allocation by pricing in ATM networks. *Porc. IFIP Broadband Comm.*, 1994.

[37] N. Semret, Market Mechanisms for Network Resource Sharing, Ph.D thesis, Columbia, 1999.

[38] K. Nichols, V. Jacobson and L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet", Internet Draft, ¡draft-nichols-diff-svc-arch-02.txt¿. Apr. 1999

[39] K. Nichols, et al., "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", *RFC 2474*, Dec. 1998.

[40] J. Nocedal and S. J. Wright, "Bandwidth allocation by pricing in ATM networks,", in *ITC*, June 1995.

[41] Parris, C., D. Ferrari, A resource based pricing policy for real-time channels in a packet-switching network. Technical report. International Computer Science Insititute, Berkeley, CA, 1992.

[42] D. Monderer and M. Tennenholtz, K-Price Auctions: Revenue, Inequalities, Utility Equivalence, and Competition in Auction Design *Games and Ecnomic Behavior*, Vol 31, 2000.

[43] I.C.Paschalidis and J.N.Tsitsiklis, Congestion-Dependent Pricing of Network Services. *IEEE/ACM Transactions on Networking*, 8(2): 171-184, April 2000.

[44] R. Rajan, E. Celenti, and S. Dutta, Service Level Specification for Inter-domain QoS Negotiation, draft-somefolks-sls-00.txt, Internet Draft, Nov 2000.

[45] P. Reichl, G. Frankhauser, B. Stiller, Auction Models for Multi-Provider Internet Connections *Proceedings of IEEE Infocomm*, July 2002.

[46] X. Wang and H. Schulzrinne, RNAP: A Resource Negotiation and Pricing Protocol, *International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, pp 77-93, 1999.

[47] Salsano et al. Definition and Usage of SLSs in the AQUILA Consortium, draft-salsano-aquila-sls-00.txt, Internet Draft, November 2000.

[48] N. Semret, R. Liao, A. T. Cambell and A. A. Lazar, "Pricing, provisioning, and peering: dynamic markets for differentiated internet services and implications for network interconnections," *IEEE Journal of Selected Areas in Communications*, vol. 18, No. 12, December 2000.

[49] J.N. Sidal, *Optimal Engineering Design: Principles and Applications*, Marcel Dekker, New York, NY 1982.

[50] B. Stiller, P. Reichl, S. Leinen, A Practical Review of Pricing and Cost Recovery for Internet Services, NETNOMICS - Economic Research and Electronic Networking, Vol 3, No. 1, March 2001.

[51] P. Thomas, D. Teneketzis, J. K. Mackie-Mason, Ä Market-based Approach to Optimal Resource Allocation in Integrated-Services Connection-Oriented Networks: *IEEE Opations Research* 50, pp 1-14, 2002.

[52] P. P. White, "RSVP and Integrated Services in the Internet: A Tutorial",*IEEE Comm.* Mag. 35(5), May 1997, pp. 100-6

[53] J. Wroclawski, "The Use of RSVP with IETF Integrated Services",*RFC 2210*, Sep. 1997.

[54] X. Wang and H. Schulzrinne, "RNAP: A resource negotiation and pricing protocol," in *International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, 1999, pp 77-93.

[55] X. Wang and H. Schulzrinne, Pricing Network Resources for Adaptive Applications in a Differentiated Services Network, *Proceedings of IEEE Infocom*, 2001.

[56] W. Yang, H. Owen and D. Blough, An Auction Pricing Strategy for Differentiated Service Networks, *Proceedings of IEEE Globecom*, GC 21-7, San Francisco 2003.

[57] M. Yuksel, S. Kalyanaraman, Effect of pricing intervals on the congestion sensitivity of network service prices. *Proceedings of IEEE Infocom*, 2001

[58] A. J. O'Donnell and H. Sethu, "A Novel, Practical Pricing Strategy for Congestion Control and Differentiated Services," *Proceedings of ICC* 2002.

[59] A. A. Lazar and N. Semret, "Auctions for Network Resource Sharing CTR Technical Report CU/CTR/TR 468-97-02", Columbia University, February 11, 1997.

[60] E. W. Zegura, K. Calvert and S. Bhattacharjee. How to Model an Internetwork *Proceedings of IEEE Infocom* 1996, San Francisco, CA.

[61] Weibin Zhao and Henning Schulzrinne, Predicting the Upper Bound of Web Traffic Volume Using a Multiple Time Scale Approach, *International World Wide Web Conference/ (WWW'03)*, Budapest, Hungary, May 2003.