# Congestion-Dependent Pricing for a Service Provider

by

Piyush Bharti

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degrees of
Bachelor of Science in Electrical Engineering and Computer Science and
Master of Engineering in Electrical Engineering and Computer Science
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1, 1999



Author .....
Department of Electrical Engineering and Computer Science
June 1, 1999

Certified by .....
John N. Tsitsiklis
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Departmental Committee on Graduate Theses

# Congestion-Dependent Pricing

# for a Service Provider

by

Piyush Bharti

Submitted to the Department of Electrical Engineering and Computer Science

June 1, 1999

In Partial Fulfillment of the Requirements for the Degrees of
Bachelor of Science in Electrical Engineering and Computer Science and
Master of Engineering in Electrical Engineering and Computer Science

## ABSTRACT

A service provider who provides users access to a communication network is constrained in the system by the limited amount of bandwidth that can be offered to users. We will discuss congestion-dependent pricing, a method of pricing connection access to the network that accounts for this bandwidth constraint. Different models for the system are analyzed and simulated. The case of multiple classes of users as well as the case of a probabilistic demand function are analyzed in detail. The dynamic congestion-dependent pricing policy that maximizes performance will be determined using dynamic programming. The steady state revenue generated for the service provider will be used as the measure of system performance. Additionally, approximation and estimation techniques to simplify analysis and implementation of different systems are analyzed.

Thesis Supervisor: Prof. John N. Tsitsiklis

# ACKNOWLEDGMENTS

I would like to thank everyone who has helped me over this last semester with this thesis and those who have made the past five years a memorable experience.

I would like to thank my thesis supervisor, Professor Tsitsiklis. His intelligence, insight, and patience have been inspiring. The opportunity to learn from such an amazing professor was one of the reasons that I chose to attend MIT in the first place. Working with him over the past six months has been exactly the type of experience that I was hoping for. Waiting until December to find the right supervisor is usually a bad idea, but in this case it worked out for the best.

I would like to thank my parents for their love and support. Without it, I wouldn't have survived the five years. They gave me the self-confidence and guidance that I needed at the right times. Also, my sister Nita's sense of humor and constant e-mails provided a much-needed occasional laugh.

As for my friends, I would like to thank Joyce for all her help. Thanks to Tameez, for providing hours of distractions to keep me sane. Thanks to my roommates, Donald, Charles, and Irene who helped me find hours of time to be unproductive and who helped take thousands of phone messages while I was out (sorry about that). Also, thanks to all my other friends who I have run across on my journey through MIT. When I remember MIT I think of all those good times we've had throughout these years. Lastly, I would like to thank Basil the rabbit and Sparky the dog, who I doubt will care that I've mentioned them here.

TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

*Number* *Page*

*Chapter 1*

# INTRODUCTION

We propose to formulate different pricing strategies for a service provider that provides access to a communications network. An example of a communications network would be a network that provides users access to the Internet. The service provider provides bandwidth to users to connect to the infrastructure of the Internet. However, the service provider has a limited amount of bandwidth that it can offer its users. In certain cases (especially the dominant flat monthly fee pricing scheme currently used by most internet service providers), the service provider can experience congestion caused by too many users simultaneously demanding access to the network. This can result in a loss in quality of service or the inability to provide any services to incremental customers. The flat fee price scheme results in an inefficient allocation of limited resources among users and an inefficient revenue maximization policy for the service provider. Congestion-dependent pricing, which is analyzed in this thesis, is a method the service provider can use to set prices taking the bandwidth constraint into consideration.

Different pricing strategies can be developed to optimize the performance of the system. Two possible performance measures are revenue generated for the service provider and total social welfare generated for the system. In this thesis, revenue will be used as the performance measure for the most part. However, a strategy that increases revenue can

sometimes also allocate resources more efficiently, increase social welfare, and reduce congestion in the system. An analogous system that uses total social welfare as the performance measure can be constructed in a similar manner to a system that uses revenue as the performance measure.

The system becomes congested because the service provider has a limited amount of bandwidth that it can provide its users. By setting prices based on demand for bandwidth and the amount of congestion in the system, the service provider can manage its limited resource of bandwidth more efficiently. This can result in an increase in user service quality (by reducing congestion in the system), a more efficient allocation of resources among users, and an increase in revenue.

**Example: A situation where congestion pricing would be useful**

An example of a situation where a poor pricing strategy resulted in problems for a service provider is the case of America Online in early 1997. The service provider had a flat fee per month that enabled users to have unlimited Internet access. The system had a large number of users and not enough bandwidth to provide users during peak usage hours. As a result, it was difficult to connect to the system during peak hours and the system was slow. Also, a flat fee price strategy results in an inefficient allocation of bandwidth during peak hours. The perceived service quality suffered and the brand image of the firm, a large selling point, deteriorated rapidly.

As a result of this, America Online eventually changed their pricing scheme to a flat fee in addition to a connection time fee. The firm also drastically increased the amount of available bandwidth in the system. The change in the pricing scheme may have helped to reduce some of the congestion during peak hours. However, a congestion-dependent pricing strategy may have helped to further reduce congestion during peak hours. Also, recognizing congestion trends in the system might have allowed the America Online to change pricing strategies or increase bandwidth earlier.

After the large increase in bandwidth available to users, America Online went back to charging users a flat-fee price per month. Although the revenue per customer is approximately the same as it was during the connection time fee price strategy, the average usage has gone up from about 5-6 hours per user per month under the old pricing policy to 20 hours under the flat fee pricing policy. America Online expects a large amount of growth in demand and is therefore currently not worried about the expenses of setting up a system with a large amount of excess bandwidth. However, as the growth rate slows, the firm will need to revisit its pricing policy in order to remain competitive.

**Justification for Research**

As the government reduces Internet funding, it is inevitable that bandwidth will become an expensive commodity for service providers and that some method of more direct pricing of bandwidth for users will be necessary. There are several proposals to price Internet services by charging for sending packets of information on the Internet, such as Paris Metro Pricing, tariffs, and auction protocols. If one such pricing scheme does become

a reality, service providers will have to pass these costs onto consumers. Congestion pricing may be a simple and effective method to do so.

As the industry matures and competition increases, firms will be forced to find a more efficient means of managing bandwidth other than increasing the total available bandwidth in the system. Service providers will compete against one another based on price, types of services provided, and quality of service. In a mature competitive environment, increasing bandwidth to compensate for congestion at peak usage hours while maintaining a flat fee pricing policy will result in an excessively large, high cost system.

**Objectives**

There are two main objective we hope to accomplish:

1. *To develop an optimal pricing policy that maximizes performance for a network under various system models.*

We hope to build and analyze different system models in order to understand how systems with different assumptions react. The optimal pricing policy for each model will be determined using dynamic programming. This policy assumes we have full information about the state of the system and the system parameters. The optimal pricing policy will be a dynamic congestion-dependent pricing policy. A dynamic policy means that the price charged an arriving customer depends on the current level of congestion in the system. If a

user enters the system when it is highly congested, they will be charged a higher price compared to a user who enters the system when the system is empty.

The models will also be built and simulated in a system dynamics application to determine how well the optimal pricing policies calculated by the dynamic programs work in simulation and how well approximations for the optimal pricing policies work.

There are two main differences between the various models:

*a) Segmenting the user population:*

*The single user-class assumption versus the multiple user-classes assumption.*

If a service provider models the entire user population as being homogenous and all users have the same bandwidth requirements, demand function, and departure rate, the system can be modeled as a single user-class system. However, a service provider may offer different products and services or may segment the user population into different categories. In this case, each class has its own characteristics but they all use the common limiting resource of bandwidth. A multiple user-class model was built to deal with this assumption. In a multiple user-class model, each user-class has its own bandwidth requirements, demand function, and departure rate. This setting has been analyzed in the work of Paschalidis and Tsitsiklis [7]. In this thesis, we hope to extend some of the work done in their paper.

*b) Fixed and probabilistic demand functions:*

*The single state assumption versus the multiple state assumption.*

If the demand function for a user-class is assumed to be fixed and does not change over time, the system is called a single state system. However, the demand function may vary over time. This can occur from the system being shocked occasionally by a large increase or decrease in the demand rate or could simply occur from the demand rate drifting over time. This assumption causes the demand function to become a probabilistic function. The probabilistic function is modeled as a multiple state Markov chain in this thesis.

2. *To develop insight from the optimal pricing method that a service provider could use in developing a practical congestion-dependent pricing policy.*

As we will show, the optimal pricing policy is a dynamic policy that is complicated to calculate and may be difficult to implement. Various approximation and estimation techniques can be used for the system model, the system parameters, and the pricing policy in order to make systems easier to implement.

An example of an approximation is to use a static congestion-dependent policy instead of a dynamic policy. A static congestion-dependent policy sets the price constant regardless of how many users are currently in the system. Therefore, a user who enters the system when there are several users will be charged the same price as a user who enters the system when it is empty. This assumption is analyzed by Paschalidis and Tsitsiklis and

shown to be an accurate approximation that causes only a modest loss in performance, but simplifies the implementation significantly. [7]

Similar to this example, other approximation and estimation techniques will be evaluated in this thesis. One such estimation is determining which Markov state the probabilistic demand function of a multiple state system is in. The dynamic pricing policy assumes we have full information about the system but in practice, the current Markov state of the demand function may be unknown. Estimation techniques will be determined, simulated, and analyzed for performance in different multiple state systems.

**Layout**

There are four additional chapters. The next section (chapter 2) will introduce the various models and assumptions in detail and the dynamic programming algorithms used to solve for the optimal dynamic congestion-dependent pricing policies. Chapter 3 will introduce the results from the dynamic programming algorithms and the system dynamic simulations for the different models. It will also note interesting observations about the systems and analyze some of the approximations for the systems. Chapter 4 will discuss estimation techniques for single state and multiple state systems where the service provider does not have full information about the system. Finally, we conclude by reviewing what we have accomplished, by stating what approximation and estimation techniques make implementation easier, and by discussing possible areas for additional research.

*Chapter 2*

# DESCRIPTION OF THE MODEL

Different models were built to deal with different assumptions about the system. This chapter will describe what assumptions were made and how the assumptions were explicitly designed for each model. The three main models discussed are the single state and single user-class model (SS model), the single state and multiple user-class model (SM model), and the multiple state and multiple user-class model (MM model).

For each model, the optimal dynamic pricing policy to maximize performance was developed. To determine the optimal dynamic pricing policy, continuous time, controlled Markov chain models were built for each of the models and solved via dynamic programming. The measure used for performance was the steady state revenue rate generated for the service provider.

## Single state and single user-class model (SS model)

The SS model assumes that all users belong to a single user-class. All users have the same demand function, bandwidth requirement, and departure rate. Additionally, the demand function is constant through time.

*Customer arrival rate (Demand Curve)*

Users enter the system according to a Poisson arrival process with a controlled arrival rate. The instantaneous arrival rate $\lambda$, is a function of the current price, $u$ that the

service provider charges. The price is assumed to be a connection-fee charge that users pay when they connect to the network. The function $\lambda(u)$, is assumed to be linear in $u$, of the form,

$$\lambda(u) = \max\{\lambda_0 - \lambda_1 u, 0\}, \tag{1}$$

where $\lambda_0$ and $\lambda_1$ are positive constants and the price, $u$, is nonnegative. As the price for access to the network increases, the arrival rate decreases. Beyond a threshold price of $u = \frac{\lambda_0}{\lambda_1}$, there will be no additional users entering the system. Although there is currently no data to support a linear demand curve for communication services, it makes sense to use a curve where user demand is monotonically decreasing with respect to price.

It is assumed that the service provider is a monopolist and can set prices independently of competition. This is a simplification of the actual system, where at least some amount competition exists. In reality, both the average price charged to users and the quality of service will affect the competitive environment of the service provider in the long run.

*User acceptance into the network*

It is assumed in the single-class model that each user demands the same amount of bandwidth. Thus, the maximum number of users the network can accommodate, $N$, can be determined. $N$ will be equal to the truncated integer value of the total bandwidth, $R$, divided by the bandwidth demanded per user, $r$,

$$N = \text{int}\left[\frac{R}{r}\right]. \tag{2}$$

The system can keep admitting users into the network until the number of users, $n$, is equal to the maximum number of users the system can accommodate, $N$. If additional arrivals occur beyond this point, the users who are not admitted into the system do not enter a queue but are lost. This is also known as balking in queuing theory.

*User departure rate*

The holding time of each user is assumed to be exponentially distributed and independent. The exponential parameter, $\mu$, is assumed to be a constant. The expected time that an individual connection lasts is $\frac{1}{\mu}$. Although there is no data to show that the departure rate for communication services is exponential, the assumption makes modeling a system with several users who arrive at different times much simpler because the exponential distribution is memoryless. Determining the rate of departures from the system at any time only depends on the exponential parameter, $\mu$, and the number of users, $n$, currently in the system. In addition, telephone calls, which are in some ways analogous to these types of communication services, have been shown to be exponential.

*Revenue*

The service provider accumulates revenue through the price paid every time a user connects to the network. One method of setting prices for a service provider would be to try to maximize the expected revenue per unit time, J*. An upper bound on the maximum

revenue per unit time can be found by considering a system that has an infinite amount of bandwidth *R*. For such a system the optimal price, *u*, will be constant, regardless of how many users are already in the system. In steady state, the expected revenue per unit time will be $J^*=u\lambda(u)$. This quantity can be maximized by setting its derivative with respect to *u* equal to zero and solving for *u*.

$$J^*_{R\to\infty} = u\lambda(u) = u\lambda_0 - u^2\lambda_1, \tag{3}$$

$$\frac{dJ^*}{du} = \lambda_0 - 2u^*\lambda_1 = 0, \tag{4}$$

$$u^* = \frac{\lambda_0}{2\lambda_1}, \tag{5}$$

$$\lambda(u^*) = \frac{\lambda_0}{2}, \tag{6}$$

$$J^*_{R\to\infty} = \frac{\lambda_0^{\,2}}{4\lambda_1}. \tag{7}$$

These values for $J^*$ and $u^*$ (for a system with infinite bandwidth) will accurately approximate finite bandwidth systems where congestion is not a problem. These are systems where the steady state arrival rate is much less than the maximum departure rate.

$$J^*_R \approx J^*_{R\to\infty} \quad \text{for } \lambda(u^*) << N\mu, \tag{8}$$

$$J^*_R \approx J^*_{R\to\infty} \quad \text{for } \frac{\lambda_0}{2} << N\mu. \tag{9}$$

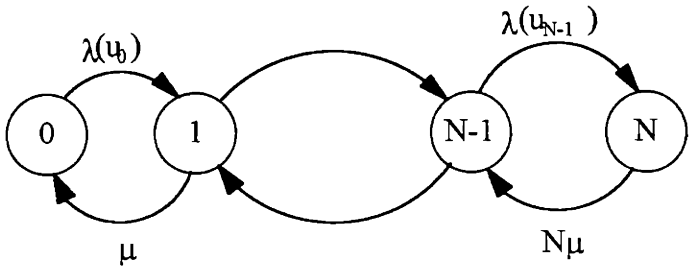
If $\frac{\lambda_o}{2} \geq N\mu$, the J* calculated for a system with infinite bandwidth becomes a poor approximation for the revenue per unit time of the congested system. To maximize revenue in a system with finite available bandwidth, the optimal pricing strategy is a function of the state of the system. This is called dynamic congestion-dependent pricing. In a system that can support a finite number of users, dynamic congestion-dependent pricing says that the user who enters the system when it is relatively empty will be charged a different price than the user who enters the system when it is relatively full.

To maximize revenue it is now necessary to set a vector of prices, $\mathbf{u^*}=(u_0, u_1, .., u_N)$, where $u_n$ is the price charged a new user entering the system when there are n previous users already in the system.

*Modeling the Markov chain*

The SS system can be modeled as having memoryless arrivals, memoryless service times, and N servers, where there is balking. This is called an M/M/N queue with balking. A continuous-time Markov chain can be constructed to model the system. The states in the Markov chain are the number of users in the system at one time. The system will have N+1 states (0,1,2,...N) and will transition as shown in the Markov chain pictured below.

Figure 1: Markov chain diagram for a single state, single user-class system.




The maximum transition rate out of any state is bounded by ν,

$$\nu = \max(\lambda(u)) + \max(n\mu) = \lambda_0 + N\mu. \tag{10}$$

The maximum possible arrival rate is $\lambda_0$ (which corresponds to a price of $u$=0) and the maximum departure rate occurs when the system is in state $N$, and is equal to $N\mu$.

*Maximizing revenue: Formulation of the dynamic program*

The optimal price vector, $\mathbf{u}^*$, can be found by solving a Bellman equation. The Bellman equation and different methods of solving Bellman equations can be found in chapter five of [2]. The general form of the Bellman equation is,

$$J^* + h(n) = \max_{u_n}\left[\lambda(u_n)u_n + \sum_{j \in neighbors(n)} P_{n,j}h(j)\right]. \tag{11}$$

In this equation, the set *neighbors(n)* consists of all the states that the system could transition to from state *n*. This list includes state n as well. For this system, *neighbors(n)* can only include *n*-1 (if $n>0$), *n*, and *n*+1(if $n<N$). The variable h(*n*) indicates the relative reward of being in state *n*. This variable h(*n*) will be highest for $n=0$ and lowest for $n=N$ because the relative reward is highest when the system is empty and lowest when it is full.

The maximum transition rate from any state in this system is bounded by ν. The probability of transitioning from the current state, *n*, to another state, *j*, is $P_{n,j}$. This probability can be defined as the rate of transitioning to another state divided by the maximum transition rate (ν). The probability of transitioning to a higher state (given $n<N$) is equal to, $\frac{\lambda(u_n)}{\nu}$. The probability of transitioning to a lower state (given $n>0$) is equal to $\frac{n\mu}{\nu}$. The probability of staying in the same state is equal to $1-\sum_{i\in Z(n)} P_{n,i}$, where Z(n) includes all of the neighboring states, except *n*. Substituting these values into the Bellman' equation, we obtain,

$$J^*+h(0)=\max_{u_0}\left[\lambda(u_0)u_0+\frac{\lambda(u_0)}{\nu}h(1)+(1-\frac{\lambda(u_0)}{\nu})h(0)\right], \tag{12}$$

$$J^*+h(N)=\max_{u_N}\left[\lambda(u_N)u_N+\frac{N\mu}{\nu}h(N-1)+(1-\frac{N\mu}{\nu})h(N)\right], \tag{13}$$

and for $0<n<N$,

$$J^*+h(n)=\max_{u_n}\Big[\lambda(u_n)u_n+\frac{\lambda(u_n)}{\nu}h(n+1)+\frac{n\mu}{\nu}h(n-1)+(1-\frac{\lambda(u_n)}{\nu}-\frac{n\mu}{\nu})h(n)\Big]. \tag{14}$$

Different dynamic programming methods can be used to solve this equation. Two possible methods are policy iteration and value iteration. Value iteration was chosen for this application because it is simpler to implement than policy iteration in this case. Value iteration is known to converge to the unique solution of Bellman's equation.

In formulating the dynamic program, the algorithm described below was used. Note that $h_t(n)$, $g_t(n)$, and $u_{t,n}$ correspond to approximations calculated by the algorithm in the $t^{th}$ iteration.

1. Set $h_0(n)=0$ (for the initial value) for all n, $0 \le n \le N$.
2. Maximize the right-hand side of equations (12), (13), and (14) with respect to $u_{t,n}$. Call the result $g_t(n)$. The price that attains the maximum can be found by taking the derivative with respect to $u_{t,n}$ and setting the result equal to 0. Solving for $u_{t,n}$,

$$u_{t,n}=\begin{cases}\dfrac{\frac{\lambda_1}{\nu}\left[h_{t-1}(n+1)-h_{t-1}(n)\right]-\lambda_0}{-2\lambda_1} & ,0 \le n < N \\ \dfrac{\lambda_0}{\lambda_1} & ,n=N,\end{cases} \tag{15}$$

If the price $u_{t,n}$ is not in the interval $\left[0, \frac{\lambda_0}{\lambda_1}\right]$, then the demand function described in equation (1) will be outside of the relevant range. To avoid this problem, we constrain $u_{t,n}$ to lie in the interval $\left[0, \frac{\lambda_0}{\lambda_1}\right]$, instead of using the price provided by equation (15). When the system is in state $N$, the price is set to $\frac{\lambda_0}{\lambda_1}$ to ensure that there are no additional arrivals into the system.

3. Set $h_t(n) = g_t(n) - g_t(0)$. Since we are interested in the relative reward, $h_t(n)$, between different states, we can arbitrarily set any one of the relative rewards equal to 0. In this case, we chose to set $h_t(0)$ equal to zero. If,

$$e > \left|h_t(n) - h_{t-1}(n)\right|, \tag{16}$$

for all $n$, where $e$ is an appropriately chosen error bound, then the last iteration has not changed any of the relative rewards above the acceptable threshold, $e$, and an acceptable approximation to the optimal policy has been found. The approximation for the optimal steady state revenue rate, is $J^*=g_t(0)$, where $g_t(0)$ is available from the last iteration. The approximation of the optimal pricing policy vector is $\mathbf{u^*}=(u_{t,0}, u_{t,1}, .., u_{t,N})$, where $u_{t,n}$ are available from the last iteration.

If the error bound requirement is not met, compute another iteration of values for $g_t(n)$ and $h_t(n)$ (return to step 2).

*Total Social Welfare*

Although we use revenue as the measure for performance in developing the optimal pricing policy, it is possible to use total social welfare as the measure of performance instead. Additionally, even if revenue is the performance measure used, the system can still be analyzed to see how the revenue maximization policy affects the total social welfare generated.

The total welfare for the system is defined as the amount of welfare generated for the service provider plus the additional welfare that the user receives from using the network. The additional welfare the user receives is defined as the amount in excess of the price, $u$, that the user would have been willing to pay to connect to the network. No welfare is generated if the user is refused access to the network.

From the demand curve, it is possible to construct the PDF for the social welfare a user would get from connecting to the network. The maximum welfare that a user could get is $x_{\max} = \frac{\lambda_o}{\lambda_1}$. The PDF for welfare is uniform from $x$=0 until $x$=$x_{max}$. At $x$=$x_{max}$, the PDF will reach 0. The PDF for the welfare that a user gets from connecting to the network can be derived directly from this information, which was determined by the demand function in equation (1). The PDF for the social welfare a user gets can be determined by,

$$f_x(x_0) = \begin{cases} \frac{\lambda_1}{\lambda_0} & ,0 < x_0 < \frac{\lambda_0}{\lambda_1} \\ 0 & , \textit{otherwise} \end{cases}. \qquad (17)$$

It is possible to derive the conditional PDF for the additional welfare that a user would get, given the user paid price $u$. The conditional PDF can be derived from equation (17).

$$f_{x|u}(x_0 \mid u) = \begin{cases} \dfrac{\frac{\lambda_1}{\lambda_0}}{\int_u^{x_{max}} f_x(x_0)dx_0} & ,u < x_0 < \dfrac{\lambda_0}{\lambda_1} \\ 0 & , \textit{otherwise}. \end{cases} \tag{18}$$

The expected value of this conditional PDF can be calculated to be,

$$E(x_0 \mid u) = u + \frac{1}{2}\left(\frac{\lambda_0}{\lambda_1} - u\right). \tag{19}$$

*Simulating the System*

A simulation for the system was built in the system dynamics modeling application Vensim, by Ventana Systems, Inc. Vensim is a system dynamics modeling application with a graphical interface. The application is used to model stocks, flows, and auxiliary variables. Stocks are variables that hold state, flows affect stock values, and auxiliary variables can be used as inputs to the system, intermediate variables, and to affect other auxiliary variables and flows. Stocks, flows, and auxiliary variables are all connected by arrows to show direct cause and effect relationships between all variables.

This application advances the time variable by discrete steps. Modeling actual time as this discretized time variable in the simulation will introduce an error because the actual time in the system is continuous. Choosing a small discretized time step ($<<\frac{1}{\nu}$) so that the probability of two actual events occurring in a single time step is negligible can reduce this error, but will increase computational requirements (number of steps).

One way to avoid this error completely is to model the discretized time in the simulation as "event time", where each discretized time step corresponds to an event occurring in the system.

One of three events can happen at each time step: an arrival could occur (with probability $\frac{\lambda_0}{\nu}$), a departure could occur (with probability $\frac{n\mu}{\nu}$), or no event could occur (with probability $1-\frac{\lambda_0+n\mu}{\nu}$). Event time can be related to real time as a summation of a series of independent exponential random variables. The PDF for this exponential distribution is,

$$f_t(x_0)=\begin{cases} \nu e^{-\nu x_0} & x_0 > 0 \\ 0 & otherwise. \end{cases} \tag{20}$$

The "event time" in the simulation can then be related to actual time by,

$$t=\frac{\sum_{i=1}^{\#\text{ of events}} -\ln(p_i)}{\nu}. \tag{21}$$

Each $p_i$ is an independent, uniform random variable between 0 and 1. The expected value for the actual time elapsed is,

$$E(t) = \frac{\sum_{i=1}^{\#\text{ of events}} 1}{\nu} = \frac{\#\text{ of events}}{\nu}. \qquad (22)$$

As the number of events becomes large, the expected value for time elapsed becomes a more accurate estimate for the actual time elapsed.

Using event time to model the arrivals and departures of the system is accurate because time is a continuous variable and the probability of having any 2 events occur at exactly the same time is negligible.

**Single state and multiple user-class model (SM model)**

A multiple user-class model allows service providers the ability to distinguish between different types of users who have different demand curves and different bandwidth requirements. For example, the service provider can distinguish between users who need to check email, users who want to check news or stock quotes real-time, and users who are requesting video conferencing. Additionally, the demand curve for each class is constant through time.

*Customer arrival rate (Demand Curve)*

In the single state, multiple user-class model, there are $M$ different classes of users, where each class, $m$, has a different arrival rate of the form,

$$\lambda_m(u) = \max\{\lambda_{0,m} - \lambda_{1,m}u, 0\}, \qquad u \geq 0. \qquad (23)$$

*User acceptance into the network*

A call from each user-class, $m$, requires a different amount of bandwidth, $r_m$. There are $n_m$ users of class $m$ already in the network and an additional user of class $m$ is accepted into the network as long as the total available bandwidth is greater than or equal to the required bandwidth,

$$r_m + \sum_{k=1}^{M} r_k n_k \leq R \ . \tag{24}$$

*User departure rate*

Users of each class have different departure rates, $\mu_m$. If there are $n_m$ users of class $m$ in the system, the overall departure rate for class $m$ users would be $n_m \mu_m$.

*Revenue*

Revenue is again generated for the service provider whenever a user enters the system. In this case, users of different classes can be charged different prices. When the system is in state $s$, a user from class m is charged a price $u_{m,s}$. For a system where bandwidth is not a limiting constraint, the service provider can set prices for each class independently and without regard to which state the system is in. The optimal pricing strategy for a system with infinite bandwidth is,

$$u_m^* = \frac{\lambda_{0,m}}{2\lambda_{1,m}}. \tag{25}$$

The optimal revenue, J*, that the service provider can expect to earn in this case is,

$$J^*_{R\to\infty} = \sum_{k\in M} \frac{\lambda_{0,k}{}^2}{4\lambda_{1,k}}. \tag{26}$$

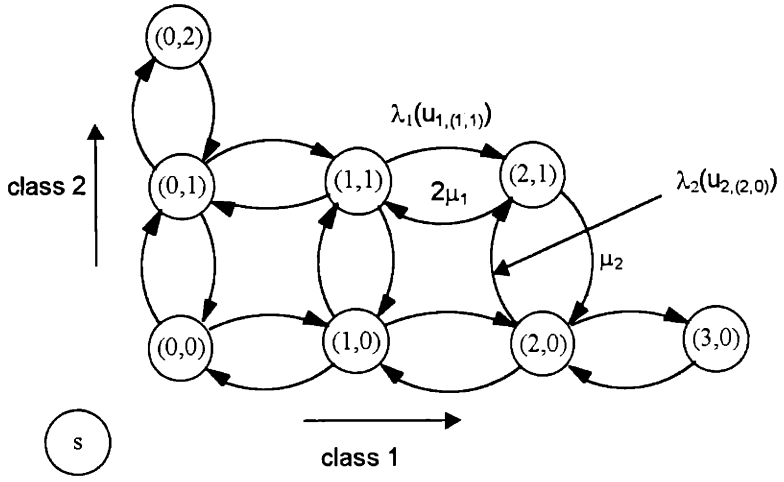
The infinite bandwidth approximation will be an accurate approximation for this system as long as,

$$\sum_{k\in M} \frac{r_k \lambda_{0,k}}{2\mu_m} << R. \tag{27}$$

*Modeling the Markov chain*

The states in the Markov chain model of the SM system must represent the number of users of each class, $n_m$. Each state, $s$, is specified by a vector with $M$ elements, where each entry indicates the number of users of each class who are currently in the system. As the number of classes, $M$, increases, the number of states required in the system increases exponentially in $M$. It is only feasible to find solutions using dynamic programming for systems where M is small. The maximum number of users of class $m$ allowed in the system (assuming there are no other users of different classes in the system) is $N_m$. Below is a Markov chain diagram for a system with $R$=7, $M$=2, $r_1$=2, and $r_2$=3.

Figure 2: Markov chain diagram for a single state, multiple user-class system with M=2.




The maximum transition rate out of any state is bounded by,

$$\nu = \sum_{k \in M} \lambda_{0,k} + \max_{k \in M} \left( \frac{R\mu_k}{r_k} \right). \tag{28}$$

*Maximizing revenue: Formulation of the dynamic program*

The basic form of the Bellman equation will still be the same. The system now consists of a larger number of states, and therefore has more equations to be solved.

$$J^* + h(s) = \max_{\mathbf{u}} \left[ \sum_{k \in M} \lambda(u_{k,s}) u_{k,s} + \sum_{j \in neighbors(s)} P_{n,j} h(j) \right]. \tag{29}$$

Like the single user-class system, *neighbors(s)* consists of all the possible states that the system could transition to from state *s*, including state *s*. However, now there are (at most) $2M+1$ states that the system could transition into.

Value iteration was used again to solve this dynamic program. The dynamic program is able to solve a system with M user-classes as long as M is small. As M increases, the number of states in the system increases exponentially, the time to solve the system increases, and beyond a threshold, the dynamic program runs out of memory.

*Total Welfare*

The PDF for the price that a user of class $m$ is willing to pay is independent of other user-classes.

$$f_x(x_0) = \begin{cases} \dfrac{\lambda_{1,m}}{\lambda_{0,m}}, & ,0 < x_0 < \dfrac{\lambda_{0,m}}{\lambda_{1,m}} \\ 0, & , \ otherwise \end{cases}. \tag{30}$$

Therefore, the expected value of the total welfare given a user of class $m$ was accepted into the system and paid $u_m$ is also independent of other user-classes. The expected value is,

$$E_{x|u_m}(x_0 \mid u_m) = u_m + \frac{1}{2}\left(\frac{\lambda_{0,m}}{\lambda_{1,m}} - u_m\right). \tag{31}$$

**Multiple state and multiple user-class model (MM model)**

The single state models described above assume that the demand function for user-classes are known and static. This is unlikely to be the case in real systems. In order to deal with the case where the demand function can change probabilistically over time and the service provider may not know the exact demand function, the multiple state model is used.

The multiple state model assumes the demand is a probabilistic function, where the demand function is a Markov chain with five states. A change in state for the demand function will result in the service provider changing the dynamic congestion-dependent price in order to maximize performance.

*Customer arrival rate (Demand Curve)*

In the multiple state, multiple user-class model, the demand is assumed to be a function of the current state, $q$, that the system is in. The system is assumed to transition, or "drift", between five different states according to the continuous-time Markov chain shown in Figure 3. If the system transitions to a higher (or lower) state, the arrival rate is modeled to increase (or decrease) by a value of $j$.

From these assumptions, the arrival rate for user-class m can be defined as,

$$\lambda_m(u) = \max\{\lambda_{0,m,q} - \lambda_{1,m}u, 0\}, \qquad u \geq 0. \tag{32}$$

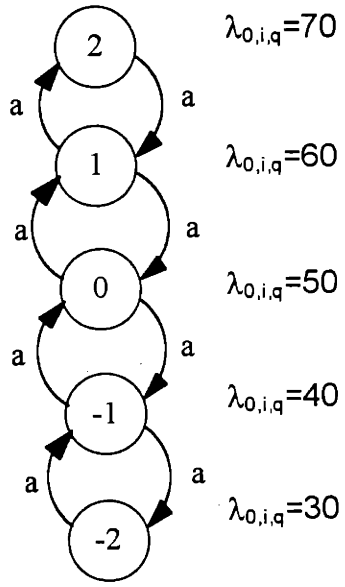
The variable $\lambda_{0,m,q}$ is defined as,

$$\lambda_{0,m,q} = \lambda_{middle,m} + qj, \tag{33}$$

where $\lambda_{middle,m}$ and $j$ are positive constants and $q$ is the current state of the demand function.

As an example, Figure 3 also shows the corresponding $\lambda_{0,i,q}$ for a specific user-class for each state of the demand function. The parameters used for this example were $\lambda_{middle,i}$ =50 and $j$=10.

Figure 3: Markov chain diagram for state q in a multiple state system with example values for $\lambda_{0,i,q}$ for a specific user-class, *i*. System parameters used were $\lambda_{middle,i}$ =50 and j=10.




The Markov chain changes states, *q*, with transition rate, *a*. A system that is modeled in such a fashion is mostly interesting if the arrival rate, λ, is much greater than *a* ($\lambda >> a$). Otherwise, the system transitions too quickly between different states for the current state of the system to have a lasting effect on the system or for an accurate estimate of the current state to be possible. This is because the state of the system is expected to transition before many arrivals occur.

*Maximizing revenue: Formulation of the dynamic program*

The basic form of the Bellman equation will still be the same. However, *q* must be part of the overall system state. There are five times as many states than in the single state system, therefore there will be five times as many equations. Also, there are possibly two

more states to transition into than before. These states correspond to a change in the value of $q$ by +1 or -1. Value iteration was used again in order to solve the set of equations.

The maximum transition rate, $\nu$, will also need to be changed to account for these two new states that the system can transition into. Since $q$ could transition to at most 2 other states and the rate of transitioning to one other state $q$ is $a$, the new value for $\nu$ is,

$$\nu = \sum_{k \in M} \lambda_{0,k} + \max_{k \in M} \left( \frac{R\mu_k}{r_k} \right) + 2a. \tag{34}$$

*Chapter 3*

# COMPARISONS AND QUALITATIVE OBSERVATIONS

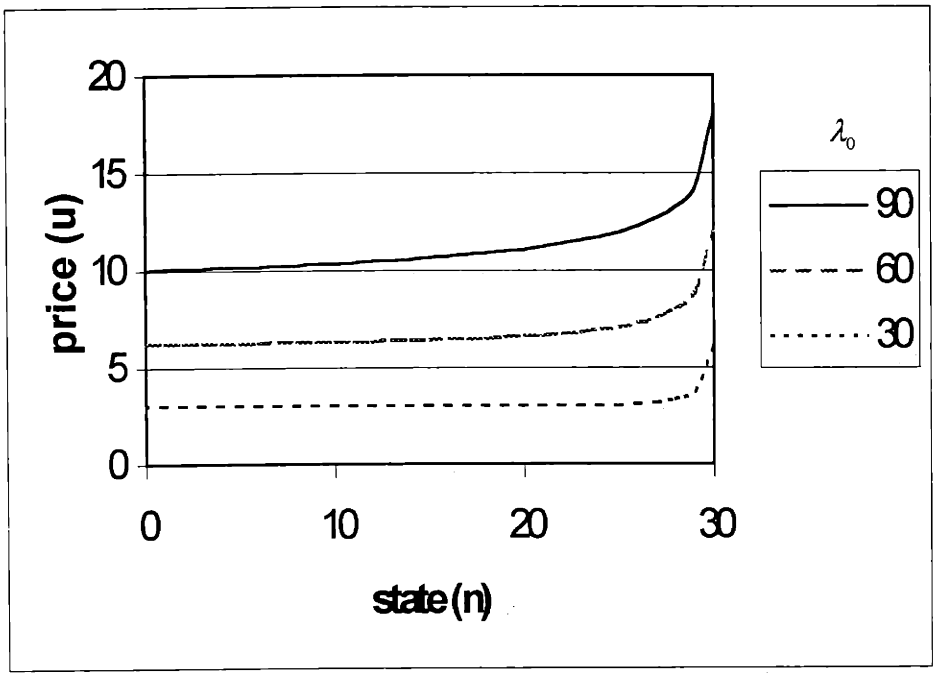
The objective of this chapter is to develop some intuition on the nature of optimal solutions and to find some methods of approximating them. The chapter begins by presenting and analyzing the results from the optimal pricing policies found by dynamic programming for each model. We then simulate the different models under the optimal pricing policy and compare the resulting simulating values for the steady state revenue rate, J, to the values for $J^*$ calculated by the dynamic program. Some of the approximations that can be used when analyzing the different models will be introduced and their accuracy will be measured. Lastly, interesting relationships between the different system parameters will be discussed and analyzed.

## Optimal pricing policies

*SS model*

The optimal pricing policies, $\mathbf{u}^*$, were determined for the SS model by dynamic programming as described on page 21 of this thesis. Examples of the optimal pricing policies for SS systems with different demand functions are shown in Figure 4.

Figure 4: Graph of optimal SS pricing policies with different $\lambda_0$'s. The system parameters were N=30, $\lambda_1$=5, and $\mu$=1.

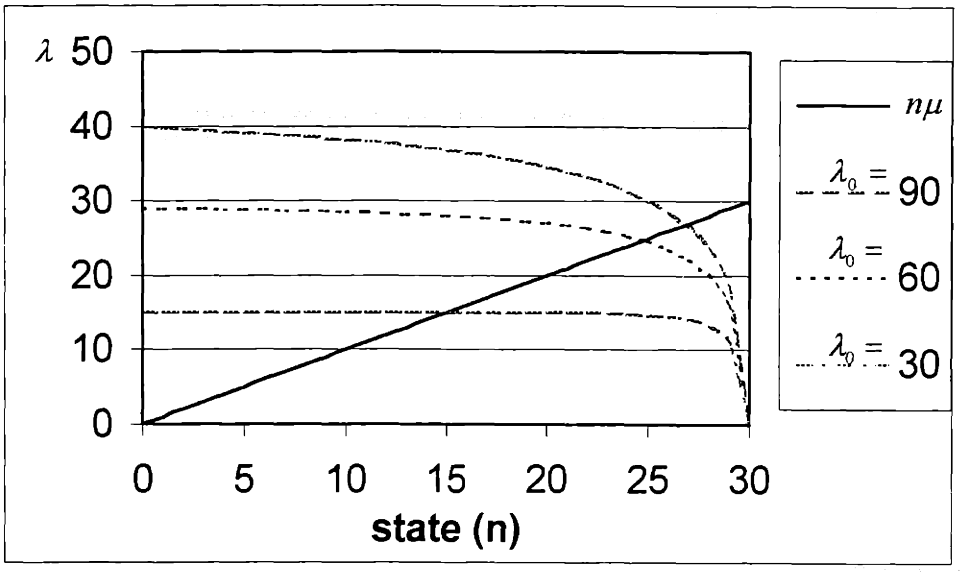



When the system is empty, prices approach the infinite bandwidth approximation prices of $u = \frac{\lambda_0}{2\lambda_1}$. As the system becomes full, prices start to increase drastically.

The distribution of the state will concentrate around the point in the system where the arrival rate is equal to the departure rate. This occurs when $\lambda(u)=n\mu$, where $\lambda(u)=\lambda_0-\lambda_1 u$, from equation (1). Figure 5 shows the intersection between these graphs. For $\lambda_0$ low, the

system will concentrate around the state $n \approx \frac{\lambda_0}{2}$. For $\lambda_0$ high, the system will concentrate close to the maximum state, $n$=N.

Figure 5: Intersection of nμ and $\lambda(u) = \lambda_0 - \lambda_1 u$ for SS systems with different $\lambda_0$ values, using the optimal pricing policies. The system parameters were N=30, $\lambda_1$=5, and μ=1.



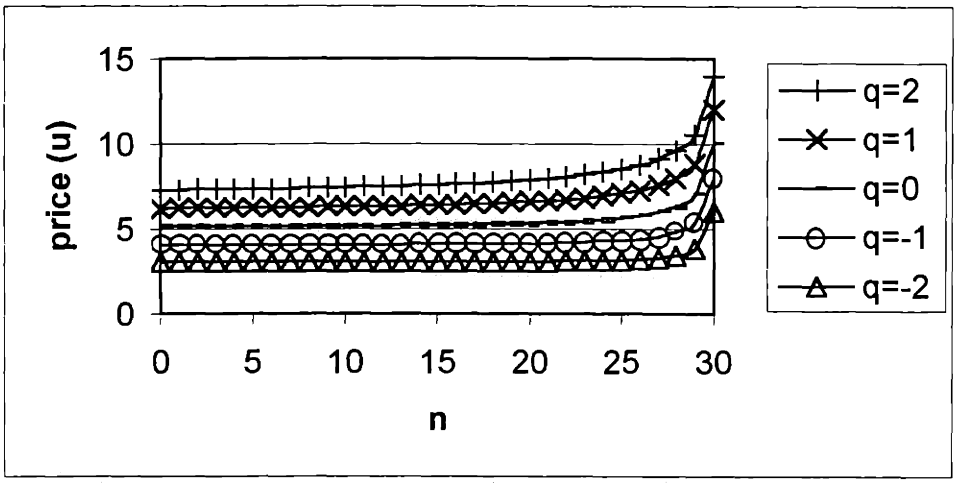

*SM model*

It is more difficult to graphically illustrate a pricing scheme for the SM model. There are on the order of $n^M$ states, instead of $n$, arranged on an $M$-dimensional grid. One interesting observation about the SM model is that even when the system is congested but not full, it is often optimal to not allow certain user-classes into the system. This is done in order to save bandwidth in case an arrival of a user from a more profitable class were to occur. More profitable user-classes can be defined as those that have a higher revenue per bandwidth * time.

*MM model*

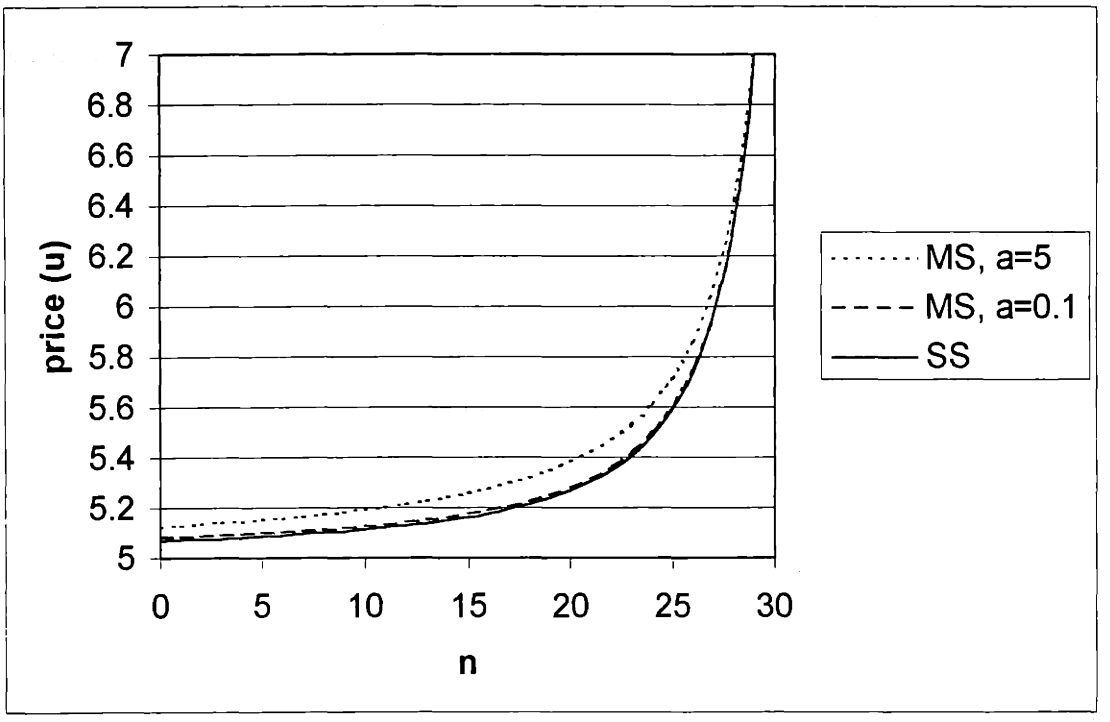
Figure 6 illustrates a simple MM model with M=1. This is a multiple state, single user-class (MS) model. The system parameters were $N$=30, $\lambda_1$=5, $j$=10, $\lambda_{middle}$=50, $a$=1, and $\mu$=1.

Figure 6: Graph of an optimal MS pricing policy.




Although these pricing policies are very similar to the pricing polices of the analogous SS systems, they are not exactly the same. The difference is accentuated when the state change rate, $a$, is large as shown in Figure 7. The system parameters for the MS systems were $N$=30, $\lambda_1$=5, $j$=10, $\lambda_{middle}$=50, and $\mu$=1.

Figure 7: Comparison of optimal MS pricing policies for q=0 with a=0.1 and a=5 to the analogous optimal SS pricing policy with $\lambda_0=\lambda_{middle}=50$.




**Simulation results versus dynamic programming results**

The values for $J^*$ from the simulation were close to the results of the dynamic program when the optimal pricing policy **u** was used. The simulations were carried out to 100,000 or 200,000 time steps, in order to ensure that the results came close to converging to the steady-state values. Table 1 and Table 2 compare these results for the SS and MS models, respectively.

Table 1: SS system dynamic programming J* values versus simulation results under the optimal policy. System parameters were N=30, $\lambda_1$=5, and $\mu$=1.

| $\lambda_0$ | J* D.P. | J Simulation |
|---|---|---|
| 30 | 45.00 | 44.83 |
| 60 | 167.78 | 167.72 |
| 90 | 317.99 | 317.25 |

Table 2: MS system dynamic programming J* values versus simulation results under the optimal policy. System parameters were a=5 and j=5.

| N | $\lambda_{middle}$ | $\lambda_1$ | $\mu$ | J* | J |
|---|---|---|---|---|---|
| | | | | D.P. | Simulation |
| 5 | 60 | 5 | 1 | 44.00 | 44.24 |
| 5 | 40 | 1 | 10 | 396.83 | 402.28 |
| 10 | 60 | 5 | 0.5 | 47.21 | 47.33 |
| 5 | 40 | 1 | 10 | 396.83 | 392.27 |

## Additional SS system observations

*Infinite Bandwidth Approximation*

For the SS system, the infinite bandwidth approximation says that the system can be modeled as having an infinite amount of bandwidth if the steady state arrival rate is less than the maximum departure rate. As shown in equations (7) and (9), this means that $J^*_{R\to\infty} = \frac{\lambda_0^{\,2}}{4\lambda_1}$ is an accurate estimate of J* and is valid when $\frac{\lambda_0}{2} << N\mu$, where $N$, the maximum number users in the system, is proportional to R, the maximum bandwidth available.

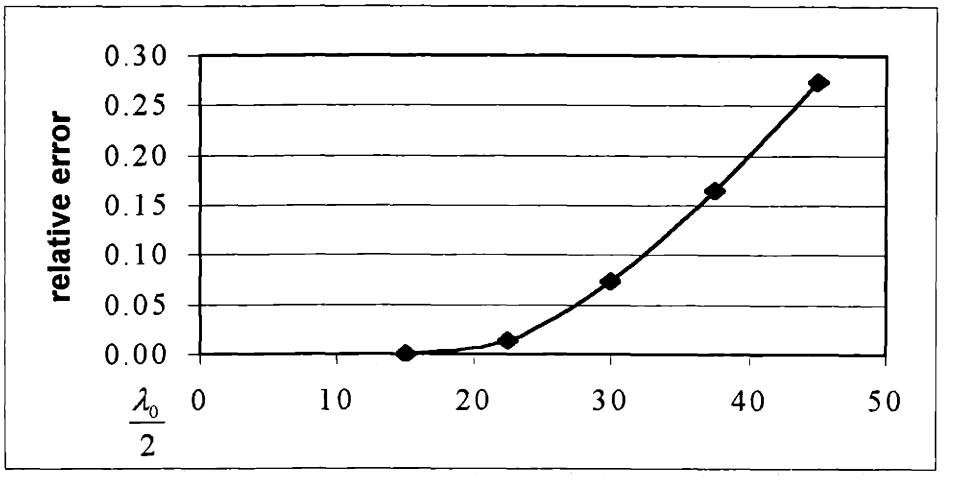
Table 3 illustrates the accuracy of the infinite bandwidth approximation for an SS system with $\lambda_1=5$. J* values were computed using dynamic programming.

Table 3: SS system dynamic programming J* values versus infinite bandwidth calculations.

| $\lambda_0$ | N | $\mu$ | $\lambda_0/2$ | $N\mu$ | J* | $J^*_{R\to\infty}$ | $\frac{J^*_{R\to\infty}-J^*}{J^*}$ |
|---|---|---|---|---|---|---|---|
| 30 | 30 | 1 | 15 | 30 | 45 | 45 | 0.00 |
| 45 | 30 | 1 | 22.5 | 30 | 99.9047 | 101.25 | 0.01 |
| 60 | 30 | 1 | 30 | 30 | 167.7775 | 180 | 0.07 |
| 75 | 30 | 1 | 37.5 | 30 | 241.4109 | 281.25 | 0.17 |
| 90 | 30 | 1 | 45 | 30 | 317.9921 | 405 | 0.27 |

As $\frac{\lambda_0}{2}$ increases beyond $N\mu$, the infinite bandwidth approximation becomes less accurate. When $\frac{\lambda_0}{2}$ becomes much larger than $N\mu$, there is a linear relationship between the error and $\frac{\lambda_0}{2}$. Figure 8 shows this relationship.

Figure 8: Graph of infinite bandwidth approximation relative error versus $\lambda_0/2$ for SS model.




*Upper bound approximation for J* when* $\frac{\lambda_0}{2} > N\mu$

If $\frac{\lambda_0}{2} > N\mu$, it is possible to get an approximation for J* that is more accurate than the infinite bandwidth approximation. For such a system, the maximum arrival rate that results in a stable system is $\lambda=N\mu$. The upper bound on performance for such a system is when all bandwidth is always being used. It is possible [7] to estimate a value for J* using that approximation,

$$J_{ub} = u\lambda = uN\mu, \tag{35}$$

$$\lambda_0 - \lambda_1 u = N\mu, \tag{36}$$

and solving for *u*,

$$u = \frac{\lambda_0 - N\mu}{\lambda_1}, \tag{37}$$

so,

$$J_{ub} = uN\mu = N\mu\left(\frac{\lambda_0 - N\mu}{\lambda_1}\right). \tag{38}$$

The accuracy of this upper bound approximation is illustrated in Table 4. The upper bound approximation is shown to be more accurate than the infinite bandwidth approximation when $\frac{\lambda_0}{2}$ is larger than Nμ. The system parameters (except for $\lambda_0$ which is indicated below) are identical to those in Table 3.

Table 4: SS system J* values versus $J_{ub}$ and infinite bandwidth approximations.

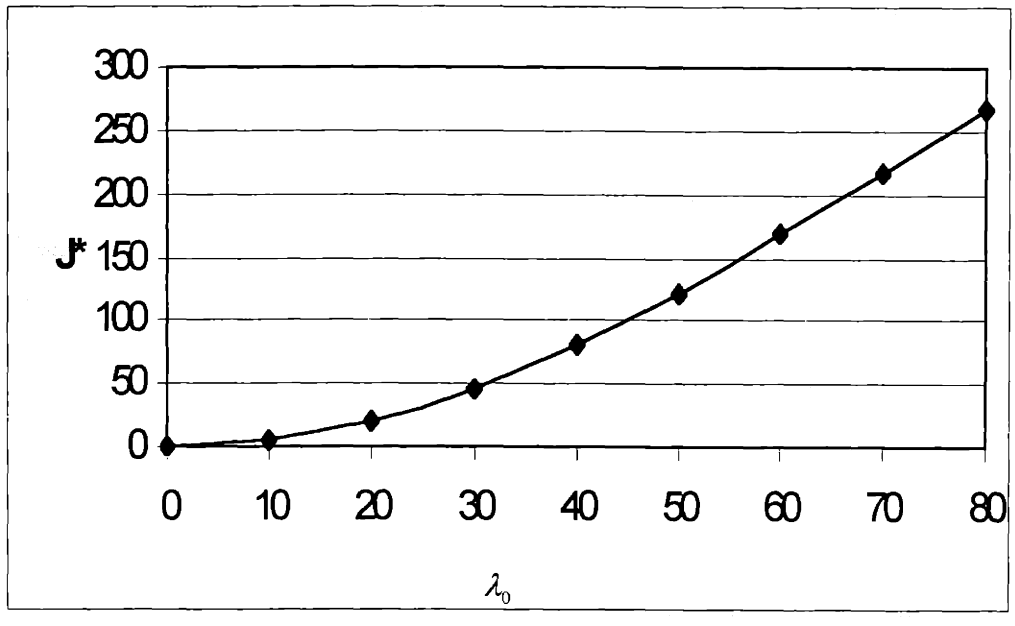
| $\lambda_0$ | J* | $J^*_{R\to\infty}$ | $J_{ub}$ | $\frac{J^*_{R\to\infty}-J^*}{J^*}$ | $\frac{J_{ub}-J^*}{J^*}$ |
|---|---|---|---|---|---|
| 60 | 167.78 | 180 | 180 | 0.07 | 0.07 |
| 75 | 241.41 | 281.25 | 270 | 0.17 | 0.12 |
| 90 | 317.99 | 405 | 360 | 0.27 | 0.13 |
| 200 | 912.199 | 2000 | 1020 | 1.19 | 0.12 |

*Relationship between J* and $\lambda_0$*

Using these two approximations, it is possible to determine the approximate relationship between J* and $\lambda_0$ when all other parameters are held constant. When $\lambda_0$ is small, the infinite bandwidth approximation indicates that J* will increase quadratically with $\lambda_0$ since $J^*_{R\to\infty} = \frac{\lambda_0^2}{4\lambda_1}$. As $\lambda_0$ increases past Nμ, the upper-bound approximation $J_{ub}$

becomes a better estimate for the system's $J^*$. $J_{ub}$ is proportional to $\lambda_0$ so $J^*$ will then increase linearly with $\lambda_0$. Figure 9 shows this relationship for a system with parameters $N$=30, $\lambda_1$=5, and $\mu$=1.

Figure 9: Graph of $J^*$ versus $\lambda_0$ for SS model.




*Social Welfare and Congestion-Dependent Pricing*

So far, congestion-dependent pricing has been evaluated from a revenue maximization perspective. However, one of the other important motivations of congestion-dependent pricing is to increase social welfare. It is possible to develop a congestion-dependent pricing system that maximizes total social welfare by solving a dynamic program similar to the one outlined in this thesis. However, even a revenue-maximization congestion-dependent pricing policy can increase total social welfare over a system whose pricing

policies results in heavy congestion. For a static pricing policy, a heavily congested system could be considered a system that is full to capacity a significant amount of the time. From this assumption, an SS system with static pricing policy, $u_s$, can be classified as heavily congested when,

$$\lambda(u_s) >> N\mu. \tag{39}$$

Table 5 compares a system where the static pricing policy leads to heavy congestion compared to the welfare generated for the same system operated under the revenue-maximization congestion-dependent policy. This does not mean that a policy that maximizes revenue will also maximize total social welfare. In fact, a monopolist who maximizes revenue will usually do so by setting prices below the total social welfare maximization price. However, the example does point out that for certain congested systems, a policy that maximizes revenue can also improve total social welfare compared to a policy that causes heavy congestion.

Table 5: Comparison of total social welfare for a system with different pricing polices. w is the total social welfare per unit time. System parameters were N=30, $\lambda_0$=80, $\lambda_1$=5, and $\mu$=1.

| Pricing Policy | System Description | Average # of users | J | w |
|---|---|---|---|---|
| $u_s$=5 | heavy congestion | 28.9 | 144.86 | 304.26 |
| optimal u* | revenue maximization | 25.4 | 266.6 | 336.05 |

**Additional MS system observations**

*Approximations for MS systems with small a*

For an MS system, if *a* is small compared to the other system parameters, the probability of the demand function changing states, *q*, is low. The MS system's steady state

J* can then be modeled as the average of five SS systems. Each value of $\lambda_{0,m}(\lambda_{middle,m}, q, j)$ for the MS system can be used in place of $\lambda_0$ for each SS system. This averaged approximation will provide a lower bound to the J* that the actual MS system will generate. Table 6 shows the averaged SS system approximation calculations for an MS system with parameters $N$=30, $\lambda_1$=5, $j$=10, $\lambda_{middle}$=50, $a$=1, and $\mu$=1. Table 7 compares J* of different MS systems and their averaged SS system approximations.

Table 6: Averaged SS system approximation calculations for an MS system with parameters N=30, $\lambda_1$=5, j=10, $\lambda_{middle}$=50, a=1, and $\mu$=1.

| SS systems: N=30, $\lambda_1$=5, $\mu$=1 | |
|---|---|
| $\lambda_0$ | J* |
| 70 | 216.45 |
| 60 | 161.78 |
| 50 | 121.57 |
| 40 | 79.65 |
| 30 | 45 |
| Averaged approximation: | 124.89 |
| | |
| Actual J* for MS system: | 126.716 |

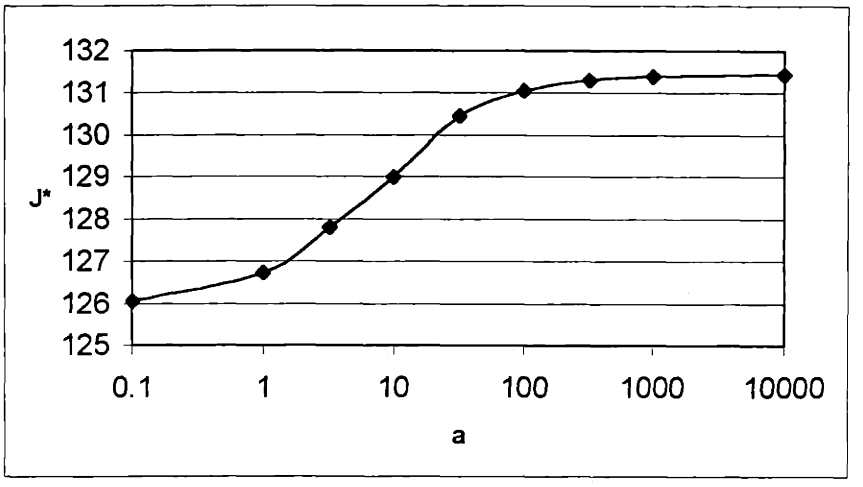
Table 7: Comparison of various MS system J* to averaged SS system approximations. MS systems have parameters N=30, $\lambda_1$=5, j=10, a=1, and $\mu$=1.

| $\lambda_{middle}$ | MS system | Averaged SS |
|---|---|---|
| | J* | Approximation |
| 20 | 29.91 | 29.93 |
| 30 | 54.42 | 54.24 |
| 40 | 87.24 | 85.60 |
| 50 | 126.72 | 124.89 |
| 60 | 171.06 | 169.23 |
| 70 | 218.62 | 216.90 |
| 80 | 268.20 | 266.60 |

*Relationship between a and J**

For an MS system, if *a* increases, (with all other system parameters kept the same) the value of J* increases monotonically. This may be a result from the optimal pricing strategy being able to take advantage of frequent state changes to further increase revenue. Figure 10 shows the relationship between *a* (on a log scale) and J* for an MS system with parameters $N$=30, $\lambda_1$=5, $j$=10, $\lambda_{middle}$=50, and $\mu$=1.

Figure 10: Graph of J* versus a for an MS system.




*Relationship between a, $\lambda_{middle}$, and J**

For a system with $\lambda_{middle}$ small, the difference in J* for analogous systems with different state change rates, *a*, is small. As $\lambda_{middle}$ increases, this effect seems to increase. This difference reaches a maximum and past a certain threshold for $\lambda_{middle}$ the difference seems to decrease as a function of $\lambda_{middle}$. This threshold for $\lambda_{middle}$ that maximizes the difference in revenue for analogous systems appears to be related to the point in a system where the system is between under-utilization and over-utilization. That occurs when

$$\lambda_{middle} \approx 2N\mu . \tag{40}$$

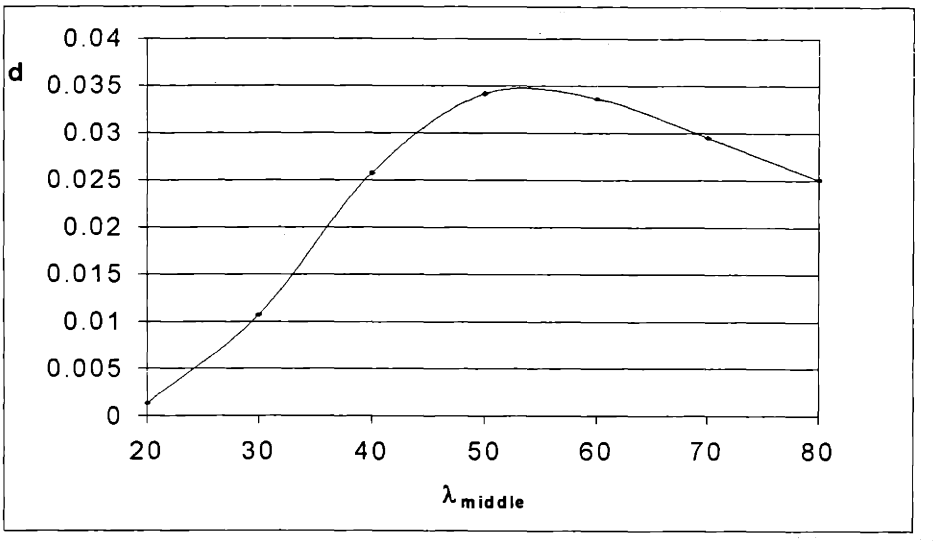
Figure 11 shows the difference in revenue for analogous systems versus $\lambda_{middle}$. The difference in revenue is graphed as the difference, *d*, between J* for systems with state

change rates of $a$=100 and $a$=1 normalized by J* for the system with $a$=1. The value $d$ is defined as,

$$d = \frac{J^*_{a=100} - J^*_{a=1}}{J^*_{a=1}}. \tag{41}$$

The system parameters were $N$=30, $\lambda_1$=5, $j$=10, $\lambda_{middle}$=50, and $\mu$=1. It is interesting to note that the maximum difference appears to occur near the point where $\lambda_{middle} \approx 2N\mu = 60$.

Figure 11: Graph of $d$ versus $\lambda_{middle}$ for an MS model.




*Relationships between j, $\lambda_{middle}$, and J**

It is possible to determine the effects of $j$ and $\lambda_{middle}$ on J* for systems with different utilization rates. As shown in Table 7, it is possible to estimate J* for an MS system using an

averaged SS system approximation. Changing $j$ in the MS system affects the $\lambda_0$ parameters of each of the SS systems that are averaged. For illustrative purposes, a system with three states for $q$ (instead of five) is used to derive the relationship between J* and $j$.

*Derivation of the relationship between J* and j*

A three-state MS system can be modeled as an average of three SS systems with $\lambda_0 = \lambda_{middle} - j$, $\lambda_{middle}$, and $\lambda_{middle} + j$. If J* for each SS system is considered a function of $\lambda_0$, an approximation for J* for the MS system is equivalent to,

$$J_{\text{MS approximation}} = \frac{1}{3}\left(J^*(\lambda_{middle} - j) + J^*(\lambda_{middle}) + J^*(\lambda_{middle} + j)\right). \quad (42)$$

*Relationships between J* and j for underutilized systems*

If the system is underutilized, the infinite bandwidth approximation for the SS system will be an accurate estimate for J*. This approximation (equation (7)) shows that $J^* \propto \lambda_0^{\,2}$. The relationship between an MS system approximation for J* and $j$ can be established,

$$J_{\text{MS approximation}} \propto \frac{1}{3}\left((\lambda_{middle} - j)^2 + (\lambda_{middle})^2 + (\lambda_{middle} + j)^2\right), \quad (43)$$

$$J_{\text{MS approximation}} \propto \frac{1}{3}\left(3\lambda_{middle}^{\,2} + 2j^2\right). \quad (44)$$

It can be concluded that J* for the MS system will also be proportional to $j^2$. J* should increase quadratically with $j$.

*Relationships between J* and j for over-utilized systems*

If the system is over-utilized, the $J_{ub}$ approximation is more accurate in estimating J* for each SS system. The $J_{ub}$ approximation (equation (38)) shows that J* is proportional to $\lambda_0$. So the relationship between the approximation and $j$ will be,

$$J_{\text{MS approximation}} \propto \frac{1}{3}\left((\lambda_{middle} - j) + (\lambda_{middle}) + (\lambda_{middle} + j)\right), \tag{45}$$

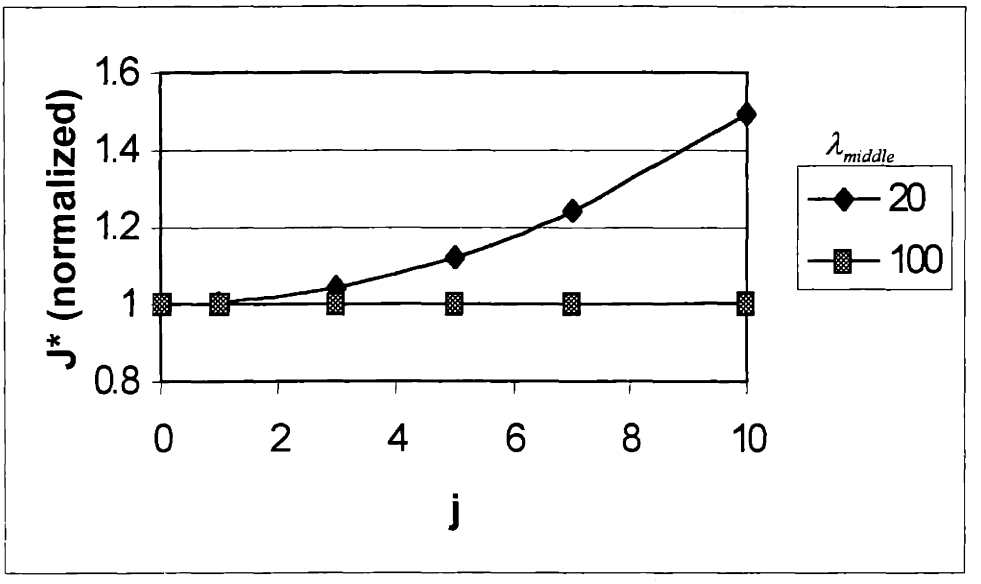
$$J_{\text{MS approximation}} \propto \lambda_{middle}. \tag{46}$$

It can be concluded that increasing $j$ will result in little change in J* for the MS system.

*A graphic representation of the relationships*

Figure 12 shows this relationship between J* and $j$ for an over-utilized system (with $\lambda_{middle}$ =100) and an underutilized system (with $\lambda_{middle}$ =20). The values for J* were normalized by J* for the corresponding MS system with $j$=0 in order to graph both systems on the same scale. Second order effects, such as limitations of the approximations and the changing utilization levels of systems associated with changes in $j$, exist and have larger effects when j is large. The system parameters were $N$=30, $\lambda_1$=5, $j$=10, $\alpha$=1, and $\mu$=1.

Figure 12: Graph of normalized J* versus j for MS systems with low utilization ($\lambda_{middle}$=20) and high utilization ($\lambda_{middle}$=100).



*Chapter 4*

# ESTIMATION OF SYSTEM PARAMETERS

In a real system, the service provider may not know all of the system parameters. For instance, in an SS system, the details of the demand curve beyond the general shape may not be known. In an MS system, even if the probabilistic demand function is known, the current state that the demand function is in may not be know. There are different methods for determining these system parameters with different advantages and disadvantages.

This chapter will discuss methods for estimating $\lambda_0$ and $\lambda_1$ for an SS system where the price, $u$, is held constant over all states. Estimating these parameters is not dealt with in detail because it is a relatively simple problem.

Estimating some of the MS system parameters poses a more challenging question. Different methods of estimating the current state of the demand function, $q$, for an MS system, where all other system parameters are known, will be discussed. We will also discuss how to set prices based on an estimate for $q$.

The estimation topics discussed in detail for the MS system are:

1. Analyzing different shaped windows for estimation.
2. Determining the length of the estimation window to minimize error.

3. Determining which windows minimize system memory requirements.

4. Comparing the effects of rounding the estimation data to determine a specific state versus interpolating prices between different states.

## SS System Estimations

*Estimating* $\lambda_0$

*Estimation when prices are constant*

It is possible to estimate $\lambda_0$ in an SS system where the price, $u$, is held constant over all states. In this case, it is possible to estimate $\lambda_0$ by,

$$\hat{\lambda}_0 = \hat{\lambda} + \lambda_1 u . \tag{47}$$

In this case, the service provider knows both $\lambda_1$ and $u$, so the service provider only needs to estimate the Poisson arrival rate, $\lambda$, of the system.

FIXED LENGTH WINDOW ESTIMATION

One method to estimate $\lambda$ is by counting the number of arrivals, $k$, that occur over a window of fixed length W. An estimate for $\lambda$ can then be determined by,

$$\hat{\lambda} = \frac{k}{W} . \tag{48}$$

The longer the length of the window, W, that is chosen, the more accurate the estimate for $\lambda$ becomes. At the very least, the length of W should be chosen so that there are many expected arrivals in the window, or E(k)>>1. This expression can be rewritten as,

$$W >> \frac{1}{\lambda}. \tag{49}$$

The problem of using such a window of finite fixed length, W, is that the maximum amount of memory that is required for such a system is equal to $k$, the number of arrivals that occur in the window. The interarrival time between each arrival must be stored in order to determine how many arrivals must be summed inside the window, W to determine $k$. As new arrivals occur, old arrivals that are no longer in the window are discarded. The number of Poisson arrivals over a window of fixed length, W, is a random variable described by the Poisson distribution. The PMF for this Poisson distribution is,

$$p_k(k_0) = \frac{(\lambda W)^{k_0} e^{-\lambda W}}{k_0!} \qquad k_0 = 0,1,2,\ldots \tag{50}$$

The PMF is positive for all positive integer values of $k_0$. This means that probabilistically, there can be a very large number of arrivals, $k$ over a window of fixed length, W. Because the amount of memory the system will require to estimate $\lambda$ is equal to $k$, it can vary and could be very large.

### FIXED NUMBER OF ARRIVALS, $K_{FIXED}$, ESTIMATION

One way to avoid this problem of memory requirements is to fix the number of arrivals, $k_{fixed}$, and estimate $\lambda$ by dividing $k_{fixed}$ by the amount of time required to get $k_{fixed}$ arrivals. The estimate for $\lambda$ will be,

$$\hat{\lambda} = \frac{k_{fixed}}{\sum_{i \in k} \text{interarrival time}_i}. \tag{51}$$

This system requires the system to store exactly $k_{fixed}$ values, regardless of the length of each interarrival time. The larger the number of arrivals, $k_{fixed}$, the system estimates over, the more accurate estimate of $\lambda$ will be.

### OVERALL AVERAGED ESTIMATION

The optimal window, W, can be found by taking the limit of $W \to \infty$. This results in computing the average number of arrivals per unit time of the system over all time,

$$\hat{\lambda} = \frac{\text{total \# of arrivals}}{\text{total time}}. \tag{52}$$

The benefits to using this window are that the estimation will be the most accurate and the memory required to compute such an estimate is low. The estimation only requires keeping track of two values.

*Estimation when prices are not constant*

For an SS system, it is possible to estimate $\lambda_0$ even if the price, $u$, changes over time. This can be easily implemented if the demand curve is linearly related with price. It is necessary to calculate $\overline{u}$, the average price the service provider charges over the window, W. An estimate $\hat{\lambda}$ for the Poisson arrival rate, $\lambda$, can again be calculated using equation (48), (51), or (52). An estimate for $\lambda_0$ can then be calculated by,

$$\hat{\lambda}_0 = \hat{\lambda} + \lambda_1 \overline{u} . \tag{53}$$

*Estimation when the demand curve is unknown*

If the demand curve is not known, it is still possible to determine an estimate for the Poisson arrival rate, $\lambda$, if the price, $u$, is held constant. This can be done using one of the methods described above in equations (48), (51), and (52).

If the price, $u$, is not held constant and the demand curve is not know, it is still possible to determine an estimate for $\lambda(\overline{u})$ in certain cases. An accurate estimate can be made if the changes in price are small enough so that the demand curve can be estimated as being linear over the relevant price range. A value for $\lambda(\overline{u})$ can be estimated by again using equation (48), (51), or (52).

*Estimating $\lambda_1$*

Determining an accurate estimate of $\lambda_1$ is possible if accurate estimates for the Poisson arrival rate, $\lambda$, are determined at two different prices. The estimate for $\lambda_1$ will become more accurate as the difference between the two prices increases. There are difficulties to setting prices that are vastly different from each other in a real system that may make this estimation implausible.

**MS System Estimations**

*Estimating the state, q*

The state, $q$, that the system is in can be estimated in an MS system where all other parameters are known. A service provider may be interested in estimating $q$ to determine when the system is in a state of shock and congestion may become a problem.

To determine the state, $q$, in a system, the service provider needs to:

1. Estimate the arrival parameter, $\lambda$, over a specific window.
2. Use the data to estimate what the current state of the system is and then set prices accordingly.

The next two sections will discuss methods to choose the appropriate window and to estimate the current state of the system in order to set prices.

*Determining the appropriate window*

Different window lengths and shapes can be chosen to estimate the arrival parameter, λ. Unlike the SS system, the optimal length of the window should not be chosen to be as long as possible. When the system changes state, the data of the system from the previous state will give an inaccurate estimate of the current state. For this reason, the window size should not be chosen much larger than $\frac{1}{2a}$. However, the window should be large enough to include several arrivals.

The two window shapes that are considered in this thesis are a rectangular window and an exponential window. The windows are discussed in the following two sections.

## THE RECTANGULAR WINDOW

Like the SS system, a rectangular window can be used to estimate λ. The rectangular window has an impulse response of,

$$h(t) = \begin{cases} \frac{1}{W} & 0 < t < W \\ 0 & \text{otherwise} \end{cases} \tag{54}$$

If x(t) is defined as the signal where each arrival into the system at time $t_{arrival}$ corresponds to an impulse, then,

$$x(t) = \sum_{t_{arrival} \in arrivals} \delta(t - t_{arrival}) . \tag{55}$$

An estimate for the arrival rate at time, t, can then be determined by,

$$y(t) = x(t) * h(t). \tag{56}$$

Since h(t) is a causal system, the estimate of y(t) can be computed in real time and will only be dependent on the number of arrivals that have occurred in the system up to the current time. This system is equivalent to the one in equation (48),

$$y(t^*) = \hat{\lambda}(t^*) = \frac{k_{t^*}}{W}, \tag{57}$$

where $k_t$ is the number of arrivals that occurred between time t*-W and t*.

Optimizing the rectangular window

The rectangular window length, W, can be set to minimize the estimation error. There are two types of error that are caused by the estimation,

(i.) The error caused by having a finite number of data points for the random variable to be estimated. This assumes that the random variable that is being estimated has fixed statistics; a change in state has not occurred.

(ii.) The error caused by the system changing state, *q*.

The total error can be expressed as the sum of these two errors,

$$E\left[\left(\hat{\lambda}_{t^*} - \lambda_{t^*}\right)^2\right] = \underbrace{E\left[\left(\hat{\lambda}_{t^*} - \frac{1}{W}\int_{t^*-W}^{t^*} \lambda_\tau d\tau\right)^2\right]}_{i} + \underbrace{E\left[\left(\lambda_{t^*} - \frac{1}{W}\int_{t^*-W}^{t^*} \lambda_\tau d\tau\right)^2\right]}_{ii}, \tag{58}$$

and then the two errors can be analyzed separately.

This expression is justified because the cross term (the covariance) can be shown to be zero since we model the arrivals into the system and a change in state as independent Poisson processes.

i. Random variable estimation error

This error is caused by having a finite set of data points from which a random variable must be estimated. It is assumed that the random variable that is being estimated is fixed and that this parameter does not change over the window of estimation.

The error can be rewritten as,

$$\begin{aligned} \text{error}_i &= E\left[\left(\hat{\lambda}_{t^*} - \frac{1}{W}\int_{t^*-W}^{t^*} \lambda_\tau d\tau\right)^2\right] \\ &= E\left[E\left[\left(\hat{\lambda}_{t^*} - \frac{1}{W}\int_{t^*-W}^{t^*} \lambda_\tau d\tau\right)^2 \middle| \{\lambda_t\}\right]\right]. \end{aligned} \tag{59}$$

This expression can be rewritten as,

$$\begin{aligned} \text{error}_i &= E\left[ E\left[ \left( \hat{\lambda}_{t^*} - \frac{1}{W}\int_{t^*-W}^{t^*} \lambda_\tau d\tau \right)^2 \middle| \{\lambda_t\} \right] \right] \\ &= E\left[ \frac{1}{W^2} \int_{t^*-W}^{t^*} \lambda_\tau d\tau \right] \\ &= \frac{E[\lambda_t]}{W}. \end{aligned} \tag{60}$$

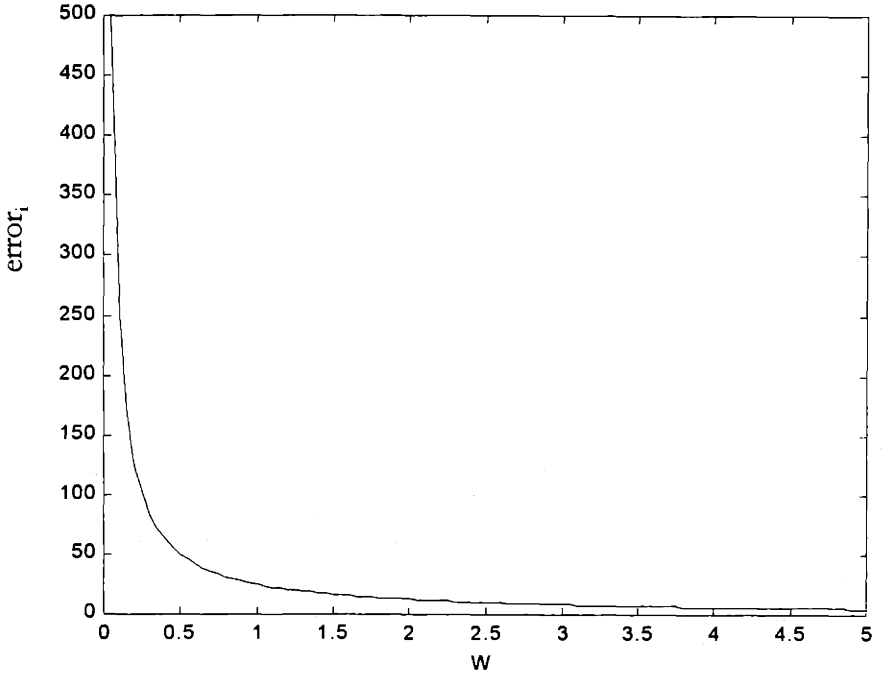
Therefore the error will be,

$$\text{error}_i = E\left[ \left( \hat{\lambda}_{t^*} - \frac{1}{W}\int_{t^*-W}^{t^*} \lambda_\tau d\tau \right)^2 \right] = \frac{E[\lambda_t]}{W}. \tag{61}$$

In this result, $E[\lambda_t]$ is the steady state expected value of the arrival rate, which can be approximated by simulating the appropriate MS system with the optimal pricing policy and full information about the current state of the system.

The error will decrease monotonically as W increases. This is because as W increases, the system will have more data about the random variable and then can estimate the Poisson parameter of this variable more accurately. The error is displayed graphically as a function of the window length, W, in Figure 13.

Figure 13: Graph of $error_i$ as a function of the fixed window length, W. System parameter used: $E(\lambda_\tau)=25$.




ii. Change of state error

This error is caused by the system changing states during the window. If the system changes states, $q$, between time t*-W and t*, there will be an associated estimation error. This error can be calculated by,

$$\begin{aligned} \text{error}_{ii} &= E\left[\left(\lambda_{t^*} - \frac{1}{W}\int_{t^*-W}^{t^*}\lambda_\tau d\tau\right)^2\right] \\ &= E\left[\left(\frac{1}{W}\int_{t^*-W}^{t^*}(\lambda_{t^*} - \lambda_\tau)d\tau\right)^2\right]. \end{aligned} \tag{62}$$

This equation can then be broken down using conditional probabilities. If the time since the system last changed states is called T,

$$T = t^* - t_{\text{last change in q}}, \tag{63}$$

the error can be broken down into,

$$\begin{aligned} \text{error}_{ii} &= E\left[\left(\frac{1}{W}\int_{t^*-W}^{t^*}(\lambda_{t^*}-\lambda_\tau)d\tau\right)^2\right] \\ &= \frac{1}{W^2}\Bigg(P(T<W)E\left[\left(\int_{t^*-W}^{t^*}(\lambda_{t^*}-\lambda_\tau)d\tau\right)^2\middle| T<W\right] \\ &\quad + P(T>W)E\left[\left(\int_{t^*-W}^{t^*}(\lambda_{t^*}-\lambda_\tau)d\tau\right)^2\middle| T>W\right]\Bigg). \end{aligned} \tag{64}$$

If the system did not change state in the interval (t*-W, t*), then T>W, and there will be no error,

$$E\left[\left(\int_{t^*-W}^{t^*}(\lambda_{t^*}-\lambda_\tau)d\tau\right)^2\middle| T>W\right] = 0. \tag{65}$$

From these results, equation (64) can be simplified to,

$$\text{error}_{ii} = \frac{1}{W^2}P(T<W)E\left[\left(\int_{t^*-W}^{t^*}(\lambda_{t^*}-\lambda_\tau)d\tau\right)^2\middle| T<W\right]. \tag{66}$$

If the system does change state, there will be an error associated with that change of state. The assumption is made that the length of the window, W, will be chosen to be short enough so that the probability of the system changing states more than once inside the

interval (t*-W, t*) will be negligible. If the state of the system is assumed to change exactly once in the interval (t*-W, t*) then,

$$|\lambda_{t*} - \lambda_{\tau}| = \begin{cases} j & t*-W < \tau < t*-T \\ 0 & \text{otherwise} \end{cases}. \tag{67}$$

The probability of the system changing state in the interval of time, (t*-W,t*) can be determined in terms of the exponential distribution with parameter, *b*,

$$f_l(l_1) = \begin{cases} be^{-bl_1} & l_1 > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{68}$$

The parameter, *b*, is the total state transition rate from a state *q* to another state. The total transition rate will depend on the state that the system is in currently. If $q$=-2 or $q$=2, then the transition rate will be *a*. If $q$ = -1, 0, or 1, the transition rate will be 2*a*. In order to simplify the system, a constant total transition rate will be chosen, regardless of the state of the system. This will allow the system to have a fixed-width window regardless of the current state, *q*, of the system. A reasonable approximation for *b* is,

$$b \approx 2a. \tag{69}$$

Thus,

$$P(T < W) = 1 - e^{-Wb} \approx 1 - e^{-2aW}. \tag{70}$$

By using the assumption that the system changes state at most once in the time interval (t*-W, t*) and equation (67), the required expectation can be simplified to,

$$E\left[\left(\int_{t^*-W}^{t^*}(\lambda_{t^*}-\lambda_\tau)d\tau\right)^2\middle|T<W\right]=j^2E\left[(T-W)^2\middle|T<W\right]. \tag{71}$$

Conditioned on having a single event in an interval of length W, the location of that even is uniformly distributed. Thus, W-T is uniform on [0,W] and,

$$E\left[(T-W)^2\middle|T<W\right]=\frac{W^3}{3}. \tag{72}$$

Substituting the results of equations (72), (71), and (70) into equation (66) leads to,

$$\begin{aligned}\text{error}_{ii} &= \frac{1}{W^2}P(T<W)E\left[\left(\int_{t^*-W}^{t^*}(\lambda_{t^*}-\lambda_\tau)d\tau\right)^2\middle|T<W\right] \\ &= \frac{1}{W^2}\left(1-e^{-Wb}\right)\frac{j^2W^3}{3}\end{aligned} \tag{73}$$

The change of state error can finally be evaluated as,

$$\text{error}_{ii}=\frac{j^2W\left(1-e^{-2aW}\right)}{3} \tag{74}$$

Analyzing this equation as a function of the window length, W, shows that if W is set to 0, there is no error. This makes sense because if the length of the window is zero, there should be no error associated with a change of state in the interval (t*-W, t*). As W increases, the error will increase monotonically. Equation (74) is only a valid approximation

for values of W where the probability of having two or more changes of state in the interval ($t^*$-W, $t^*$) is small. This is the range of values for W where,

$$W << \frac{2}{b} \approx \frac{1}{a}. \tag{75}$$

Figure 14: Graph of $error_{ij}$ as a function of W. System parameters used were a=0.5 and j=10.

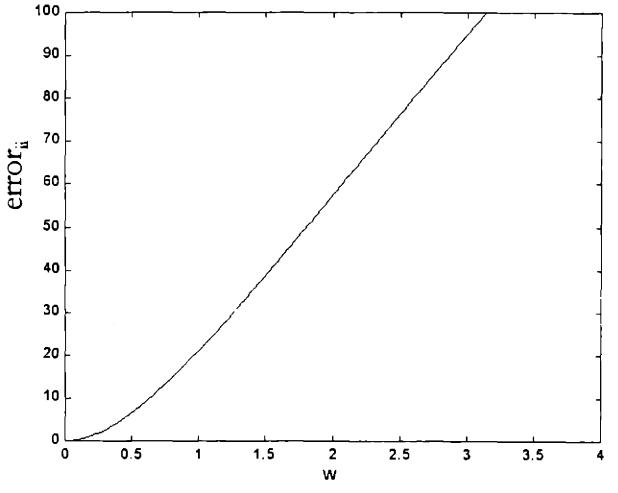
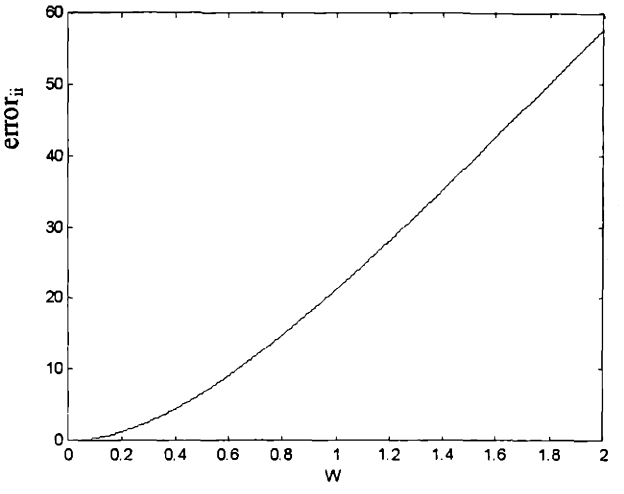


Figure 15: Zoomed in version of Figure 14, showing the relevant range for W.




Minimizing total error

The total error will be the sum of $\text{error}_i$ and $\text{error}_{ii}$,

$$\begin{aligned}\text{error} &= \text{error}_i + \text{error}_{ii} \\ &= \frac{E(\lambda_t)}{W} + \frac{j^2 W\left(1 - e^{-2aW}\right)}{3}.\end{aligned} \tag{76}$$

To optimize the length of the window, W*, the total error must be minimized with respect to W,

$$\min_W(\text{error}) = \min_W\left(\frac{E(\lambda_t)}{W} + \frac{j^2 W\left(1 - e^{-2aW}\right)}{3}\right). \tag{77}$$

The value of W that minimizes the error can be found numerically. Figure 16 shows the total error as a function of W for an MS system. The system parameters were $a$=1, $j$=10,

and $E(\lambda_i)=20$. The value of $W^*$ that minimized the error in this system was $W^*=0.736$. Note that this solution for W satisfies the requirement of equation (75). This means that the solution for the optimal window length is consistent with the assumption made of no more than one change of state for the duration of the window, W.

Figure 16: Graph of the total error as a function of W for an MS system.

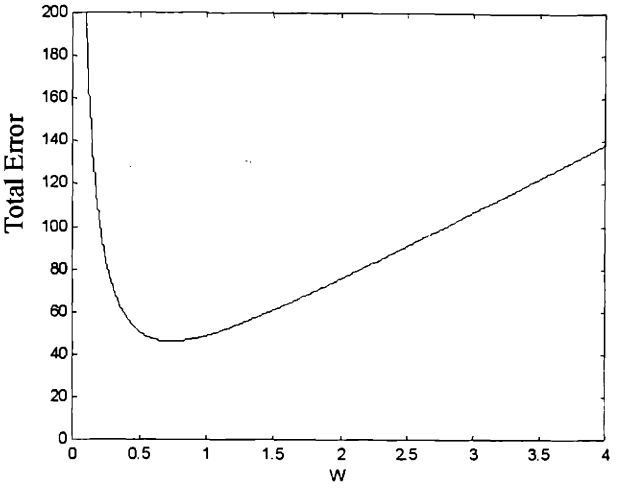
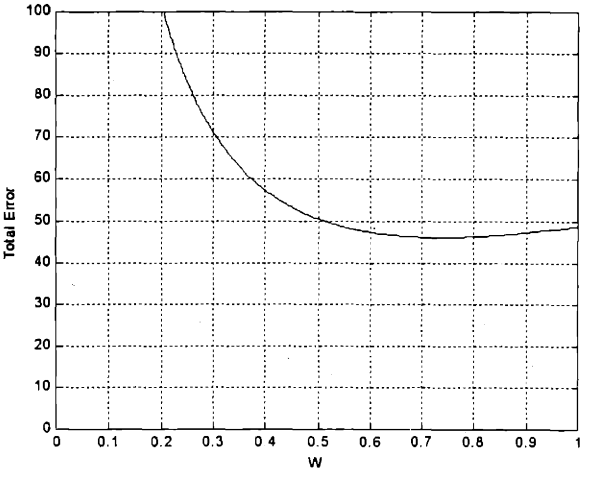


Figure 17: Zoomed in version of Figure 16, minimization of the total error showing the relevant range.




Observations about the optimal length, W*

There are many interesting observations on the relation between the optimum value of W found in equation (77) and the different system parameters. Some of these observations are discussed below.

1. *For most systems, increases in the state change rate, a, will decrease the window length, W.*

If the state change rate is small, increasing it will cause the change of state error ($error_{ii}$) to have a higher slope in the range of interest and the total error will be minimized at a lower value for W. However, if the state change rate is sufficiently large, the window length may increase. This is because the state change error ($error_{ii}$) will already have a relatively flat slope for the optimal value of W and increasing the state change rate may decrease the slope at W instead of increasing it.

2. *Increasing the jump distance, j, will decrease the window length.*

If the jump distance, $j$, is increased, the slope of the change of state error ($error_{ii}$) will increase for all W. This will again cause the total error to reach its minimum value earlier.

3. *Increasing both $E(\lambda_i)$ and $j$ by a common multiplicative factor, will decrease the window length.*

If both $j$ and $E(\lambda_i)$ are increased by a factor of $x$, the slope of the state change error ($error_{ii}$) will increase by $2x$ and the slope of the random variable estimation error ($error_i$) will increase by $x$. Since the slope of the state change error will have a larger change in magnitude, the error will be result in a lower value for W* than before.

Implementing the optimal rectangular window

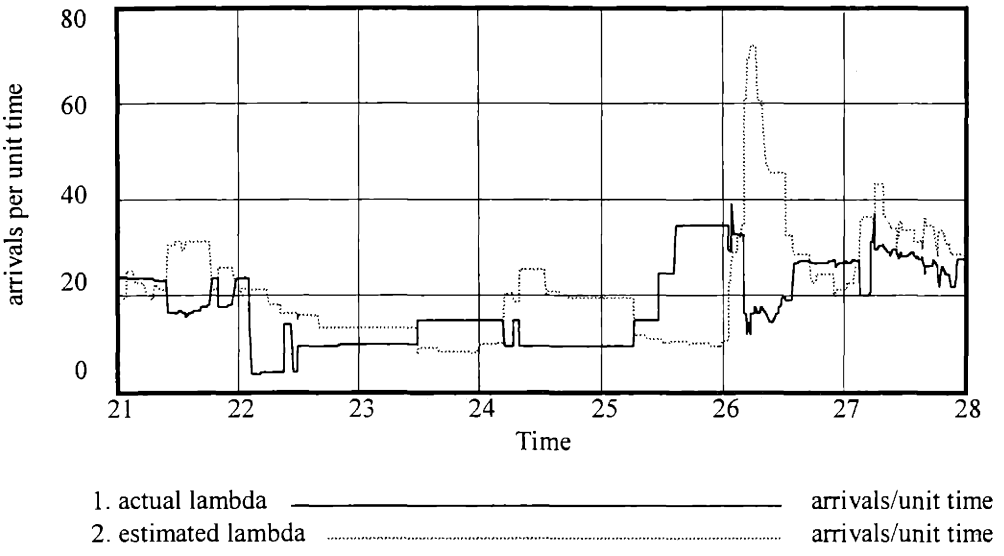
Implementing a fixed length window in an MS system has the same problems as in an SS system. The amount of memory required is large since the maximum memory that is required is unbounded. As in an SS system, to avoid this problem, a fixed number of arrivals estimation can be used, instead. This type of estimation is similar to the fixed length window but the amount of memory required is bounded and constant. An optimal $k_{fixed}$ can be inferred from the optimal W* by,

$$k_{fixed} = E(\lambda_i)W^*. \tag{78}$$

Using this technique introduces some additional error in estimating $\lambda$, which is shown in Figure 18. Price interpolation of the optimal pricing policy (which is described on page 80) was used to determine the price from the estimate of the arrival rate. The number

of arrivals technique tends to overemphasize shorter arrival lengths and therefore can overestimate the actual arrival rate when several arrivals occur in a short amount of time. However, the results (which will be introduced later) indicate that this overestimation is not a large problem.

Figure 18: Comparison of $\lambda$ versus the estimate of lambda ($\hat{\lambda}$) for an MS system using a rectangular window with $k_{fixed}=15$ and price interpolation. System parameters used were N=30, a=1, j=10, $\lambda_{middle}=50$, and $\lambda_1=5$, and $\mu=1$.




## THE EXPONENTIAL WINDOW

An exponential window can be used instead of a rectangular window. An exponential window has the impulse response,

$$h(t) = Ce^{-Ct}u(t), \tag{79}$$

where C is the smoothness parameter. As C increases, the window will decay faster and as C decreases, the window will decay slower.

If x(t) is once again defined as the signal where each arrival into the system at time $t_{arrival}$ corresponds to an impulse, then,

$$x(t) = \sum_{t_{arrival} \in arrivals} \delta(t - t_{arrival}) . \tag{80}$$

An estimate for the arrival rate at time, t, can then be determined by,

$$y(t) = x(t) * h(t) . \tag{81}$$

Since h(t) is a causal system, the estimate can be calculated in real time.

There are three main advantages to using an exponential window compared to the rectangular window:

i. *The exponential window requires less memory than the rectangular window.*

An exponential window only requires storing one value, which is the current estimate, $\hat{\lambda}$. This is because an exponential window is memoryless. The current estimate of $\hat{\lambda}$ can be found by discounting the previous estimate by the appropriate factor and adding the new information,

$$\begin{aligned}\hat{\lambda}_{t^*} &= y(t^*) = x(t) * h(t)\big|_{t=t^*} \\ &= \int_{-\infty}^{*} x(\tau)h(t^* - \tau)d\tau \\ &= \int_{-\infty}^{t_p} x(\tau)Ce^{-C(t^*-\tau)}d\tau + \int_{p}^{*} x(\tau)Ce^{-C(t^*-\tau)}d\tau \\ &= \hat{\lambda}_{t_p} e^{-C(t^*-t_p)} + x_{new}(t) * h(t)\big|_{t=t^*}.\end{aligned} \tag{82}$$

where $t_p < t^*$ and $x_{new}(t)$ is,

$$x_{new}(t) = \begin{cases} x(t) & t_p < t \\ 0 & \text{otherwise.} \end{cases} \tag{83}$$

ii. *The exponential window emphasizes the most recent information more than previous information.*

Since the window is shaped like a decaying exponential, the more recent information is weighted more than previous information. This results in a more accurate estimation of λ, since the more recent information is more accurate than the previous information.

iii. *The exponential window does not run into the problems that the fixed number of arrivals technique does.*

The rectangular window cannot be implemented directly because of memory limitations. As previously mentioned, the fixed number of arrivals technique that is used to overcome the memory limitation tends to overemphasize shorter arrival lengths and therefore can overestimate the actual arrival rate when several arrivals occur in a short amount of time. This was shown in Figure 18. However, the exponential window can be implemented without memory limitation issues and does

not result in this overestimation that can occur from the fixed number of arrivals technique. This can be seen by comparing Figure 18 to Figure 19.

Optimizing the exponential window

The same technique that was used to optimize the rectangular window parameter, W, can be used to optimize the exponential window parameter, C. The error can be expressed as the sum of the random variable estimation error (i) and the change of state error (ii) since the cross term (the covariance) is zero,

$$E\left[\left(\hat{\lambda}_{t^*}-\lambda_{t^*}\right)^2\right]=\underbrace{E\left[\left(\hat{\lambda}_{t^*}-C\int_0^\infty \lambda_{t^*-\tau}e^{-C\tau}d\tau\right)^2\right]}_{i}+\underbrace{E\left[\left(\lambda_{t^*}-C\int_0^\infty \lambda_{t^*-\tau}e^{-C\tau}d\tau\right)^2\right]}_{ii}. \tag{84}$$

i. Random variable estimation error.

Using the same line of reasoning as with the rectangular window,

$$\begin{aligned}\text{error}_i &= E\left[\left(\hat{\lambda}_{t^*}-C\int_0^\infty \lambda_{t^*-\tau}e^{-C\tau}d\tau\right)^2\right]\\ &= E\left[E\left[\left(\hat{\lambda}_{t^*}-C\int_0^\infty \lambda_{t^*-\tau}e^{-C\tau}d\tau\right)^2\middle|\{\lambda_t\}\right]\right].\end{aligned} \tag{85}$$

The estimate for $\hat{\lambda}_{t^*}$ can be expressed as,

$$\hat{\lambda}_{t^*}=C\int_0^\infty e^{-C\tau}dA_{t^*-\tau}, \tag{86}$$

where dA is the area under the impulse (from x(t)) when an arrival occurs. The error can then be evaluated as,

$$\begin{aligned}\text{error}_{i} &= E\left[E\left[\left(\hat{\lambda}_{t^*} - C\int_0^{\infty}\lambda_{\tau}e^{-C\tau}d\tau\right)^2\middle|\{\lambda_t\}\right]\right] \\ &= E\left[C^2\int_0^{\infty}e^{-2C\tau}E\left[(dA-\lambda_{\tau}dt)^2\right]\right] \\ &= E\left[C^2\int_0^{\infty}\lambda_{\tau}e^{-2C\tau}d\tau\right] \\ &= \frac{CE[\lambda_t]}{2}.\end{aligned} \tag{87}$$

ii. Change of state error

The change of state error can be defined as,

$$\begin{aligned}\text{error}_{ii} &= E\left[\left(\lambda_{t^*} - C\int_0^{\infty}\lambda_{t^*-\tau}e^{-C\tau}d\tau\right)^2\right] \\ &= E\left[\left(C\int_0^{\infty}\left(\lambda_{t^*} - \lambda_{t^*-\tau}e^{-C\tau}\right)d\tau\right)^2\right].\end{aligned} \tag{88}$$

If T is assumed to be the time since the last change of state, $q$, occurred, then,

$$\begin{aligned}\text{error}_{ii} &= E\left[\left(C\int_0^{\infty}(\lambda_{t^*} - \lambda_{t^*-\tau})e^{-C\tau}d\tau\right)^2\right] \\ &\approx C^2 j^2\int_0^{\infty}be^{-bT}\left(\int_T^{\infty}e^{-Cq}dq\right)^2 dT,\end{aligned} \tag{89}$$

where we again assume that there can be at most one change of state similar to our analysis for the rectangular window. Simplifying the equation, results in,

$$\begin{aligned} \text{error}_{ii} &= bj^2 \int_0^\infty e^{-bT} e^{-2CT} dT \\ &= \frac{bj^2}{b+2C}. \end{aligned} \tag{90}$$

The total error is then,

$$\text{error} = \frac{\text{CE}[\lambda_i]}{2} + \frac{bj^2}{b+2C}. \tag{91}$$

To find the optimal smoothness parameter, C*, this error must be minimized with respect to C. Taking the derivative and setting it equal to zero leads to,

$$C^* = \frac{1}{2}\left(\sqrt{\frac{4bj^2}{E[\lambda_i]}} - b\right). \tag{92}$$

Observations about the optimal smoothing parameter, C*

The optimal rectangular window length, W* should be inversely proportional to C*. That is, the higher the smoothing parameter, C, the more emphasis the exponential window puts on recent information and the less emphasis it puts on previous information. This is analogous to shortening the rectangular window length, W. As the smoothing parameter is decreased, the exponential window decays slower and more weight is put on past information. This is analogous to increasing the rectangular window length, W. The observations that were made for the rectangular window should hold for the analogous exponential window as well. If changing a system parameter increases the length of the

optimal rectangular window, W*, it should also decrease the optimal smoothness parameter, C*.

Implementing the exponential window

As previously mentioned the exponential window takes less memory to implement than the rectangular window. The exponential window can be implemented from equation (82). However, the problem with doing this is that it requires the estimate to be continuously updated. It is possible to only update the estimate when an arrival occurs if the expected change in the estimate between arrivals is small. This happens if,

$$C << E(\lambda_t). \tag{93}$$

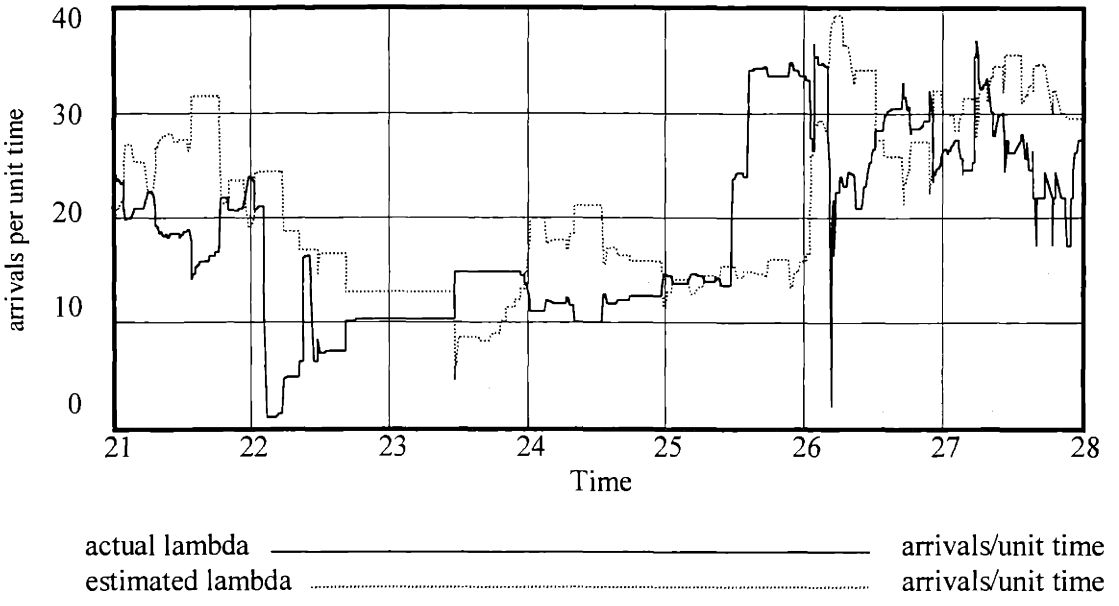
This is the case for many practical systems, and the estimate for the arrival rate can be updated only when an arrival occurs.

The exponential window can then be implemented by,

$$\hat{\lambda}_i = \hat{\lambda}_{i-1} e^{-C(t_i - t_{i-1})} + C, \tag{94}$$

where $t_i$ corresponds to the time when the $i^{th}$ arrival occurred. Figure 19 compares the arrival rate, $\lambda$, versus the estimate using an exponential window with price interpolation of the optimal pricing policy.

Figure 19: Comparison of $\lambda$ versus the estimate of lambda ($\hat{\lambda}$) for an MS system using an exponential window with smoothness parameter C=2.2 and price interpolation. System parameters used were N=30, a=1, j=10, $\lambda_{middle}$=50, and $\lambda_1$=5, and $\mu$=1.




*Determining the appropriate price to charge*

Once the arrival rate, $\lambda$, has been estimated using the appropriate window, an estimate for $\lambda_0$ has to be determined. As with the SS system, $\lambda_0$ can be estimated using the equation,

$$\hat{\lambda}_0 = \hat{\lambda} + \lambda_1 \bar{u}\,, \tag{95}$$

where $\bar{u}$ is the average price charged customers entering the system. This equation can be used because the demand curve is linear with respect to the price, $u$. The average price

should be calculated using the same window type (rectangular or exponential) and the same parameters (for W or C) that were used to calculate $\hat{\lambda}$. From the estimated value of $\lambda_0$, it is possible to estimate the state that the system is in,

$$\hat{q} = \frac{\hat{\lambda}_0 - \lambda_{\text{middle}}}{j}. \tag{96}$$

This estimate for $q$ can take on noninteger values. However, the actual state, $q$, can only take on integer values between [-2,2]. The dynamic pricing policy is only defined for these integer values of $q$ and the estimate, $\hat{q}$ must be discretized in order to determine a pricing policy. Figure 20 and Figure 21 show the resulting estimates, $\hat{q}$, versus the actual states, $q$, using a rectangular window and an exponential window.

Figure 20: Comparison of actual state $q$ versus the estimated state, $\hat{q}$, for an MS system using a rectangular window with $k_{fixed}=15$ and price interpolation. System parameters used were N=30, a=1, j=10, $\lambda_{middle}=50$, and $\lambda_1=5$, and $\mu=1$.

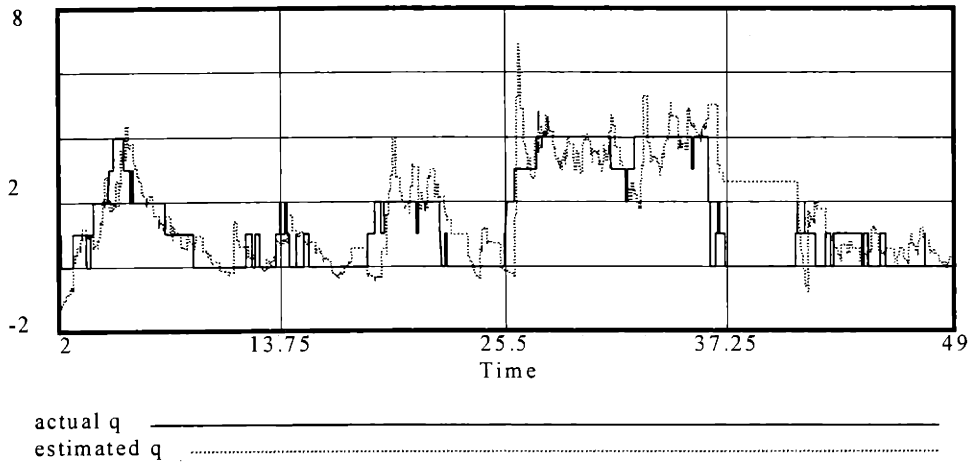
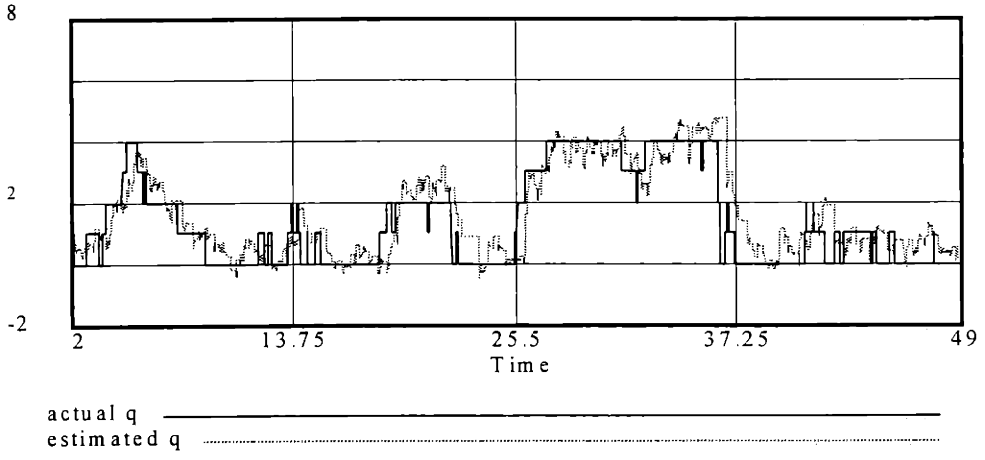



Figure 21: Comparison of actual state q versus the estimated state, $\hat{q}$, for an MS system using an exponential window with C=2 and price interpolation. System parameters used were N=30, a=1, j=10, $\lambda_{middle}=50$, and $\lambda_1=5$, and $\mu=1$.



To determine a price using the estimate, $\hat{q}$, one of two methods can be used:

1. *Rounding the estimated state data*

An integer estimate for the state, $q$, can be determined by simply rounding $\hat{q}$ to the nearest integer inside the range [-2,2]. This can be expressed as,

$$\hat{q}_{round} = \underset{[-2,2]}{round}(\hat{q}). \tag{97}$$

The price the service provider should charge can be determined by using $\hat{q}_{round}$ as an estimate for $q$ and then applying the optimal pricing policy determined by dynamic programming.

This method provides a reasonable estimate for the state of the system. However, if applied to a real system with a long delay in estimation or implementation of the pricing policy, this rounding method could become problematic. Also, if the arrival rate jump distance, $j$, is large, the system response rate could become slow to changes and inaccurate estimates will negatively impact the effectiveness of the system significantly. This is because if the system changes states, it will take a long period of time for the estimate, $\hat{q}_{round}$, to change integer values. Furthermore, an inaccurate estimation of the state could result in $\hat{q}_{round}$ estimating the state to be in the wrong integer value, which would result in the system setting prices that are significantly different from optimal prices.

2. *Interpolating prices between different states*

In order to build a system which is more accurate than rounding for the cases described above, the system can use the estimated $\hat{q}$ and interpolate a price between the integer values for the states that $\hat{q}$ falls between. In implementing this system, linear interpolation for prices was used.

*Comparison of estimation results*

This section will discuss simulation results of the estimation techniques that were introduced. This includes comparing the different window shapes and window lengths, the effect of using interpolation versus rounding, the effect of changing the state change rate, and the total social welfare generated with different pricing policies.

EVALUATING THE NO ESTIMATION POLICIES: THE BASE CASES

If a service provider does not have full information about the system, it can decide not to estimate the current state of the system in determining prices. The Service Provider can model the MS system as an analogous SS system, (with no state, $q$) and set $\lambda_0 = \lambda_{middle}$. The results of applying an SS system pricing policy to an MS system are compared to the results of applying the optimal MS system pricing policy with complete information in Table 8.

There is a large difference in the steady state revenue rate, J, that is generated between an MS system where the optimal 5 state policy and the optimal 1 state policy are used (from comparing rows ii and iii). If the estimation is successful, it should allow the

Service Provider to significantly increase the steady state revenue rate, J, compared to using the 1 state policy. By setting prices based on the SS model, the Service Provider is losing about 12.5% of the optimal revenue rate with full information.

Another observation is that a large difference between the expected revenue and the actual revenue could be a sign of an insufficient model. For example, assume that a service provider models the system as an SS system and has calucted the optimal revenue rate to be J*=121.57 (from row i of Table 8). Setting prices accordingly, the Service Provider will earn significantly less than J* ($J \approx 109$ from row ii of Table 8) if the system is in reality an MS system. This difference could be a flag to the Service Provider that the SS model being used may be insufficient.

Table 8: Evaluation of applying an SS pricing policy to an MS system. The optimal 1 state and 5 state policies were used. MS system has parameters N=30, $\lambda_{middle}=50$, $\lambda_1=5$, j=10, a=1, and $\mu=1$. SS system has $\lambda_0=\lambda_{middle}=50$.

| Simulation Results | | (from dynamic program) J* | J |
|---|---|---|---|
| SS system | | | |
| i | 1 State Policy | 121.57 | 121.49 |
| MS system | | | |
| ii | Applying 1 State Policy | | **108.95** |
| iii | Applying 5 State Policy | 126.72 | **124.56** |

## SIGNIFICANT ADVANTAGES FROM ESTIMATION

The MS system was then simulated where the state, $q$, was unknown and had to be estimated. Table 9 shows the results of using a rectangular estimation window with linear price interpolation. The system parameters were the same as those in Table 8. Solving for

the optimal window length by minimizing the expected error resulted in, W*=0.736 and the optimal number of arrivals was $k_{fixed}$=14.7.

Table 9: MS system simulation results for J, the steady state revenue rate. A rectangular window, with number of arrivals, $k$, was used with linear price interpolation and the optimal MS pricing policy determined by dynamic programming.

| k | J |
|---|---|
| 10 | 117.72 |
| 13 | 118.32 |
| 14 | 118.37 |
| 15 | 118.42 |
| 16 | 118.32 |
| 17 | 118.48 |
| 18 | 118.34 |
| 20 | 118.33 |

The data shows that estimation is able to generate a significant increase of revenue compared to using an SS pricing policy. The Service Provider is now only losing about 5% of the optimal revenue due to estimation error.

The calculated value for $k_{fixed}$ produces a nearly maximum revenue, J. Differences between the calculated and actual optimal number of arrivals could have occurred from

1. Assumptions and simplifications used in the derivation

There were assumptions used in the derivation for the optimal window. Making these assumptions allowed the calculations to be simplified but did introduce some error into the calculations. However, the assumptions should not have significantly affected the optimal window.

2. Lack of simulation convergence

The simulation was run to 200,000 "event time" steps. Therefore, the revenue per unit time, J, that was calculated is only an estimation of the steady state revenue. Factors, such as the random number generator seeds being set to particular values can also introduce error into the system.

The error in determining the optimal number of arrivals, $k$, causes an insignificant difference in J. Also, the data indicates that in this particular system, J is fairly insensitive to the value for $k$ used, over a range of values.

### COMPARISONS BETWEEN PRICE ROUNDING AND INTERPOLATION

Comparing the results of Table 9 (estimation with linear price interpolation) and Table 10 (estimation with price rounding) shows that price interpolation produces higher steady state revenue rates. This was an expected result because the interpolation adds a level of accuracy that rounding is not able to capture. However, the difference between the two methods is insignificant in this case (~1%). For many real-world applications, rounding prices may provide an accurate enough estimate without the additional computation and complexity of interpolation.

Table 10: MS system simulation results for J, the steady state revenue rate. A rectangular window, with number of arrivals, *k*, was used with rounding and the optimal MS pricing policy determined by dynamic programming.

| k | J |
|---|---|
| 12 | 117.13 |
| 13 | 117.39 |
| 14 | 117.3 |
| 15 | 117.13 |

## COMPARISONS BETWEEN RECTANGULAR AND EXPONENTIAL WINDOWS

Results for the rectangular window with price interpolation are shown in Table 9. The same MS system was then estimated using an exponential window. Table 11 shows the results using an exponential window with different values for the smoothness parameter, C, and price interpolation. Comparing these two tables shows that as expected, an exponential window results in a higher steady state revenue rate. However, once again the difference between the steady state revenue rate is insignificant (~1%). Additionally, an exponential window also requires less memory than a rectangular window and is therefore probably the preferred method.

Table 11: MS system simulation results for J, the steady state revenue rate. An exponential window, with smoothness parameter, C, was used with interpolation and the optimal MS pricing policy determined by dynamic programming.

| C | J |
|---|---|
| 1.4 | 118.81 |
| 1.7 | 119.01 |
| 2 | 119.21 |
| 2.1 | 119.11 |
| 2.2 | 119.17 |
| 2.4 | 119.2 |
| 2.6 | 119.23 |
| 2.8 | 119.09 |
| 3 | 119.1 |

The optimal smoothness parameter, C*, found by minimizing the expected error is C*=2.16 .Once again, differences between the optimal C from simulation and the C* that was derived may have been caused by error introduced by assumptions or by simulation error. However, this error does not cause a large difference in J, the steady state revenue rate.

### THE EFFECT OF THE STATE CHANGE RATE ON ESTIMATION

If the state change rate of the MS system is increased, the effectiveness of estimation should be expected to decline. This is because an increase in the state change rate means the data of previous interarrival times will give a less accurate estimate of the current state.

The data in Table 12 compares the effectiveness of using estimation to the optimal MS pricing policy with full information and the SS pricing policy in an MS environment. For estimation, an exponential window was used with price interpolation. The system parameters used were $N=30$, $\lambda_{middle}=50$, $\lambda_1=5$, $j=10$, and $\mu=1$.

Table 12: Comparison of steady state revenue rate, J, for MS systems with different state change rates, *a*, and different pricing policies. The optimal MS pricing policy is simulated with the system having full information about the current state, *q*. The estimation policy applied an exponential window, price interpolation, the optimal MS pricing policy, and smoothness parameter C*. The optimal SS and MS pricing policies were determined by dynamic programming.

| | | state change rate (a) | | | |
|---|---|---|---|---|---|
| | | 0.2 | 0.5 | 1 | 5 |
| Steady State Revenue per unit Time (J) | Applying optimal SS Pricing Policy | 105.75 | 107.35 | 108.95 | 114.18 |
| | Applying optimal MS Pricing Policy | 122.11 | 122.91 | 124.56 | 127.53 |
| | Applying Estimation Policy | 120.08 | 119.37 | 119.17 | 117.07 |

| | | | | | |
|---|---|---|---|---|---|
| i | % loss from using the SS Policy (compared to optimal MS Policy) | 13.4% | 12.7% | 12.5% | 10.5% |
| ii | % loss from using the Estimation Policy (compared to optimal MS Policy) | 1.7% | 2.9% | 4.3% | 8.2% |
| iii | % increase from using the Estimation Policy (compared to optimal SS Policy) | 13.6% | 11.2% | 9.4% | 2.5% |

Three observations can be inferred from the data:

1. *An increase in the state change rate results in a decrease in the loss from using an SS pricing policy as opposed to the optimal policy.*

Row i shows that as the state change rate is increased, the percent loss of steady state revenue becomes smaller when compared to the optimal MS pricing policy with full information.

2. *As the state change rate increases, the estimation policy becomes worse at approximating the optimal MS pricing policy with full information.*

Row ii shows that as the state change rate increases, the percentage difference in steady state revenue between the optimal policy and the estimation policy increases. This is

because when the state change rate is low, the estimation will give a more accurate estimate of the current state than when the state change rate is high.

3. *As the state change rate increases, the effectiveness of the estimation policy over the SS policy decreases significantly.*

Row iii shows that when the state change rate is low, a significant amount of excess revenue can be generated by using estimation. However, as the state change rate increases, the effectiveness of estimation versus the SS pricing policy diminishes rapidly and past a certain point, will cause an insignificant difference in the steady state revenue.

As expected, the data shows that if the state change rate is high, the value of perfect information is higher than if the state change rate is low. In a real system, where the service provider may not have perfect information, it is beneficial to use an MS model only if the state change rate is much smaller than the other system parameter rates (the arrival rate and departure rates). If the state change rate is high, no significant increase in revenue will result from estimation, and the service provider can simplify the model of the system and use an SS system instead of an MS system.

## EFFECT ON TOTAL SOCIAL WELFARE

Although the pricing policies discussed in this chapter were aimed at revenue maximization, it is possible to derive the optimal pricing policies to maximize total social welfare instead. However, a revenue maximization policy can in some cases lead to an increase in total social welfare as well. This was shown in chapter 3, page 45 of this thesis.

In an MS system without full information, using estimation can increase the total social welfare generated (as well as the revenue earned by the service provider) compared to using an SS pricing policy. Table 13 shows this for an MS system with system parameters $N=30$, $\lambda_{middle}=50$, $\lambda_1=5$, $a=1$, $j=10$, and $\mu=1$.

Table 13: Comparison of total social welfare generated by different pricing policies. The policies SS and MS pricing policies are the optimal pricing policies determined by dynamic programming. The MS pricing policy assumed the system has full information. The estimation policy uses an exponential window with C=2.2 and price interpolation with the optimal MS pricing policy.

| | | J | Total Social Welfare |
|---|---|---|---|
| Steady State Revenue per unit Time (J) | Applying SS Pricing Policy | **108.95** | **162.59** |
| | Applying MS Pricing Policy | **124.56** | **171.56** |
| | Applying Estimation Policy (C=2.2) | **119.17** | **165.24** |

*Chapter 5*

# CONCLUSIONS

This chapter will review the objectives that were set out in the introduction. It will also review some of the conclusions that were reached for the optimal dynamic congestion-dependent pricing policy and some of the practical techniques that can be used in analyzing, modeling, and setting prices for real systems.

## Objectives

Different models were introduced and studied for optimal congestion-dependent (dynamic) pricing for a service provider. There were two main objectives for the analysis:

1. *To develop an optimal pricing method that maximizes performance for a network under various system models.*

The service provider is considered to be a monopolist who can set prices without worrying about competition or substitution effects. A price is charged to users when they enter the system (a connection-fee). Prices are set based on the level of congestion in the system and the number and type of users already in the system. The pricing policy that maximizes the performance was then found using dynamic programming. Although the steady state revenue rate generated for the service provider was used as the measure for performance of the network, total social welfare could be used as the measure of

performance. The policy that will optimize performance is a dynamic congestion-dependent pricing policy.

In this thesis, three models were examined in detail.

a) **The single state, single user-class (SS) model.**

The model assumes all users have similar requirements and characteristics. Therefore, the users can be grouped into a single user-class. Furthermore, users enter the network according to a Poisson process and the demand function, which is a linearly decreasing function of price, will remain constant through time.

b) **The single state, multiple user-class (SM) model.**

Users are grouped into different classes based on their requirements, characteristics, and demand functions. Prices can then be set to maximize performance. The model also allows a service-provider to differentiate between various types of products and services that use the common resource of bandwidth. This could include:

i. Different services, such as Internet service and video conferencing.

ii. Different qualities of connection options, such as an analog modem or cable modem.

iii. Different market segmentation strategies.

Determining the optimal dynamic pricing policy when there are a large number of possible user-classes and users is difficult because of memory requirements of the dynamic program. However, simplifications can be made to help in analytically evaluating the system.

c) **The multiple state, multiple user-class (MM) model.**

The model considers the case where the arrival rate can drift into different states as a Markov chain. The service provider can optimize the expected revenue rate by taking this possibility of "drifting" into other states into account when setting prices. It makes sense to use an MM model only when the probability of drifting into another state is small and the difference in arrival rates between the different states is significantly large. The multiple state, single user-class (MS) model was introduced as a simplified version of the MM model to aid in analysis. For the case where the probability of drifting is small, the MM pricing policy for a system in state q will resemble the SM pricing policy for the analogous SM system.

2. *To develop insight from the optimal pricing method that a service provider could use in developing a congestion-dependent pricing policy in practice.*

Implementing the optimal dynamic congestion-dependent pricing policy is difficult in practice because it is complex, requires memory of the present state of the system, and creates a large amount of uncertainty in prices for users. Users may not feel comfortable with a such a large amount of uncertainty in prices and the service provider may not want to

use a system that is complex to operate. Additionally, implementing the optimal policy requires that the service provider know the system parameters and the current state of the system, which may not necessarily be the case. The optimal pricing policy was analyzed in order to build intuition about how to implement a pricing policy that increases revenue in a real system. Although such a pricing policy may not be optimal, if it results in only a modest loss in performance, but at the same time is less complex, less computationally intensive, and more acceptable to users compared to the optimal dynamic pricing policy, it may be a better policy to implement.

## Practical Approximation and Estimation Techniques for Real Systems

It is useful to categorize real systems into one of two categories: systems which are over-utilized or systems which are underutilized. For underutilized systems, prices can be set as if there were no limiting bandwidth constraint. For these systems, congestion-dependent pricing will not cause a significant increase in the performance of the system. For over-utilitzed systems, congestion-dependent pricing will lead to a large increase in the performance of the system.

### *Static versus dynamic congestion pricing*

Although dynamic congestion-dependent pricing will maximize the performance of the system, it may not be a practical policy to implement in a real system because it causes a large amount of uncertainty in prices and is more complex to implement. Static congestion-dependent pricing (which maximizes the performance of the system with the price constant

regardless of the number of users in the system) seems to be an acceptable alternative to dynamic programming. Static pricing results in less price uncertainty for users and an easier system to implement for the service provider. Although static pricing will not result in the optimal performance like dynamic pricing, may come close. This is shown by Paschalidis and Tsitisiklis. They conclude that, "... the static policy offers its substantial implementation advantage at a modest performance cost," (p.21), [7].

*Modeling the network as a single state versus a multiple state system*

It may be useful to model a real network as a multiple state system if the arrival rate is believed to drift or if the system can go into a state of shock. However, it only makes sense to model the network as a multiple state system if it changes states infrequently (the expected time between changes in state is small). If the system changes states too frequently, it becomes difficult to accurately estimate the current state of the system and the additional complexity of the system does not justify the modest increase in performance. If the model used for the network is a single state model and the actual performance is significantly lower than the calculated theoretical performance, it may be a signal that a multiple state system should be used. The disadvantages of using a multiple state model are that it causes more uncertainty in prices and it is significantly more complex than a single state model.

*Using static congestion-dependent pricing in a multiple state system*

In a multiple state system, the dynamic policy can be simplified to a set of static pricing policies with a different static price for each state of the demand function. In the

optimal static policy, the price charged in each state will depend on the other states that the system could transition into. However, if the expected time of a state change is large, this effect on the pricing policy will be negligible. The static pricing policy in each state can be solved for independent of the other states, as if each state corresponds to a separate single state system.

*Estimating the current state in a multiple state system*

In a multiple state system, the current state of the system can be estimated by using either a rectangular window or an exponential window. An exponential window is preferred to a rectangular window because it requires less memory to implement and it results in a more accurate estimate. However, the difference in performance when using the different window shapes is small. The smoothness parameter, C, for the exponential window should be chosen to minimize the total error of estimation. This value can be found by using equation (92), but performance seems to be insensitive to C over a range of values. However, C should be chosen so that there are a large number of data points in the window before it decays significantly but, by the expected state change time, the value of the window must have decayed to a negligible value.

Once an estimate for the state of the system has been calculated, a pricing policy can be determined by either rounding or interpolating the arrival rate to determine the price that should be charged. Although interpolation results in higher system performance, rounding will result in only a modest decrease in value if the expected time between a change in state

is long. Rounding has the benefits that it is less computationally intensive and causes less price uncertainty than interpolation (if multiple state, static pricing is used). With rounding, the price charged will be in a discrete set of values (one price for each possible state of the system). With interpolating, the price charged is in a continuous range of values.

## Extensions

There are many areas of additional study that can be conducted. Some of the interesting areas for additional study are:

- *Model the effects of competition and substitution on the pricing policy*

The model assumed that the service provider has a monopoly over the system and can set prices without worry of competition or substitution. The current landscape indicates that this is an idealized assumption, as there are many different service providers and different suplementary products available at different prices to users. One area to extend the model is to take into account the effects of competition and substitution on the pricing policy in a system with more than one service provider.

- *Use profit instead of revenue as a measure of performance*

In the case where the marginal cost realized by the service provider is negligible, the revenue and profit maximization policies will be the same. However, in the case where the

marginal cost is not negligible, the profit maximization and revenue maximization pricing policies will be different. It is possible that in the future the service provider will be charged a fee for access to the Internet. The fees charged can be modeled as a marginal cost and will have to be passed on to users. One area of additional study would be to see how the pricing policy differs when using profit for a measure of performance where the marginal cost incurred by the service provider is not constant.

- *Determine the real demand curve, the real departure probability distribution function, and user psychology towards price uncertainty*

A demand function of $\lambda(u)$ was modeled as decreasing linearly with respect to price. There is no data to support this assumption. Further study needs to be done to determine the actual demand function with respect to price. Also, the departure PDF was assumed to be exponential. Although telephone calls have shown to have an exponential departure PDF, this has not been shown for network connections. Additionally, the findings of this thesis are based on certain assumptions about user psychology towards price uncertainty. These assumptions need to be studied in more detail to determine the best pricing policy to implement.

- *Optimize performance for a system with other price structures*

In this thesis the system was modeled as charging users a connection-fee. In a real system, it may make more sense to use a connection-time charge since using a connection-

fee would probably result in users staying connected to the network for long periods of time. This is because once connected to the network, there is no incentive for a user to disconnect. In an over-utilized system, it is necessary for the departure rate to be large to increase turnover. A connection-time charge may result in a higher departure rate than a connection-fee charge. The analysis of the connection-fee system is in many ways similar to the analysis of a system with a connection-time charge. The two systems can be equated by setting the connection time charge equal to the connection-fee charge divided by the expected length of a connection session. However, there are some subtle differences that need to be explored, such as how the connection time fee affects the length of a connection session and the departure rate PDF.

In addition to a single price structure, a system where a two-charge structure is used would be interesting to analyze. For instance, how a service provider should maximize performance using both a flat fee and a connection-time charge would be an interesting area to study.

# REFERENCES


[1] Andres, San and Francis Chow, Jun Shu, et all, *Network Pricing, Costs, and Settlements*, May 1997.

[2] Bertsekas, Dimitri P., *Dynamic Programming: Deterministic and Stochastic Models*, Prentice-Hall, Englewood Cliffs, New Jersey, 1987.

[3] Drake, Alvin W., *Fundamentals of Applied Probability Theory*, McGraw-Hill, New York, 1967 (Reissued 1988).

[4] Larson, Richard C. and Amedeo R. Odoni, *Urban Operations Research*, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

[5] Odlyzko, Andrew, "A Modest Proposal for Preventing Internet Congestion", AT&T Labs – Research, September 1997.

[6] Oppenheim, Alan and Alan Willsky, *Signals and Systems*, Prentice Hall, Upper Saddle River, New Jersey, 1997.

[7] Paschalidis,Ioannis, and John N. Tsitsiklis, *Congestion-Dependent Pricing of Network Services*, October 1998.

[8] *The Washington Post*, "An interview with Stephen M. Case, CEO of America Online", December 1997.