Pricing with Bandwidth Guarantees for Clients with multi-ISP Connections

Rohit Tripathi and Gautam Barua
Department of Computer Science and Engineering
Indian Institute of Technology, Guwahati
Guwahati-781039, India
{t.rohit,gb}@iitg.ernet.in

ABSTRACT

The increase in internet coverage and decrease in internet access price has resulted in demand for good internet service. Clients want some guarantee in internet access quality. In this paper, we present a model in which clients are guaranteed connection and bandwidth and if clients do not get the service they request, the service provider pays a penalty to the clients. We consider a system of internet clients with multiple internet service provider (ISP) connections to a set of ISPs. When a client arrives, an ISP has to decide whether to accept the client, and then the price to charge from the client for the duration of its connection. Rejection of a client results in a penalty and delay in getting the requested bandwidth also incurs a penalty. We assume a Poisson arrival process with the rate of arrival sensitive to the price being charged. A client requests bandwidth for a time that is exponentially distributed, then the client is idle for a time that is also exponentially distributed; and then either the client departs or requests bandwidth again after the idle period is over. A service provider tries to maximize its income by charging appropriate prices based on its current state and deciding whether to accept more clients or not. Since penalties are imposed, such solutions also automatically balance load among service providers, and so the quality of service to clients improves. We present solutions that maximize the income of service providers. The solutions are then compared using simulation. Simulation results show that our solutions significantly improves quality of service of clients and increases the income of service providers as compared to a simple heuristic based solution that is otherwise could to be used.

1. INTRODUCTION

Many mobile phones now have the capability to install multiple subscriber identity modules (SIMs). This opens up the possibility of a mobile phone client choosing to connect to one of his internet service providers (ISPs) dynamically based on the nature of services being offered by the ISPs he has SIMs of. Users can now demand better quality of service and get something better than the fixed pricing models with no guarantees currently prevalent in the Internet world. We present a scheme in which clients are given bandwidth and connection guarantees, and penalty is paid to clients if assured service is not provided. Price is used to attract or deter clients from connecting. So if a client makes a request to connect to a network, a penalty is paid if the connection is refused (connection guarantee), and once a connection is established at the offered price, any delay in providing the negotiated bandwidth also results in a penalty proportional to the delay (bandwidth guarantee). The scheme results in load balancing among the active ISPs automatically and this benefits the clients and the ISPs. If a service provider gets congested, clients will shift to another service provider which is not congested. However shifting at arbitrary times will result in disconnecting all current channels, as the IP address of the client will change. So we assume that such shifts will take place only at the behest of the user when all current connections are closed.

The rest of this paper is organised as follows. Section 2 describes related work in this area. In Section 3, we introduce the problem. In Section 4, we present our model. Sections 5 gives a solution. In section 6, a simple heuristic is described and it is used as a basis of comparing the usefulness of our scheme. In section 7, our solution and the random solution are compared using simulations. We conclude the paper in section 8.

2. RELATED WORK

[6] provides a comprehensive survey of various pricing schemes. In a static pricing scheme, a fixed price is offered and clients pay this price for service (with a flat price ([7]), or depending on usage ([2]), or time-of-day ([1]), on pre-defined priorities ([10])). Service providers cannot change their earnings based on demand. These limitations are handled by dynamic pricing schemes where prices can be changed at any time by service providers. Dynamic pricing schemes can also handle congestion that results from dynamic bandwidth demand of clients. Besides controlling prices to maximize income, there is also a need to provide guarantees to users on different service parameters. Underlying most of these dynamic pricing schemes is the effort to provide a certain degree of quality of service. This could be by defining priorities (and so also quality of service) and charging different prices at different priorities ([3]), by increasing the price when congestion occurs ([4]), or by auctioning the available bandwidth and

giving it to the highest bidder ([11]). In [5], authors have proposed a scheme which has some features similar to our scheme. The quality of service being guaranteed is bandwidth and penalties are imposed when the guarantees cannot be met. A client negotiates a rate for some assured bandwidth. When the client does not want to use some of the assured bandwidth, he can request the service provider to reduce the assured bandwidth and thereby pay a discounted rate for the bandwidth used. When a client wants that guaranteed bandwidth back, the service provider has to give it to him. In case it is unable to do so, a penalty is imposed. Although they mention delay dependent penalty, they use a fixed penalty to reduce complexity. Since prices are fixed at admission time, and there are no connection guarantees, the policy is to do appropriate admission control to maximize income. The problem is then to find a suitable trunk reservation scheme where spare capacity is kept to handle bandwidth return requests, and admission is done if sufficient bandwidth can be reserved for the incoming client. They propose a heuristic to find such a reservation scheme. Their scheme is more suitable for leased connections or for VPNs, but not suitable for a retail environment like ours.

Each of these proposals has a different model of user interaction and currently it is not clear what will be an acceptable user model. No proposal considers the possibility of giving a user the choice of connecting to competing providers. No proposal talks about providing connection guarantees to users with penalties in case a connection is refused. Our view is that a realistic but simple model should be used to cater to retail internet users. Such a model should have connection guarantees and bandwidth guarantees with penalties in case of failure to provide these guarantees. Because of the complexity of the problem, we restrict ourselves to only providing bandwidth guarantees and no other guarantees such as response time. Finally, users must have some degree of certainty in pricing and a completely dynamic pricing scheme is unlikely to be acceptable to users. So we propose a dynamic pricing scheme where the price is fixed at the time of admission into the system.

In [9], we presented such a scheme that provides connection and bandwidth guarantees to clients. There are two ISPs and clients have options to connect to both the ISPs. We presented Nash equilibrium solutions that maximize the income of service providers. However, the Nash equilibrium solutions have high time and space complexities and the complexities increase by increasing the number of ISPs. In this paper, we present a non game theoretic scheme and solution whose time and space requirement is less and is also independent of the number of ISPs.

3. THE SYSTEM ARCHITECTURE

The interaction between a client and service providers is shown in Figure 1. When a client wishes to connect, he requests a price for some bandwidth in the range of '1 to e' units (we assign probabilities $G(i)$ for i units in our model (see below)). The ISP then offers him a price (per unit of bandwidth consumed). The user obtains such prices from all the ISPs he can connect to and makes a connection request to the ISP offering the lowest price. In case more than one ISP is offering the lowest price, the client randomly chooses one of them. The price offered to a user remains the same during the entire duration the user is connected to the ISP. So this gives a connection of a fixed number of units of bandwidth at a fixed price to a client for the entire duration of his connection to the internet. When a connection request is made to an ISP, the ISP decides whether to accept the connection or not. If it refuses the connection, the client is paid a fixed penalty and no further requests are entertained from that client till the price offered changes. These steps prevent a client from taking advantage of a congested ISP by making repeated requests to collect penalties. So, when the price is offered, the ISP will offer the highest possible price if it is going to refuse connections. The client will not know if the connection is going to be refused till it actually makes a request and so it may make a request to this ISP only if all other ISPs are also offering the highest price. Once a connection is refused, no further requests are entertained to prevent a client getting multiple penalties. The ISP consults a table where, for each possible state of the ISP, the decisions are given (the decisions are a) whether to refuse or accept a connection, and b) if accepting, the price to offer). We need to populate this table so that the mean income of the ISP at steady state is maximised.

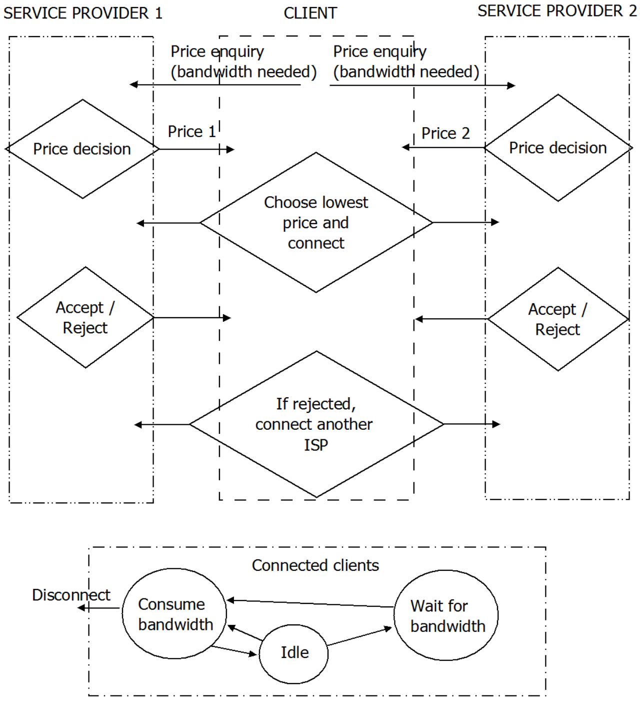

Figure 1: The connection process

Once a client enters an ISP's system, it is assumed that he will require bursts of bandwidth (of the number of units he requested) and then he will be idle for a while and then he will again request for a burst. So a client can be in one of three states while connected: a) consuming bandwidth (he is said to be in session), b) idling (he is in state idle), or c) waiting for bandwidth to be allocated (he is waiting to enter a session). After consuming a burst of bandwidth, the client may leave the system or it may go into the idle state. When a connected client has to wait for a session, he earns a session delay penalty which is proportional to the time he is delayed. The client has to pay for the amount of bandwidth he consumes.

As already mentioned, this scheme introduces competition among ISPs. So on the one hand, an ISP will try to attract as many clients as possible by lowering its price. However, as its available bandwidth decreases, the chance of paying delay penalties increases and so it will slow down the arrival of new clients by increasing the offered price. A stage will be reached when it will be cost-effective to pay a penalty and refuse a client's request to connect. So a lightly loaded ISP is likely to offer a low price, while a heavily loaded ISP is likely to offer a high price. As clients will connect to that ISP offering the lowest price, load balancing among ISPs is built-in in the scheme. From a client's point of view, an assurance that he will not get delayed due to want of bandwidth is given by the ISP in the scheme. To prevent "squatting" (a client remains in idle state for long periods as he got a very good price when connecting, but now the ISPs are all offering higher prices), connections will have to be dropped if idle times exceed a threshold a pre-defined number of times during a connection. The state of an ISP is represented by an array of integers $(m, n_1, n_2, ..., n_e, r_1, r_2, ..., r_e)$. For ease of exposition we group them and express it as (m, N, R) where m represents the number of clients connected. Since a service provider can reject an arriving client, the value of m is within some range from 0 to m_{max} where m_{max} is the maximum possible value of m. e is the size of the arrays N and R and it represents the maximum bandwidth which a client can request. It is assumed that bandwidth is requested in discrete units from 1 unit to e units. N is an array which represents the number of clients in session and its index ranges from 1 to e. So n_i represents the number of clients in session with i units of bandwidth. R is an array which represents the number of clients waiting for a session and its index also ranges from 1 to e. r_i represents the number of users who requested for i units of bandwidth but are queued. So we have a finite state space and our goal is to define the decisions to be taken on an arrival, for each state. So we have a decision matrix $C(m, N, R)$. When the value of $C(m, N, R)$ is zero or less, the client is to be accepted and the price to be charged is $price(-C(m, N, R))$, where is price is an array containing the different prices that can be levied. When $C(m, N, R)$ is 1, the client is to be rejected.

4. THE MODEL

In order to find the values of the matrix, we introduce a simplified model that is amenable to analysis. Further, the size of C is very large for practical ISPs and so it is not feasible to implement. Our model is depicted in Figure 2. The model mirrors the architecture described above, except that it makes the following assumptions:

- Clients arrive at an ISP according to a Poisson process. The mean arrival rate is dependent on the price being offered. It increases with a decrease in price. These rates are assumed to be known a priori. In practice, they will be determined by actually observing the arrival rates and using these rates to make future decisions.

- The service time of a client consuming bandwidth, and the time spent in idle state by a client are both assumed to be distributed exponentially with fixed mean rates. All arrivals and departure processes are therefore "memoryless".

- Among the connected clients the opening of and closing of sessions is modelled as a finite population queuing system. Clients queued for entering a session will in general have different bandwidth requirements and the total bandwidth available is fixed. Therefore, there may not be enough spare bandwidth to service the client first in the queue, but another client behind in the queue could be serviced. So, to ease analysis, the queue servicing discipline is assumed to be "least bandwidth required first". It is asserted (without proof due to lack of space) that this does not impact the steady state mean delay of clients. In actual practice, a first-come-first-serve policy with "queue-jumping" if service is not possible, should be used.

- Rather than associating a bandwidth requirement with each arriving client, in the model it is assumed that clients' bandwidth requirements are determined at the time of moving out of the idle state and the allotted bandwidth is i units with probability $G(i)$. Here too it is asserted that this change does not impact the steady state analysis.

- The state of a service provider is now represented by an integer m, which is the number of connected clients. This reduces the state space considerably. The decision a service provider takes at state m is given by $C_a(m)$. Out of the m clients the expected number in session and the expected number in idle state are estimated as shown below in the analysis. The problem is to find that value of $C_a(m)$, for every m, which maximises the average income of the service provider. Once the solution is obtained, $C_a(m)$ is then used for all $C(m, N, R)$ as given below.

$$C(m, N, R) = C_a(m), \forall N, R$$

Figure 2 depicts the model.

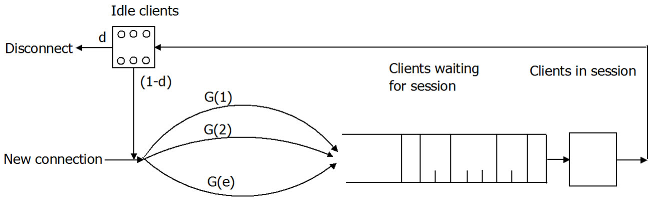

Figure 2: A service provider in our model

5. THE SOLUTION

5.1 Definition of Symbols

Let $\lambda(i)$ be the mean arrival rate of clients that connect to the service provider when the price being charged is $price(i)$. As the price increases, the arrival rate goes down. The actual correlation between the prices and the mean arrival rate is left as inputs to the model. In practice, it is assumed that the mean arrival rates and prices will be set periodically based on historical data. The arrival of clients is modeled as a Poisson process. The session duration for a client consuming i units of bandwidth is modeled by an exponential distribution with a mean service time of $\frac{1}{S(i)}$. The idle duration is also modeled by an exponential duration with mean $\frac{1}{S(0)}$. After the idle duration ends, a client disconnects with probability d or again requests for bandwidth with probability $(1-d)$.

As already mentioned, the state of a service provider is represented as (m, N, R) where m represents the number of clients connected. All the symbols and their definitions are described and shown in Table 1 and Table 2. The definition of those symbols whose values are taken as input are given in Table 1 and the definition of symbols whose values are calculated are given in Table 2. The goal is therefore to find the values of the decision matrix that gives an optimal solution. These values can be found off-line, beforehand.

At steady state, a service provider earns some income per unit time. The decisions that produces the maximum steady state income per unit time are optimal decisions. We do not assume that an optimal decision is unique. There may be multiple optimal decisions and each of those decisions may produce the same income at steady state. The objective is to get any optimal decision. To make the analysis tractable, we consider only the number of clients connected in taking any decision. For a given value of the number of connected clients, the number that will be idle and the number that will be waiting for a session are estimated. So we find the optimal set of values for the array $C_a(m)$ and then map them to the array $C(m, N, R)$.

5.2 Finding solution $C_a()$

The method is to iterate through all possible values of $C_a()$ and in each case find the expected income. The solution is that $C_a()$ which provides the maximum expected income. It is assumed that a service provider charges a low price when no client is connected and increases its price when more clients are connected. Therefore, the number of possible permutations of $C_a()$ which have to be considered is not too high if the number of possible prices are limited. The details of the number of possible permutations of $C_a()$ to be considered is given in section 5.3.

5.2.1 Finding Income

The expected income per unit time is found by multiplying the probability of a service provider being in state m by the income per unit time earned at state m for every possible state m and summing the results. The expected income earned at state m consists of two terms as shown in the equation below: the first term is the loss because of clients waiting for a session. There are waiting(m) clients expected to be waiting and the penalty rate is P(0). The first part of the second term is the expected income from arriving clients. This is the rate at which clients arrive when in state m times the expected number of data units each client requests times the price per data unit when in this state. The second part of the second term is to handle the case when the client is rejected, and this is a loss and is the arrival rate at the highest price times the penalty to be paid.

Table 1: Definition of Symbols

Symbol	Description
$\lambda(i)$	The mean arrival rate of clients at the service provider when $price(i)$ is being charged; i varies from 0 to $T-1$ (unit: number of clients/time)
$G(i)$	Probability of a client requesting i units of bandwidth when he initially has no bandwidth. Here i is an integer ranging from 1 to e.
$\frac{1}{S(i)}$	Mean session duration or idle duration for which a client consumes i units of bandwidth. If i is zero, it is the mean time a client remains idle. If i is non zero, it is the mean time for which a client remains in session consuming i units of bandwidth. (time)
$price(i)$	$price(i)$ is an array which stores the possible prices per unit data that can be charged from an arriving client. The values stored in $price(i)$ are in ascending order. $price(0)$ is the least price and $price(T-1)$ is the maximum. (rupee/data-unit)
T	Number of prices
m_{max}	It is the maximum possible value of the number of clients connected.
B	The total units of bandwidth available with the service provider (data-units / time)
e	Maximum number of units of bandwidth a client can request from his service provider. A client can request 1 to e units of bandwidth (data-units / time; also used as an index to arrays).
d	Probability that a client disconnects immediately after leaving the idle state.
$P(0)$	Penalty per unit time which is paid to a client when he waits for Session entry. (rupee/time)
$P(1)$	Penalty which a client gets when his request to connect is rejected. (rupee)
m	The number of clients connected. It ranges from 1 to m_{max}.
N	An array which represents the number of clients in session and its index ranges from 1 to e. So n_i represents the number of clients in session with i units of bandwidth.
R	An array which represents the number of clients waiting for a session and its index also ranges from 1 to e. r_i represents the number of users who requested for i units of bandwidth but are queued where i ranges from 1 to e.

Table 2: Definition of Symbols whose values are calculated

Symbol	Description
$C(m, N, R)$	The decision a service provider takes on the arrival of a client when the state of a service provider is (m, N, R). When the value of $C(m, N, R)$ is zero or less, the client is to be accepted and the price to be charged is $price(-C(m, N, R))$. When $C(m, N, R)$ is 1, the client is to be rejected. $C(m, N, R)$ is a function in the analysis, but its values are calculated offline and stored in an array C and at run time, it is the array that is consulted. (an integer)
$C_a(m)$	It stores the decisions to be taken at state m in the simplified model. The values of $C(m, N, R)$ are obtained as given below (an integer) $C(m, N, R) = C_a(m), \forall N, R$.
$Income$	Expected income per unit time. (rupee/time)
$Pri(m)$	Steady state probability of m clients being connected when arriving clients have infinite population
$Pr(m, n_1, ..n_e, r_1, ..r_e)$	The steady state probability that n_i clients in session consuming i units of bandwidth and r_i clients queued for session for i units of bandwidth where i is an integer ranging from 1 to e and the total population is finite and is m.
$Waiting(m)$	Expected number of clients waiting for bandwidth when m clients are connected
$I(m)$	Expected number of idle clients when m clients are connected
E_d	It is the expected data transferred by a newly arriving client (data-unit)

$$
\begin{aligned}
Income = \sum_{m=0}^{m_{max}} & Pri(m) \times \\
& [-Waiting(m) \times P(0) \\
+ & \left(\begin{cases} \lambda(-C_a(m))\times & \\ E_d \times price(-C_a(m)) & ,C_a(m) \leq 0 \\ -\lambda(T-1) \times P(1) & ,otherwise \end{cases} \right)]
\end{aligned} \quad (1)
$$

As can be seen, the expected income depends on the decision taken which is the value of $C_a(m)$. So to find the best expected income rate, we have to find the proper values of $C_a(m)$.

5.2.2 Finding $Waiting()$

As the state space has been reduced, the number of clients waiting for bandwidth (for a session) is approximated by calculating the expected number of clients waiting when m clients are connected and this is denoted by $Waiting(m)$. The method to find $Waiting(m)$ is as follows. The expected number of clients waiting when the state is $(m, n_1, n_2, ..., n_e, r_1, ..., r_e)$ is multiplied by the probability that the state is $(m, n_1, n_2, ..., n_e, r_1, ..., r_e)$ when m clients are connected and this is added for every possible value of n_i and r_i, for all i. When m clients are connected, the probability that the state is $(m, n_1, n_2, ..., n_e, r_1, ..., r_e)$ is given by $Pr(m, n_1, n_2, ..., n_e, r_1, ..., r_e)$. The number of clients waiting when state is $(m, n_1, n_2, ..n_e, r_1, ..., r_e)$ is by definition $\sum_{i=1}^{e} r_i$. So

$$
\begin{aligned}
& Waiting(m) = \\
\sum_{(n_1,...,n_e,r_1,...,r_e)} & \left(Pr(m, n_1, n_2, ..n_e, r_1, ..., r_e) \times \sum_{i=1}^{e} r_i \right)
\end{aligned} \quad (2)
$$

When a client is connected, he may be in idle state, in session or waiting for a session. When m clients are connected, the probability that n_i clients are in session and r_i clients are waiting for a session for i ranging from 1 to e is given by $Pr(m, n_1, n_2, ..n_e, r_1, ..r_e)$. If the value of e is 1 (all clients request exactly one unit of bandwidth), Pr() can be obtained using the finite source queueing theory formula[8].

$$
\begin{aligned}
& Pr(m, n_1, r_1) = \\
\frac{m!}{(m - n_1 - r_1)! \times n_1! \times B^{r_1}} & \left\{ \left(\frac{S(0)}{S(1)} \right)^{n_1 + r_1} \right\} \\
& \times Pr(m, 0, 0)
\end{aligned} \quad (3)
$$

and

$$
Pr(m, 0, 0) \quad = \quad \frac{1}{\sum_{n_1, r_1} \frac{m!}{(m-n_1-r_1)! \times n_1! \times B^{r_1}} \left\{ \left(\frac{S(0)}{S(1)} \right)^{n_1+r_1} \right\}}
$$

For values of $e > 1$, we expand the terms of the above equation in a "natural" manner (we are unable to prove that this is an accurate formula), and, for m clients, we use the following expression for steady state probability.

$$Pr(m, n_1, n_2, ..n_e, r_1, ..., r_e) = \frac{m!}{(m - \sum_{i=1}^{e}\{n_i + r_i\})! \times (\sum_{i=1}^{e} n_i)! \times \prod_{i=1}^{e} \left(\frac{B}{i}\right)^{r_i}} \times \prod_{i=1}^{e} \left\{ \left(\frac{S(0) \times G(i)}{S(i)} \right)^{n_i + r_i} \right\} \times Pr(m, 0, 0, ..., 0) \tag{4}$$

The sum of probabilities is 1. Therefore, another equation is

$$\sum_{(n_1,...,n_e,r_1,...,r_e)} Pr(m, n_1, n_2, ..n_e, r_1, ..., r_e) = 1$$

On substituting the value of $Pr(m, n_1, n_2, ..n_e, r_1, ..., r_e)$ from equation 4, we get

$$\sum_{(n_1,...,n_e,r_1,...,r_e)} [\frac{m!}{(m - \sum_{i=1}^{e}\{n_i + r_i\})! \times (\sum_{i=1}^{e} n_i)! \times \prod_{i=1}^{e} \left(\frac{B}{i}\right)^{r_i}} \times \prod_{i=1}^{e} \left\{ \left(\frac{S(0) \times G(i)}{S(i)} \right)^{n_i + r_i} \right\} \times Pr(m, 0, 0, ..., 0)] = 1$$

From this the value of $Pr(m, 0, 0...0)$ is

$$Pr(m, 0, 0, ..., 0) = (\sum_{(n_1,...,n_e,r_1,...,r_e)} \{ \frac{m!}{(m - \sum_{i=1}^{e}\{n_i + r_i\})! \times (\sum_{i=1}^{e} n_i)! \times \prod_{i=1}^{e} \left(\frac{B}{i}\right)^{r_i}} \prod_{i=1}^{e} \left(\frac{S(0) \times G(i)}{S(i)} \right)^{n_i + r_i} \})^{-1} \tag{5}$$

5.2.3 *Finding* $Pri()$

$I(m)$, the expected number of idle clients when there are m clients in the system can be found the same way as $Waiting(m)$. So, the expression is

$$I(m) = \sum_{(n_1,...,n_e,r_1,...,r_e)} \{ Pr(m, n_1, n_2, ..n_e, r_1, ..., r_e) \times (m - \sum_{i=1}^{e}\{n_i + r_i\})!\} \tag{6}$$

Now, the departure rate from the system is $I(m) \times d \times S(0)$ by definition of the model. If we consider this departure process to be memoryless and if we ignore the rejection of clients at connection time (making the arrival process memoryless), then the system can be modelled as a Markov Chain and the global balancing equation can be applied:

$$Pri(m) \times \textit{departure rate} = Pri(m-1) \times \textit{arrival rate} \tag{7}$$

equation where $Pri(m)$ is the steady state probability of m connected clients. The mean arrival rate when there are $m - 1$ clients is λ_{m-1}, and this depends on the price being charged in state $m - 1$. By our definition given in Table 1 this is $\lambda(-C_a(m - 1))$. So we get,

$$Pri(m) \times I(m) \times d \times S(0) = Pri(m-1) \times \lambda(-C_a(m-1)) \tag{8}$$

This can be re-written as

$$Pri(m) = \frac{1}{I(m)} \left(\frac{\lambda(-C_a(m-1))}{d \times S(0)} \right) \times Pri(m-1) \tag{9}$$

Further, the following will hold and the two equations can be used to solve for $Pri(m)$.

$$1 = \sum_{i=0}^{m_{max}} Pri(i) \tag{10}$$

5.2.4 *Finding* E_d

The method to find the approximate data consumption by an arriving client, E_d is given below. When a client connects, he consumes i units of bandwidth for a period with mean $\frac{1}{S(i)}$ with probability $G(i)$ and then becomes idle for a period with mean $\frac{1}{S(0)}$. At the end of this idle period he disconnects with probability d. Otherwise, he again requests for bandwidth with probability $(1 - d)$ and therefore again transfers E_d units of data. So we get,

$$E_d = \sum_{i=1}^{e} G(i) \times \frac{1}{S(i)} \times i + (1 - d) \times E_d$$

This can be re-written as

$$E_d = \sum_{i=1}^{e} \left(\frac{G(i)}{d \times S(i)} \times i \right) \tag{11}$$

.

5.3 Complexity Analysis

The values of $Pr()$ and $Waiting(m)$ have to be calculated only once because these do not depend on the values of $C_a()$. The number of times $Pr()$ has to be computed depends on the number of unique possible values of $Pr(m, n_1, ..n_e, r_1, ..r_e)$. m ranges from 0 to m_{max}. When m_{max} is large, n_1 to n_e depend on B and not on m_{max} so they are of order (B). r_1 to r_e are of order (m_{max}). By multiplying all of them, the number of possible values of $Pr()$ for m_{max} connected clients is O$({m_{max}}^e \times B^e)$. Since m can range from 0 to m_{max}, this complexity has to be multiplied by m_{max} to get the number of possible values of $Pr()$. For a given value of m, the value of $Pr()$ is computed by using equations 4 and 5. Their complexity is the same as the number of values computed. The time complexity is therefore given by O($m_{max}{}^{e+1}$×B^e). For example, if e is 2, m_{max} is 100 and B is 20, the time taken is O(100^{2+1}×20^2) (O(4×10^8)).

As can be seen from equation 4, the values of $Pr(m, n_1, ..., n_e, r_1, ..r_e)$ are in terms of $Pr(m, 0, 0, ..., 0)$. For finding $Pr(m, 0, 0, ..., 0)$, equation 5 is solved which has constant space requirement. Once $Pr(m, 0, 0, .., 0)$ is computed, every other value can be computed by using equation 4. Therefore, the space requirement to compute all values of $Pr()$ is a constant.

Once $Pr()$ is found, $Waiting()$ is computed and its value has to be stored. Therefore, the space complexity depends on the value of $Waiting()$. The space complexity is O(m_{max}).

The time taken to find income and $Pri(m)$ from equations 9 and 10 is O(m_{max}).

The next step is to find the number of possible values of $C_a()$ which has to be multiplied by m_{max} to get the final complexity. As already mentioned, a service provider has to consider all possible values of $C_a()$ and to choose that $C_a()$ which maximizes the expected income. A service provider charges low price when few clients are connected. As more clients connect, it increases the price. If there are two prices, the number of possible decisions which can be taken are three (the price to charge or to reject an arriving client). The number of possible decisions is therefore $(T + 1)$. We can define an array *decision*(i), which stores the minimum value of m at which ith or higher decision is taken. Lower decision means changing to a lower price and higher decision means changing to a higher price or rejecting an arriving client. If i is less than the number of prices, *decision*(i) represents the minimum value of m at which the ith price is charged or higher decision is taken. If i is the number of prices, *decision*(i) means the value of m at which an arriving client should be rejected. *decision*(0) will always be zero because when m is zero, the lowest price will be charged. We have to find the number of possible values of *decision*(). If T is the number of prices, *decision*() will have T+1 values. But *decision*(0) will always be zero. So the number of possible values of *decision*() to be computed which maximizes income are T. It can be easily seen that the complexity to find all possible valid values of *decision*() will be of order O($m_{max}{}^T$). This is multiplied by the time taken to find income which is O(m_{max}) to get O($m_{max}{}^{T+1}$). This complexity of O($m_{max}{}^{T+1}$) is under the assumption that the value of $Pr()$, $Waiting()$, E_d are already known. The overall time complexity is the maximum of the two and it is O($m_{max}{}^{e+1}$×B^e+$m_{max}{}^{T+1}$). If the solution is recomputed after modifying only the mean arrival rate $\lambda()$, the values of $Pr()$, $Waiting()$, E_d need not be recomputed and therefore the time taken will be O($m_{max}{}^{T+1}$). When the solution is implemented, a service provider observes the mean arrival rate of clients and based on this it calculates the output. If the mean arrival rate of client changes, it may recalculate the solution.

It may be possible to reduce the time taken to find the value of $C_a()$ if instead of considering all possible values of *decision*(), binary search technique is used. Because of lack of proof that this will work in all cases, we do not consider this method.

6. A SIMPLE HEURISTIC

In this section, we present a simple heuristic which fixes the price based on an estimate of how loaded the system is. In this method, a service provider estimates the bandwidth requirement by connected clients. If it is greater than or equal to the actual bandwidth, a newly arriving client is rejected. Otherwise, an arriving client is charged an appropriate price and is allowed to connect. The appropriate price is given below. An arriving client is charged price *price*(i) if the following two conditions are satisfied.

$$i \leq \left(\frac{\text{<estimated bandwidth requirement>}}{\text{<total bandwidth>}} \times T \right)$$

and

$$i + 1 > \left(\frac{\text{<estimated bandwidth requirement>}}{\text{<total bandwidth>}} \times T \right)$$

where T is the number of prices. The value of total bandwidth and number of prices is known. The only thing to be found is the bandwidth required by connected clients. The method to find the estimated bandwidth requirement by connected clients is as follows. Let there be m connected clients. With burst times much less than inter-arrival times of new clients, this can be considered a finite population system in steady state. A connected client sometimes remains idle and sometimes consumes bandwidth. Let x be the estimated number of idle clients. These clients open sessions needing i units of bandwidth at the rate $S(0) \times x \times (1-d) \times G(i)$. The estimated number of clients consuming i units of bandwidth is n_i. The clients who are in session with i units of bandwidth complete their sessions at the rate of $S(i) \times n_i$. Both the rates are equal at steady state and so $S(0) \times x \times (1-d) \times G(i)$= $S(i) \times n_i$

Therefore,

$$\frac{x}{n_i} = \frac{S(i)}{G(i) \times (1-d) \times S(0)}$$

Rearranging we get,

$$\frac{n_i}{x} = \frac{G(i) \times (1-d) \times S(0)}{S(i)} \qquad (12)$$

Therefore,

$$n_i = x \times \frac{G(i) \times (1-d) \times S(0)}{S(i)} \qquad (13)$$

The total number of clients connected is m. As a simplification for estimation purposes, we assume that no client waits for bandwidth. Therefore, m is the sum of the estimated number of idle clients and the sum of the estimated number of clients consuming bandwidth:

$$m = x + \sum_{i=1}^{e} n_i$$

With this the following terms are obtained.

$$m = x + x \times \sum_{i=1}^{e} \frac{G(i) \times (1-d) \times S(0)}{S(i)}$$

The ratio of the number of connected clients and the estimated number of idle clients is obtained by the following steps.

$$\frac{m}{x} = 1 + \sum_{i=1}^{e} \frac{G(i) \times (1-d) \times S(0)}{S(i)}$$

$$\frac{x}{m} = \frac{1}{1 + \sum_{i=1}^{e} \frac{G(i)\times(1-d)\times S(0)}{S(i)}} \tag{14}$$

Therefore,

$$\frac{n_i}{m} = \frac{n_i}{x} \times \frac{x}{m}$$

After substituting the value of $\frac{n_i}{x}$ from equation 12 and the value of $\frac{x}{m}$ from equation 14, we get

$$\frac{n_i}{m} = \frac{G(i) \times (1-d) \times S(0)}{S(i)} \times \frac{1}{1 + \sum_{i=1}^{e} \frac{G(i)\times(1-d)\times S(0)}{S(i)}}$$

$$n_i = m \times \frac{G(i) \times (1-d) \times S(0)}{S(i)} \times \frac{1}{1 + \sum_{i=1}^{e} \frac{G(i)\times(1-d)\times S(0)}{S(i)}}$$

Estimated bandwidth consumed is

$$\sum_{i=1}^{e} i \times n_i$$

After substituting the value of n_i, the estimated bandwidth consumed is found to be

$$\sum_{i=1}^{e} i \times m \times \frac{G(i) \times (1-d) \times S(0)}{S(i)} \times \frac{1}{1 + \sum_{i=1}^{e} \frac{G(i)\times(1-d)\times S(0)}{S(i)}}$$

Table 3: Simulation Details

Variable	Value
d	0.4
B	60
m_{max}	200
e	2
T	3
$price(0)$	0.1 per unit data
$price(1)$	0.12 per unit data
$price(2)$	0.15 per unit data
$P(0)$	0.4 per second
$P(1)$	5
Mean idle time ($\frac{1}{S(0)}$)	20
Mean session duration ($\frac{1}{S(1)}, \frac{1}{S(2)}$)	$\frac{\text{Mean idle time}}{5}$
$G(1)$	0.3
$G(2)$	0.7

7. SIMULATION

The performance of our solution is compared with the simple heuristic through simulation to find out whether our more complex algorithm will provide benefits or the simple heuristic is good enough. In our model, we have assumed multihomed clients and multiple service providers. At steady state, a service provider knows the mean arrival rate of clients when it charges a particular price. A service provider gets this information by observing the past arrival rates. When a service provider charges prices $price(i)$, the mean arrival rate of clients is $\lambda(i)$. With this service providers compute the solution matrix. The solution matrix of the service provider is static and the decision of service providers is according to the solution matrix.

One method of simulating arrivals is to generate arrivals according to the price charged by the service provider. When a service provider charges $price(i)$, the mean arrival rate of clients is $\lambda(i)$. The value of $\lambda(i)$ is already known to the service providers. This type of simulations are given in section 7.1.

Another method of simulating arrivals is to consider multiple service providers with each client able to connect to two service providers. The mean arrival rate of clients with which they request for connection to a pair of service providers is fixed. The rate at which clients connect to one of the two service providers when some price is charged is unknown. A service provider observes the rate at which clients connect to it and updates its decision matrix periodically. The second set of simulations are given in section 7.2. In this simulation, service providers have incomplete information and the mean arrival rate is observed for some time.

The output of the simulations consist of three components. These are written as income, delay and rejection. The income is the total income in the form of money earned by a given service provider. The delay is the total amount of time, the connected clients have to wait. If delay is multiplied by the $P(0)$, it will give the penalty paid to waiting clients. Rejection is the total number connection requests rejected. If it is multiplied by $P(1)$, it will give the total penalty paid to rejected clients.

Table 4: Simulation result for price based arrival rates

$\lambda(0)$	$\lambda(1)$	$\lambda(2)$	Analysed Solution			Simple Heuristic		
			Income	Delay	Rejects	Income	Delay	Rejects
6	4	2	20,516	9,907	7	18,615	11,905	67
8	4	2	20,512	10,200	11	18,851	11,659	75
8	6	2	21,807	10,317	11	17,032	15,611	143
10	6	2	22,072	9,169	10	17,005	15,564	165
8	6	4	25,597	8,096	53	18,001	12,568	162
10	6	4	25,332	9,007	65	17,912	13,687	147
10	8	4	24,497	11,969	60	16,969	15,334	195
12	8	4	24,923	10,462	58	17,563	13,622	199
Total			185,256	79,127	275	141,948	109,950	1153
Percentage of deviation from values in the simple heuristic solution			+31%	-28%	-76%	-	-	-

7.1 Simulation with price based arrival rates

In these simulations, the mean arrival rate of clients is $\lambda(i)$ when a service provider charges price $price(i)$. The simulation runs for one hour. We did multiple simulations to compare the performance of the solutions. We considered multiple combinations of mean arrival rates and compared the result.

In these simulations, there are two identical service providers and they are using the analysed solution, and the simple heuristic respectively to find the appropriate price for an arriving client and the decision to accept or reject the connection request of an arriving client.

Simulation specifications are given in Table 3. There are three prices and therefore there are three mean arrival rates that are based on the price charged. The value of $\lambda(2)$ takes two values 2 and 4. For each value of $\lambda(2)$, $\lambda(1)$ takes two values $\lambda(2) + 2$ and $\lambda(2) + 4$. For each combination of $\lambda(2)$ and $\lambda(1)$, $\lambda(0)$ takes two values $\lambda(1)+2$ and $\lambda(1)+4$. The total number of combinations are 2×2×2=8. Therefore, there are 8 simulations.

The simulation results are given in Table 4. There are eight simulations and the value of mean arrival rates for each simulation is given. For finding the average performance of the two solutions, the incomes, delays and rejections are added for each of the simulations. The performance difference of the analysed solution as compared to the simple heuristic solution are written as percentage of performance of the simple heuristic solution. Simulation results show that income produced by our solution is 31 % more as compared to the simple heuristic solution. The quality of service is also improved in the analysed solution, with the mean delay reducing by approximately 28 % and number of rejections by approximately 76 %. This shows that the analysed solution significantly improves income and quality of service.

Table 5: Simulation Details

Variable	Value
d	0.4
B	40 to 70
m_{max}	200
e	2
T	3
$price(0)$	0.1 and 0.2 per unit data
$price(1)$	prices (0)+0.1 and price(0)+0.2 per unit data
$price(2)$	price(1)+0.1 and price(1)+0.2 per unit data
$P(0)$	0.4 per second
$P(1)$	5
Mean idle time ($\frac{1}{S(0)}$)	20
Mean session duration ($\frac{1}{S(1)}$, $\frac{1}{S(2)}$)	$\frac{\text{Mean idle time}}{5}$
$G(1)$	0.3
$G(2)$	0.7

7.2 Simulation with observed arrival rates

When simulation starts, the service providers assume some randomly estimated value of $\lambda()$ based on the mean arrival rate of clients. It is because the rate at which clients will connect to a service provider when some price is charged is not known. However the rate at which clients request for connection is known. Every 5 minutes, one service provider recomputes its optimal decision based on the observed mean arrival rate of clients. A service provider observes the mean arrival rate of clients and recomputes its optimal decisions once in nearly 5×(number of service providers) minutes. We also consider the simple heuristic. If a service provider uses the simple heuristic, it does not observe the mean arrival rate. When the turn of the service provider who uses the simple heuristic comes, no update is done. The income displayed is income for one hour duration. The simulation runs the system for three hours and the income displayed is of the third hour. The first two hours of simulation is removed. Among these simulations, the simulations in which any service provider earns negative income are removed.

We did multiple simulations to compare the performance of the solutions. We considered multiple combinations of prices and bandwidth and compared the result. There are six identical service providers and three of them are using the analysed solution, and three of them are using the simple

Table 6: Simulation result for observed arrival rates

	Analysed Solution	Simple Heuristic
Total Income	6,725,348	5,986,527
Percentage change in total income as compared to total income in the simple heuristic solution	+12%	-
Total delay	703,418	3,008,866
Percentage change in total delay as compared to total delay in the simple heuristic solution	-77%	-
Total rejections	198	67,054
Percentage change in total rejection as compared to total rejection in the simple heuristic solution	-100%	-

heuristic to find the appropriate price for an arriving client and the decision to accept or reject the connection request of an arriving client. The mean arrival rate of clients who are common to each pair of service providers is 1 per second. The overall mean arrival rate of clients for a service provider is 1×5 because a service provider has clients common with five other service providers. This is just the rate at which clients will request for internet access and it is not the rate at which clients will connect to a service provider.

Simulation specifications are given in Table 5. The value of bandwidth ranges from 40 to 70 in multiples of 5 (40, 45, 50,...70). For each of the bandwidth, there are eight combinations of price(0), price(1) and price(2) as in the earlier simulation. The total number of combinations are 7×8=56. Therefore, there are 56 simulations. The simulations in which any service provider earns negative income are removed.

The simulation results are given in Table 6. The incomes, delays and rejections are added for each of the simulations and these are displayed in Table 6. The results show that the increase in income by the analysed solution is 12% as compared to the simple heuristic solution. This solution reduces delay by approximately 77% and the number of rejections by approximately 100%. The analysed solution therefore significantly improves the quality of service.

8. CONCLUSION

We have considered the problem of providing QoS guarantees to clients in which penalty is paid to clients whenever promised service is not provided. This model puts pressure on service providers to give good service to its clients. Simulation result shows that service providers who use our solution give better service to its clients as compared to a simple heuristic based solution. Simulation results also show that our solution significantly improves quality of service in terms of number of connection request rejected by service providers and the average time for which a connected client has to wait for receiving requested bandwidth. Our solution also increases the income of service providers.

The simulation results show that a solution based on our model provides significant improvements as compared to a simple but reasonable ad hoc solution. So our work indicates that a move away from the current simple pricing schemes is required to provide a level of quality in service while not compromising on income to providers. To the best of our knowledge, this is the first analytical approach towards tackling this problem.

9. REFERENCES

[1] S. Ha, C. Joe-Wong, S. Sen, and M. Chiang. Pricing by timing: Innovating broadband data plans. In *Proceedings of SPIE - The International Society for Optical Engineering*, volume 8282, 2012.

[2] P. Hande, M. Chiang, R. Calderbank, and J. Zhang. Pricing under constraints in access networks: Revenue maximization and congestion management. In *Proceedings - IEEE INFOCOM*, 2010.

[3] Ventura N. Golovins E. Ozianyi, V.G. A novel pricing approach to support qos in 3g networks. *Computer Networks*, 52(7):1433–1450, 2008.

[4] Tsitsiklis J.N. Paschalidis, I.Ch. Congestion-dependent pricing of network services. *IEEE/ACM Transactions on Networking*, 8(2):171–184, 2000.

[5] R. S. Randhawa R. Garg. A sla framework for qos provisioning and dynamic capacity allocation. In *10th International Workshop on Quality of Service (IWQoS 2002)*, pages 129–137, May 2002.

[6] S. Sen, C. Joe-Wong, S. Ha, and M. Chiang. A survey of smart data pricing: Past proposals, current plans, and future trends. *ACM Computing Surveys*, 46(2), 2013.

[7] S. Shakkottai, R. Srikant, A. Ozdaglar, and D. Acemoglu. The price of simplicity. *IEEE Journal on Selected Areas in Communications*, 26(7):1269–1276, 2008.

[8] János Sztrik. Basic queueing theory. *University of Debrecen: Faculty of Informatics*, 2011.

[9] R. Tripathi and G. Barua. Dynamic internet pricing and bandwidth guarantees with nash equilibrium. In *16th Asia-Pacific Network Operations and Management Symposium*, 2014.

[10] S. Yaipairoj and F. C. Harmantzis. Dynamic pricing with "alternatives" for mobile networks. *2004 IEEE Wireless Communications and Networking Conference, WCNC 2004*, 2:671–676, 2004.

[11] Y. Zhang, C. Lee, D. Niyato, and P. Wang. Auction approaches for resource allocation in wireless systems: A survey. *IEEE Communications Surveys and Tutorials*, 15(3):1020–1041, 2013.