Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Computer Communications 27 (2004) 374–385

computer communications

www.elsevier.com/locate/comcom

A dynamic pricing scheme for e-content at multiple levels-of-service

Srinivasan Jagannathan*, Kevin C. Almeroth

Department of Computer Science, University of California, Santa Barbara, CA 93106-5110, USA

Received 8 August 2003; accepted 8 August 2003

Abstract

Businesses selling multimedia rich software or *e-content* are growing in the Internet. The e-content can be downloaded or streamed immediately after an on-line transaction. Since Internet connection speeds are variable, ranging from dial-up access speeds to broadband speeds, a content provider may provide content at different speeds or levels-of-service. Providers offering content at different service levels face two major challenges: (1) revenue maximization, and (2) resource provisioning. In this article, we discuss how these challenges are inter-related, and develop a formal model for pricing and resource provisioning in content delivery systems. We use simulations to study price and resource utilization dynamics in systems implementing our model. We present simulation results in a variety of scenarios that illustrate the scalability and robustness of our model.

Keywords: Levels-of-service; HYBRID; e-content

1. Introduction

Available bandwidth and usage have increased in the Internet. Use of the Internet to purchase goods and services is also increasing. At the same time, the multimedia capabilities of computers are improving while remaining affordable. Together, these trends have spawned services offering video-on-demand (VoD), downloadable CDs, etc. In these services, customers should have adequate bandwidth to receive content and multimedia-capable computers to view it. While such services are growing, there exists great heterogeneity in the resources of users, both in terms of connection speeds and also in the multimedia capabilities of their computers. For instance, many users connect at dial-up speeds, while many others at broadband speeds. Similarly, some users may be using multimedia capable, but power-deficient mobile processors, while others are using the latest desktop based Gigahertz-machines. This heterogeneity appears to be an inherent feature of the future Internet. To accommodate such heterogeneity, a content provider may serve content at different quality levels. For instance, many web sites offer streaming content at different quality levels and in different formats.

Offering content at multiple levels-of-service (LoS) has many advantages. It offers greater flexibility to users, and multiple revenue streams to content provider. But there are two main challenges in a system with multiple LoS: (1) how do we choose revenue-maximizing prices for content, and (2) how do we allocate resources among the different LoS so that we realize the maximum expectation of revenue. Choosing a revenue-maximizing price depends on a good understanding of customer behavior. Customer behavior can vary widely in an Internet-based market. For instance, geographically dispersed users can access the same web site at different times during the same day. Such users can differ significantly in their purchasing behavior. We thus need a pricing model that is robust to dynamic customer behavior. Similarly, we need a scalable and robust mechanism to manage available resources. An important problem in managing available resources is to be able to quantify it. For instance, consider a system where resources are quantified in terms of channels. Consider a system with 100 channels, offering two types of content: A and B. Suppose that for a stream serving content A, one channel is allocated, and for a stream serving content B, two channels are allocated. Then the system can accommodate 100 requests for content A, or 50 requests for content B. The actual number of requests that the system serves will vary with the relative fraction of content A versus content B

* Corresponding author.
E-mail address: jsrini@cs.ucsb.edu (S. Jagannathan).

0140-3664/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2003.08.010

requests. Moreover, when there are more requests than the system can serve, it is difficult to decide which requests to satisfy. For instance, if it is known that customers are willing to pay at least $5 for content *A* and $7 for content *B*, accepting *A* requests will increase the revenue when resources are constrained. However, since how much customers are willing to pay is not known, deciding which requests to serve is difficult.

In this article, we develop a model to address these two problems. In particular, our focus is on the server and not the network. Thus, when we talk about allocating bandwidth for a stream, we refer to bandwidth reservation at the outgoing link of the server and not along the network-path to the user. In our earlier work [1], we compared a number of simple pricing schemes using simulations. These pricing schemes could be classified as being *static* or *dynamic*. In a static pricing scheme, the price of the content does not change frequently. In a dynamic pricing scheme, the price may vary on much smaller time scales based on factors like current server load, request arrival rate, etc. Based on our simulations, we believe that there exist fixed prices that generate very high revenues but finding these prices is non-trivial. We formulated a dynamic pricing scheme called HYBRID, which not only generated consistently high revenues across a range of simulation scenarios and customer populations, but also reduced the number of requests rejected due to lack of server resources. In this article, we primarily focus on extending the HYBRID pricing scheme to systems with multiple LoS (multi-LoS systems). We validate our work through simulations.

We now briefly survey related work. Pricing research can be divided into two parts: (1) connectivity pricing, and (2) content pricing. Connectivity pricing deals with charging, accounting, and pricing for usage of network resources. There is copious literature on connectivity pricing. Stiller et al. [2], Falkner et al. [3], and DaSilva [4] present excellent overviews of different concepts in connectivity pricing, and also evaluate many pricing approaches along different dimensions. Our work on content pricing differs from connectivity pricing in that we seek to choose a price for content as opposed to setting a price for the transport medium. Our work does borrow some ideas from connectivity pricing, especially the idea of setting a congestion price when the system is over-loaded.

There has been very little work on pricing on-demand delivery of content. Content pricing research has mainly focused on issues like price-wars in multi-agent markets [5,6], niche-discovery in computational economies [7], and pricing of information bundles [8,9]. None of these works consider delivery constraints of content providers. The presence of delivery constraints, coupled with dynamic request patterns makes it impossible to apply solutions presented in these works to on-demand content pricing. There has also been some work on pricing and scheduling in Video-on-Demand (VoD) systems from the perspective of revenue maximization. Basu and Little [10], have formulated models for VoD and pricing issues related to them. Wolf et al. [11] study how to maximize profits when broadcasting digital goods. When resources are constrained, they schedule the delivery at a later time, and pay a penalty for late delivery by charging a lower price. They do not discuss how the prices and penalties are chosen. Chan and Tobagi [12] design profit maximizing scheduling schemes for batched delivery of VoD, when the fixed price for the content is known. Their work does not consider multiple LoS. Krishnamurthy [13] investigates resource allocation for VoD based on dynamic pricing. The resources allocated to a stream are dynamically modified over time based on the current load. The price varies as a function of the allocated resources. In this model, customers know the exact price only after the streaming is complete. This is significantly different from our approach where the price is known before the purchase, and resources remain allocated until the end of streaming.

This work builds on our earlier research for pricing on-demand delivery of e-content when there is a single LoS [14–16]. In our earlier work, we have presented an overview of the on-demand content pricing problem, and studied the factors that influence revenue [14]. We have developed deterministic [1], as well as probabilistic customer behavior models [15,16] in on-demand content delivery markets with a single product [16] as well as with multiple products, all sharing the same server resources [15]. We have also studied revenue maximization in the presence of smart server management schemes like batching [1]. In all our earlier work, we assumed that each request consumes the same amount of server resources. In this work, we study how the revenue maximization problem is affected when different requests consume different amounts of server resources.

The remainder of the paper is organized as follows. Section 2 describes a formulation for revenue earned in multi-LoS systems. Section 3 describes our HYBRID pricing scheme and two other dynamic pricing schemes adapted from the work by Sairamesh and Kephart [6]. Section 4 discusses the simulation framework and the experiments we perform. Results are presented in Section 5. We conclude the paper in Section 6.

2. Revenue model and resource provisioning

We consider a system where requests are satisfied if resources are available and the customer agrees to pay the quoted price. We assume that all server resources can be quantified and mapped to a real number. One approach to doing this is to consider the bottleneck resource at the server as the indicator of system resources. For example, if bandwidth is the bottleneck, then the total available bandwidth is modelled as the system capacity. For the purposes of this paper, we shall assume that available connection bandwidth of the content provider is the measure

of system resources. When a request is served, some of the connection bandwidth is allocated to that request.[1] Requests are processed on a First-Come-First-Served basis. If there is insufficient bandwidth available when a request arrives, then the request is rejected. In our model, we assume that once the content provider makes the initial infra-structural investment, there are either negligible or fixed costs in maintaining the resources (caches, servers, bandwidth, etc.), i.e. there are no additional costs based on number of requests served. This is a reasonable assumption because servers incur fixed costs and bandwidth can be bought at a flat monthly rate. If maintenance costs are negligible or fixed, profit maximization is equivalent to revenue maximization. We also assume that the market is monopolistic, i.e. there is no other entity selling the same content. This is a realistic assumption in many scenarios where the content owner personally sells the content or has licensed it to a single distributor.

Table 1 presents the symbols we have used in our analysis. Consider an arbitrary customer who wants to purchase content $p_{i,j}$. We denote his/her decision to purchase the service by the random variable $Y_{i,j}$ which can take two values, 1 for accept and 0 for reject. Let $E[Y_{i,j}|\psi_{i,j}]$ denote the expectation of the decision to purchase content $p_{i,j}$ when the price is $\psi_{i,j}$. The expectation of revenue per unit time is given by:

$$\Re = \sum_{i=1}^{m}\sum_{j=1}^{L} \lambda_{i,j}\psi_{i,j}E[Y_{i,j}|\psi_{i,j}] \tag{1}$$

The revenue function described above does not consider resource constraints. To model resource constraints, we use the notion of system utilization. System utilization, ρ, is defined as the ratio of the number of requests entering the system per unit time to the maximum number of serviced requests exiting the system per unit time. In a stable system, this ratio must be less than or equal to 1. If there are more requests than the system can serve, the predicted system utilization exceeds 1. Therefore, we impose an additional constraint that the predicted system utilization should be less than or equal to 1. System utilization is easily defined when there is a single LoS. If l is the resources consumed by a request at this LoS, the system utilization can be computed as shown in Eq. (2):

$$\rho = \frac{dl}{\mathfrak{B}}\sum_{i=1}^{m} \lambda_i E[Y_i|\psi_i] \tag{2}$$

However, with multiple LoS, and requests at each LoS consuming different amount of resources, it is not possible to quantify the maximum number of serviced requests exiting the system. We therefore take a different approach. Suppose that the system resources are partitioned into $\langle b_1, b_2, \ldots, b_L\rangle$, where b_j is the resource provisioned for level j. Then, we can impose the system utilization constraint independently for each LoS. We solve an independent constrained maximization problem for each LoS. The total revenue earned critically depends on how the resources are partitioned for each level. Notice that resources consumed by requests for level j will be less than or equal to $\sum_{i=1}^{m} l_j\lambda_{i,j}$. Based on this, we provision resources as follows:

$$b_j = \frac{\sum_{i=1}^{m} l_j\lambda_{i,j}}{\sum_{j=1}^{L}\sum_{i=1}^{m} l_j\lambda_{i,j}} \tag{3}$$

The revenue maximization problem is then given by:

- Maximize: $\sum_{j=1}^{L} \Re_j$ where $\Re_j = \sum_{i=1}^{m} \lambda_{i,j}\psi_{i,j}E[Y_{i,j}|\psi_{i,j}]$
- Subject to:

$$\psi_{i,j} \geq 0, 1 \leq i \leq m, 1 \leq j \leq L$$

$$\rho_j \leq 1, 1 \leq j \leq L$$

where

$$\rho_j = \frac{dl_j}{b_j}\sum_{i=1}^{m} \lambda_{i,j}E[Y_{i,j}|\psi_{i,j}]$$

As can be observed, the revenue model relies on knowledge of the request arrival rate and the expectation of the decision to purchase, given the price. The request arrival rate can be monitored. However, the expectation of the decision to purchase is not known. In Section 3 we outline the HYBRID scheme which estimates the expectation of the decision to purchase.

Table 1
Symbols used

Notation	Description
m	Number of products
L	Number of levels of service
$\mathfrak{B}$	Total system resources
b_j	Resources provisioned for jth LoS
l_j	Resources for serving a request at jth LoS
$p_{i,j}$	ith product at jth LoS
$Y_{i,j}$	Decision to purchase $p_{i,j}$ (0 or 1)
$\psi_{i,j}$	Price of $p_{i,j}$
$\lambda_{i,j}$	Request arrival rate for $p_{i,j}$
$\Re$	Total revenue per unit time
d	Mean service time
ρ	System utilization

3. Dynamic pricing algorithms

In this section, we briefly describe the HYBRID pricing scheme. The HYBRID algorithm is based on the premise that customers are rational human beings. We are interested in the fraction of requests that will result in successful transactions. For a rational customer population, it can be argued that this fraction is a non-increasing function of the quoted price. For a price x, let $f(x)$ denote the fraction of

[1] This does not imply that network resources are reserved.

customers who will accept the price. Let x_{low} be a price below which $f(x)$ is exceptionally high, say more than t_h and let x_{high} be a price above which $f(x)$ is exceptionally low, say below t_l. Then $f(x)$ can be approximated in the domain $[x_{low}, x_{high}]$ using some non-increasing function. We propose a family of decreasing functions which depend on a parameter δ described as follows:

$$f(x) = \begin{cases} t_h, & 0 \le x < x_{low} \\ (t_h - t_l)\left[1 - \left(\dfrac{x - x_{low}}{x_{high} - x_{low}}\right)^{\delta}\right] + t_l, & x_{low} \le x \le x_{high} \\ t_l, & x > x_{high} \end{cases} \quad (4)$$

Fig. 1 illustrates the family of non-increasing functions. By experimenting with different prices to observe the fraction of customers who accept the price, and using statistical methods like least squared errors, one can estimate the parameter δ, and the threshold prices x_{low} and x_{high}. Notice that $f(x)$ is also the expectation of the decision to purchase, given price x. Once all the parameters are known, the content provider can predict the customer behavior and thereby choose a price[2] using the optimization problem described in Section 2. In HYBRID, the customer reaction is continuously monitored, and the price is varied at regular intervals.[3] The details of this algorithm are presented in our earlier work [1].

After performing simulations with the scheme described above, we observed that while the revenue earned was high, the number of requests rejected due to lack of resources was also high. This was mainly because when the algorithm experimented with low prices, more customers accepted the service than could be accommodated by the server. We therefore modified the algorithm as follows. Whenever the server load increased beyond a certain threshold, an exponentially increasing price was quoted. Suppose that x is the fraction of available resources that have been allocated to satisfy requests. Let L and H be the lowest price and highest price that the content provider decides to quote to customers. Then, if x is greater than a threshold, the price quoted to a customer, irrespective of the content requested, is given by: $(L-1)+(H-L+1)^x$. This modified algorithm, generated consistently high revenues while at the same time minimizing the number of requests rejected due to lack of resources.

3.1. Other dynamic pricing algorithms

We present two dynamic pricing algorithms adapted from the work of Sairamesh and Kephart [6]. These algorithms were observed to converge to the game-theoretic optimal price in a competitive market in the simulations performed by Sairamesh and Kephart. We chose these algorithms for the purposes of evaluation and comparison with our algorithm. Though many other algorithms were presented in their work, the market assumptions for the other algorithms did not match the market scenario of our work. For instance, we assume a monopolistic market and that there are constraints on the distribution resources. Their work was for a competitive market with no constraints on the delivery mechanism.

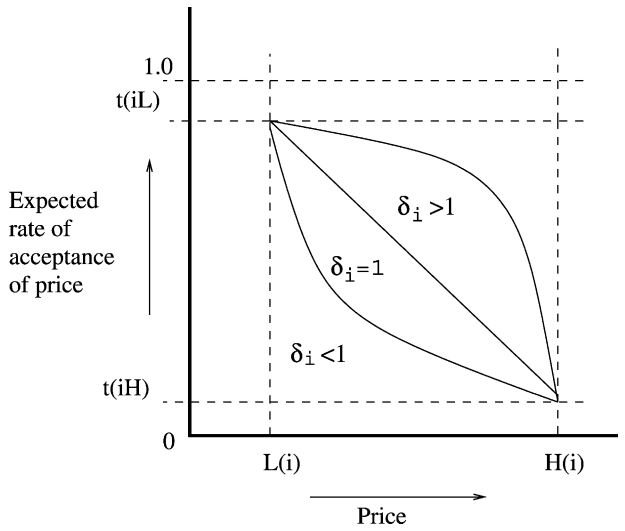

Fig. 1. Customer model.

Let L and H be the lowest and highest prices, respectively, that the content provider decides to quote. In the Trial-and-Error-Pricing (TEP) algorithm, an initial price is chosen at random in the range $[L, H]$. At regular intervals, with a small probability (called small jump probability) a random price increment is chosen from a standard normal distribution having very small σ. After the price change, revenue earned in the next interval is monitored. If revenue earned per request is lower than before, the old price is restored. In addition, with a very small probability (called big jump probability), a new price is chosen at random. The big jump probability is much smaller than the small jump probability.

The Derivative-Following-Pricing (DFP) algorithm is similar to the TEP algorithm. An initial price in the range $[L, H]$ is chosen at random. At regular intervals, the price is varied by a random step size. If in the next interval, revenue per customer increases, then the next increment is chosen in the same direction, i.e. price is increased. If, however, the revenue decreases, then the direction of increment is reversed, i.e. the price is decreased. At all times, the price is kept in the range $[L, H]$.

4. Simulations

We performed simulations to evaluate our pricing algorithm. Using simulations for evaluation has two advantages. First, we can test the robustness and scalability

[2] The price so obtained may not be the global optimum.

[3] Temporal price variations are an inherent feature in many commodity markets.

of the system by generating artificially high load profiles. Second, we can test performance under different customer behavior profiles. This is especially useful because it may be impossible to create scenarios that stress-test the algorithm when real customers are involved.

Our simulation implementation was designed to model a content delivery system. All our simulations are averaged over five runs with different seed values for the random number generator. We describe the components of our simulation below.

4.1. System description

We performed simulations on a hypothetical content delivery system with a T3 (45 Mbp) outgoing link. We assumed that there are enough servers to accommodate all the incoming requests. Therefore, outgoing bandwidth is the bottleneck resource. In our system, customers could choose from one of two LoSs: 64 or 256 kbp. We chose request service times from a uniform distribution between 90 and 110 min. This closely models the typical length of movies in a VoD system.

4.2. Customer choice of products

For the simulation results presented in this article, we assume that there are only two products: A and B. A represents a stream requiring 64 kbp CBR, and B represents a stream requiring 256 kbp CBR. We chose only two products so that we can illustrate the dynamics of price, customer behavior, and system load. Our model is, however, more general, and can be applied to systems with more than two products or LoS. We have presented simulation results for systems with 100 products in some of our earlier work [17].

4.3. Customer behavior

In our simulations, a customer's valuation for a product is a random variable drawn from a probability distribution. If the price quoted by the content provider is less than this valuation, the customer accepts the content, otherwise the customer rejects the content. We assume that the content provider has no knowledge about the probability distribution. The content provider can only observe how many customers accept or reject the current quoted price.

Since humans typically think in terms of discrete values, we chose two discrete probability distributions for modelling customer valuations: Uniform, and Zipf. Furthermore, the valuations were always chosen to be an integer dollar value. This is because it is intuitive for humans to think of a round figure instead of a decimal number to represent their valuation of a product. For the sake of completeness, we also chose one continuous distribution—Normal. We ignored negative values drawn from this distribution. Valuations drawn from this distribution could be decimal numbers. In real life, customer valuations may not conform to any of these distributions. But in the absence of real life data, our objective was to test the robustness of the pricing algorithms over a range of 'feasible' customer behavior patterns.

In all our simulations, our unit of currency is dimes (10 dimes = \$1). We performed simulations with numerous customer valuations. We present results for valuations corresponding to prices charged in movie theaters.[4] In case of the Uniform and Zipf distributions, customer valuations for product A are drawn from the set {20, 30, 40, 50, 60}, while customer valuations for product B are drawn from the set {50, 60, 70, 80, 90}. For Normal distribution, the $\langle \mu, \sigma \rangle$ of the distributions are: $\langle 40, 5 \rangle$ for product A, and $\langle 70, 5 \rangle$ for product B. We chose a higher mean valuation for product B because it is offered at a higher LoS.

4.4. Pricing policy

We assume that the content-provider will charge at least \$1 and not more than \$10 for serving the content. We simulated all three pricing algorithms described in Section 3. For the TEP algorithm, we set small jump probability to be 0.05 and big jump probability to be 0.001 as mentioned by Sairamesh and Kephart [6]. For the DFP algorithm, price increments were chosen from a uniform distribution in the range $[0, n]$, where n is a random number in the range $[L, H]$ chosen at the beginning of the simulation. In case of the HYBRID algorithm, we chose a server load threshold of 0.80. When current server load exceeded this threshold, exponentially increasing prices were charged.

In addition to these dynamic pricing algorithms, we also simulated a fixed pricing algorithm. Since the content provider will not charge less than \$1 or more than \$10, and does not have any prior knowledge of customer behavior, any price in the range \$1 to \$10 is equally likely. However, the content provider can make two observations about the customer behavior: (1) valuation for product B is likely to be higher than valuation for product A, and (2) valuations are likely to be integer dollars.[5] Based on these observations, we performed separate simulations where the fixed price for A was chosen from {19.99, 29.99, 39.99, 49.99, 59.99, 69.99, 79.99, 89.99} and that of B was chosen from {19.99, 29.99, 39.99, 49.99, 59.99, 69.99, 79.99, 89.99, 99.99}. The prices are slightly less than an integer value because customers will accept the content only if the price is strictly less than their valuation. We ran simulations for each fixed price of A, and each price of B which was greater than or equal to the price of A. Since any of these prices is equally likely, in our results we shall only present the mean performance over these prices.

[4] We have observed theaters charging anywhere in the range of \$2.50 to \$8.50 for movies.

[5] This observation is clearly false in case of the Normal distribution, but is likely to be true in real life.

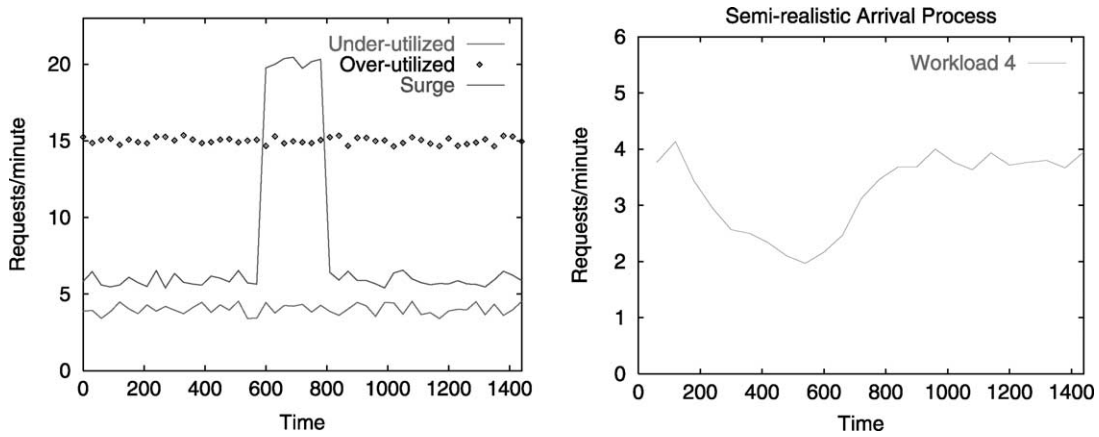

Fig. 2. Request arrival process.

4.5. Request arrival process

In order to test the robustness of the pricing algorithms, we used four different request arrival process, which we call Workloads. These are shown in Fig. 2. Workload 1, 2, and 3 are artificial request arrival processes that we generated to illustrate the behavior of the pricing algorithms under different situations. Workload 4 is a semi-realistic request arrival process that we generated from the request logs of a Content Delivery Network.[6] Workload 1 represents a scenario where there are very few requests for content, and the server has enough resources to accommodate all the requests. Workload 2 represents the scenario where the server is continuously overloaded. But even this scenario is 'static' in the sense that there is no change in the request arrival pattern over time. Workload 3 represents a scenario where there is a sudden synchronizing event, which generates very large number of requests in a short time frame. The heavy load subsides just as suddenly as it starts. Workload 4 represents the number of requests for multimedia content in Quicktime format on April 15, 2001. We chose this day arbitrarily from the log files for the period December 28, 2000 to September 8, 2001. We could not ascertain the actual bit rate of the streams from the available data. However, based on the recorded value for total bits transmitted in each hour, we generated the arrival process assuming that the streams are either 64 or 256 kbp.

For the first three Workloads we assume that requests for *A* and *B* are equally likely and that the relative fraction of 64 versus 256 kbp requests does not vary throughout the day. This allows us to study the impact of the request arrival process on the pricing algorithm in greater detail. In case of Workload 4, our objective is to study the behavior on a 'normal day'. Therefore, we also vary the relative fraction of 64 and 256 kbp requests. The relative fraction of 64 and 256 kbp requests estimated from the log files is shown in Fig. 3. As can be seen, the fraction of requests for the 64 kbp stream though varying throughout the day, is much higher than the fraction of requests for the 256 kbp stream.

[6] The name of the CDN has been withheld on request.

5. Results

Our simulation results can be divided into two parts. In the first part, our objective is to understand the behavior of the HYBRID pricing algorithm. To this end, we present results from a sample simulation run under different customer behavior and system load profiles. In the second part, we compare the overall performance of the HYBRID algorithm with that of other pricing algorithms under different customer behavior and system load profiles.

5.1. Understanding the behavior of the hybrid pricing algorithm

All our simulations are performed over one day of simulated time. For all the simulation results presented in this sub-section customers' valuations were drawn from the Uniform distribution described earlier. In our results, we present price as a number in the range 0–1. We obtained this

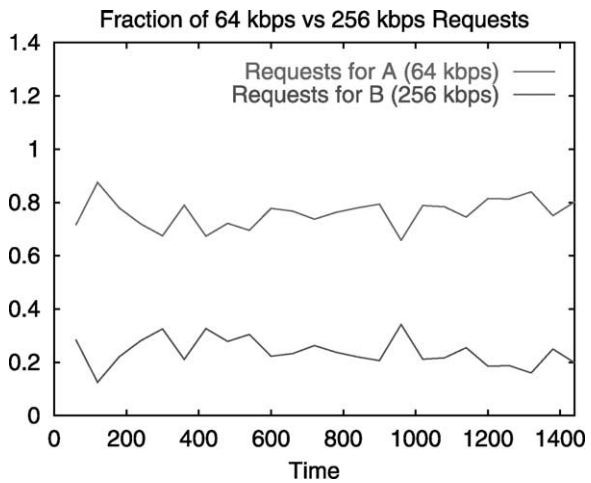

Fig. 3. Relative fraction of 64 and 256 kbp requests.

number by dividing the price by 100 (the price upper bound). We study the behavior in terms of the following metrics

- Evolution of price for each product. We study how price varies throughout the day and how it is correlated to external events.
- Evolution of server load due to each product. In our simulations, we assume that outgoing bandwidth is the bottleneck resource. We define server load due to a product to be the ratio of outgoing bandwidth consumed by active requests for that product to the maximum allocation for that product.
- Evolution of relative fraction of server resources reserved for each product. We reserve server resources according to Eq. (3). We study how the relative fraction of resources varies with time.

In many of our simulation results we shall observe that HYBRID takes around 300–400 min to converge to a stable price. This occurs because HYBRID requires five data points before it estimates customer behavior parameters described in Eq. (4). A data point is the result of observing customer behavior for an interval of time using a test price. For the simulations discussed in this article, we chose an interval size of 45 min. Hence, price fluctuations can be observed for at least the first $45 \times 5 = 280$ min. This time period can be reduced using a smaller time interval or by considering fewer data points for estimating the customer behavior parameters. We now present illustrative simulation results for five different scenarios.

5.1.1. Scenario 1: low request arrivals, sufficient resources

We used Workload 1 to generate this scenario. Requests were generated using a Poisson process with mean 1.4 min^{-1}. Requests for A and B were equally likely. Fig. 4 shows a sample simulation run.

HYBRID first experiments with different prices in order to observe the fraction of customers accepting those prices. This can be seen in the fluctuation in prices of both A and B. In some cases, the price chosen results in no customers accepting the price, or all customers accepting it. We can see this behavior in time interval 0–45 for product A where no customer accepts the price (because the maximum customer valuation is 60), and in time interval 135–180 for product B, where all customers accept the price (because the minimum customer valuation is 50). When this happens, a new price is chosen and the experiment is repeated. Once HYBRID learns the parameters, it predicts the 'optimal' price using the maximization problem defined in Section 2. As can be seen, the price converges for product A, while there is some re-learning in case of product B.

The relative fraction of resources allocated to each product does not vary significantly throughout the day. This is because the request arrivals for A and B are nearly

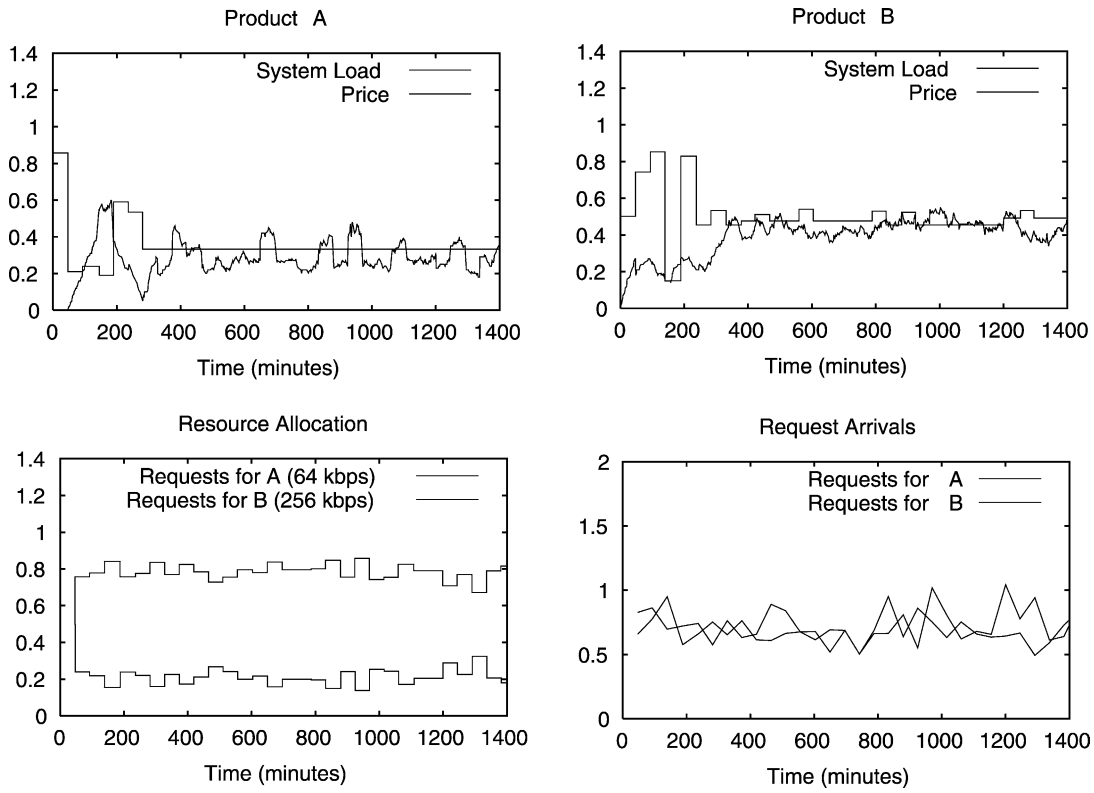

Fig. 4. Low request arrivals, sufficient resources.

the same throughout the day. Also note that the server load is not very high for both the products due to the low rate of request arrivals.

5.1.2. Scenario 2: high request arrivals, insufficient resources

We used Workload 2 to generate this scenario. Requests were generated using a Poisson process with mean 20 min^{-1}. Requests for A and B were equally likely. Fig. 5 shows a sample simulation run.

HYBRID behaves very differently in this scenario. As before, for product A, the algorithm begins with a very high price as a result of which, no customer accepts the price. After the 45 min interval, HYBRID lowers the price. Immediately, because of the high request arrival rate, the server load shoots up and soon surpasses the threshold of 0.8. Whenever the server load exceeds this limit, HYBRID charges an exponentially increasing price computed using the formula $(9 + 91^x)$, where x is the current server load. The high price immediately arrests the increase in server load. The server load remains static around 0.9 because of the requests that are already being serviced. Once these requests are no longer active, the server load drops and consequently the price. This again leads to an increase in server load. This cyclic behavior is observed for the first 350 min. However, at the end of this time-period, HYBRID converges to a price that is just sufficient to maintain a high system utilization but still lower than that imposed by the threshold server load. On the brief occasions that the server load exceeds the threshold, the price increases. This can be observed by noticing the spikes in the figure, and the corresponding server load. A similar behavior is observed for product B.

5.1.3. Scenario 3: cataclysmic synchronizing event, very high load for short interval

We used Workload 3 to generate this scenario. Requests were generated using a Poisson process with mean 3.4 min^{-1}, which represents a moderate load for the system. In the interval 600–800 there is a sudden surge in the request arrivals. During this interval requests are generated using a Poisson process with mean 40 min^{-1}. Requests for A and B were equally likely. Fig. 6 shows a sample simulation run.

HYBRID converges to a stable price within 350 min in case of product A. But there is a sudden surge in arrivals after time 600. This is reflected in a high server load and exponentially increasing prices. HYBRID returns to the original price shortly after the surge in arrivals ends. The behavior for product B is similar. The price is nearly stable except for the occasional spikes in price due to the moderate load of the system. When the sudden surge in arrivals begins at time 600, very high prices are charged for the next 200 min. After the surge dies down, the prices fall to original levels. But owing to the moderate load, there are constant spikes in the price.

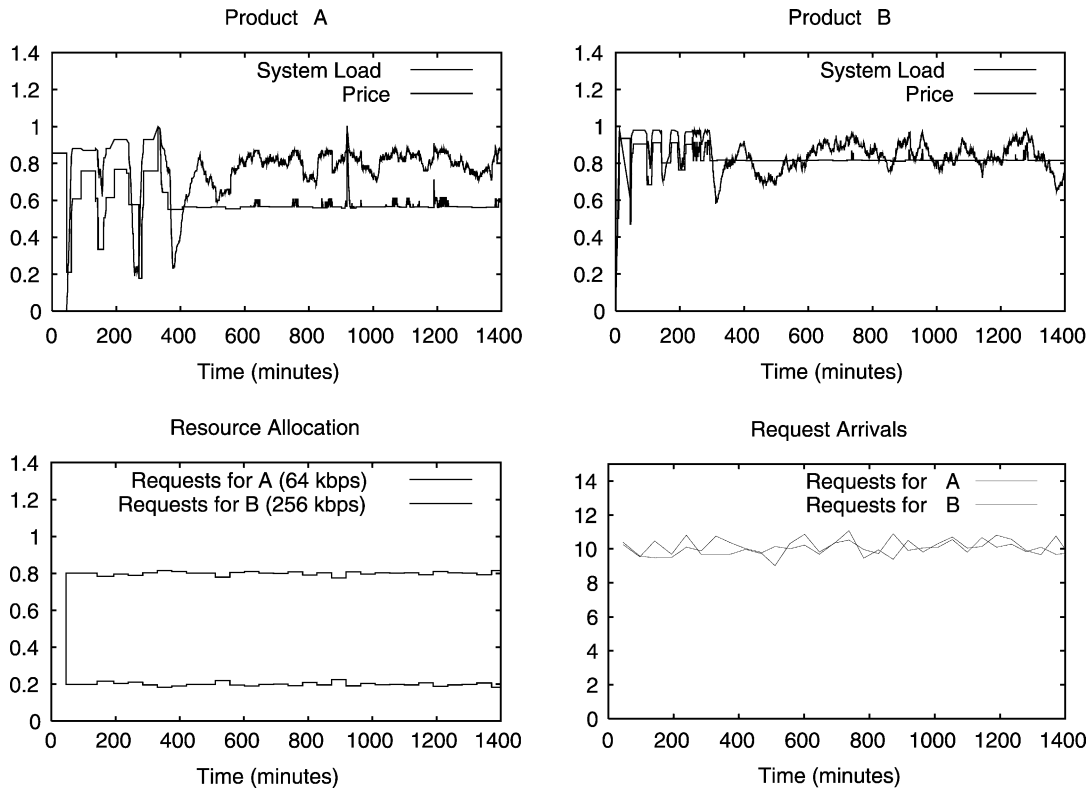

Fig. 5. High request arrivals, insufficient resources.

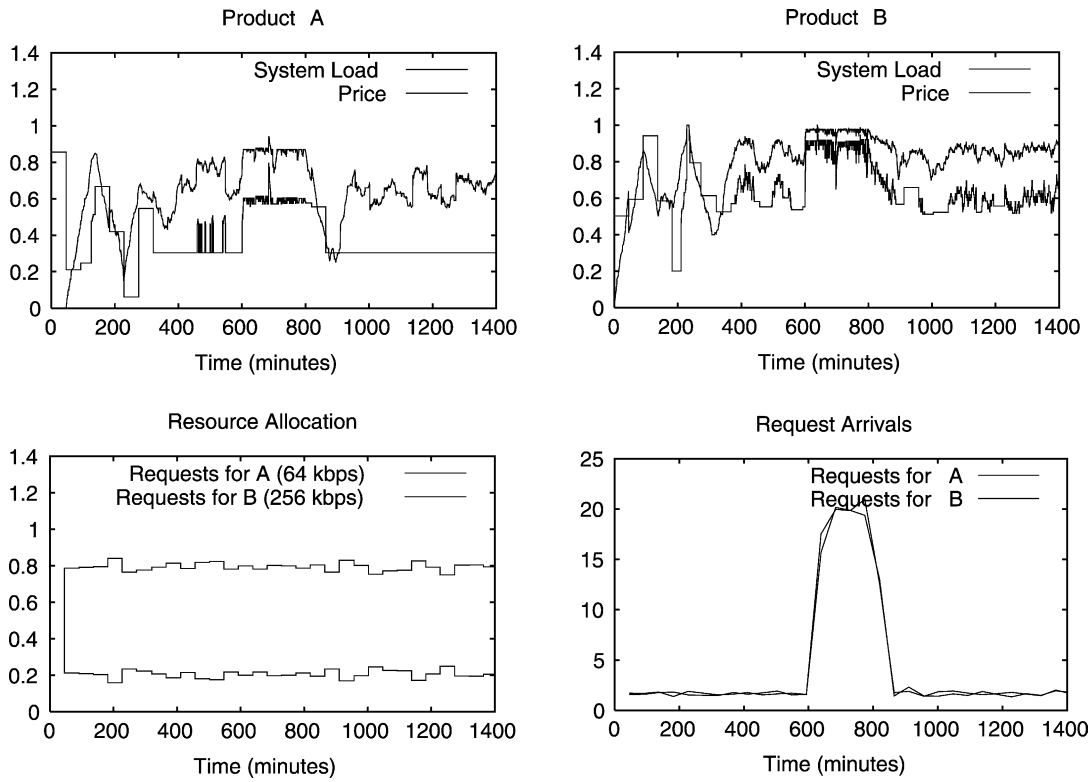

Fig. 6. Cataclysmic synchronizing event, very high load for short interval.

5.1.4. Scenario 4: a normal day at a real content delivery network

We used Workload 4 for this scenario. Requests were generated using the request records on April 15, 2001 at a real Content Delivery Network. The arrival process and the relative fraction of 64 and 256 kbp requests are shown in Figs. 2 and 3, respectively. Fig. 7 shows a sample simulation run.

This scenario generates moderate load on the server resources. However, because request arrivals are not very high, HYBRID is able to converge to a stable price, which it maintains throughout the day. Notice that even though total request arrivals are similar for this scenario and the previous scenario (except during the surge), the system is comparatively lightly loaded in the current scenario. This is because the number of requests for product *A* far exceed that for request *B*. Since requests for *A* consume fewer resources, the server is less loaded than in the previous scenario. The interesting feature in this scenario is the allocation of resources to both products. In this scenario, the relative fraction of 64 and 256 kbp requests varies throughout the day. The system responds by changing the allocations accordingly.

5.1.5. Scenario 5: sudden change in customer behavior, sufficient resources

In this scenario, we changed the customer behavior after 700 min of the simulation. The valuations for product *A* were drawn from the set {50, 60, 70, 80, 90} and the valuations for the product *B* were drawn from the set {20, 30, 40, 50, 60}. This is an unrealistic scenario because the valuations for a higher quality product are lower than the valuations for a lower quality product. However, our objective is to study how the pricing algorithm reacts to changes in customer behavior. We used Workload 4 to generate the request arrival process for this scenario. Fig. 8 shows results from a sample simulation.

Ideally, the price of product *A* should increase after time 700, and the price of product *B* should decrease after time 700. But we observe a very interesting behavior in the sample simulation. While the price of *B* converges to a lower value within 300 min (approximately five intervals of observation) after the change in customer behavior, price of *A* does not converge immediately as expected. Instead, because the old price is very low price, many customers accept the price. This leads to an increase in server load, and consequently exponentially increasing prices are charged for some time. Thus convergence time is longer when the customer valuations increase.

5.2. Comparing HYBRID with other pricing algorithms

In this section, we compare HYBRID with other pricing algorithms with the objective of evaluating its performance in terms of revenue, as also its robustness. We use two metrics for our comparison: (1) revenue earned, and (2) fraction of customers denied service in spite of accepting the quoted price. The higher the revenue earned, the better

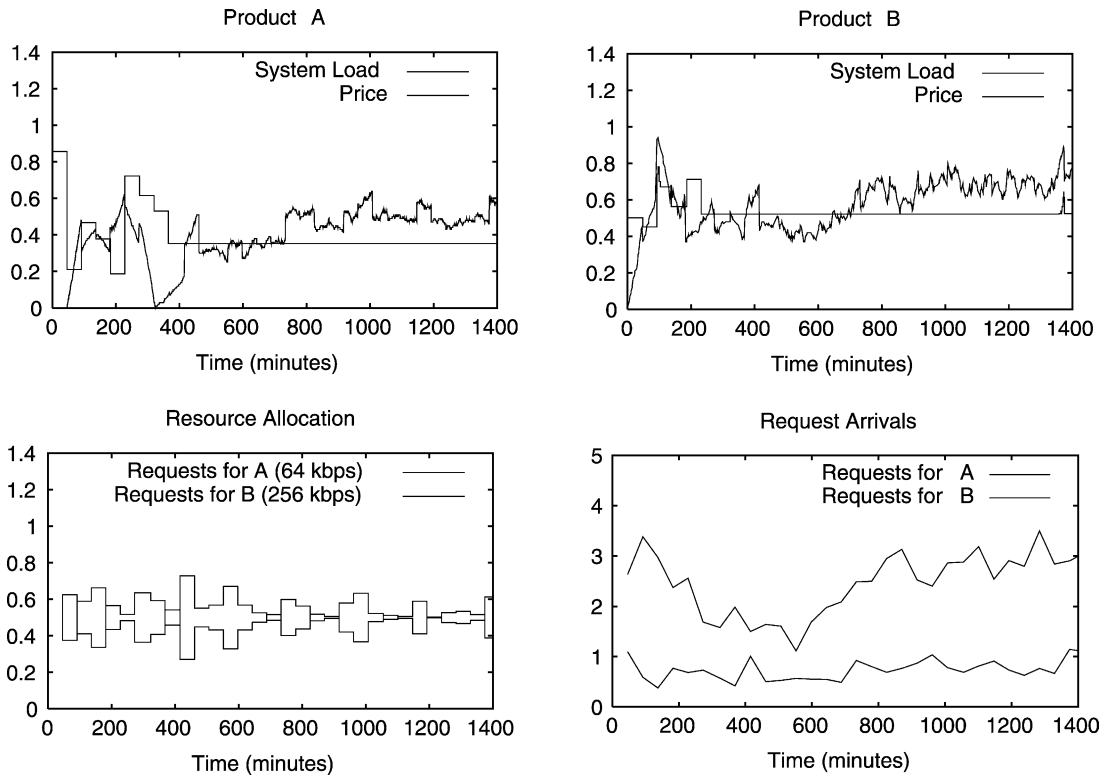

Fig. 7. Scenario from a Real content delivery network.

the algorithm. The smaller the fraction of customers denied service in spite of accepting the price, the better the algorithm. The simulation results presented in this section are averaged over five simulation runs with different seed values of the random number generator. The results are presented in tabular form as a tuple $\langle R, r \rangle$ where R is the revenue earned rounded off to the nearest 1000, and r the fraction of requests denied service.

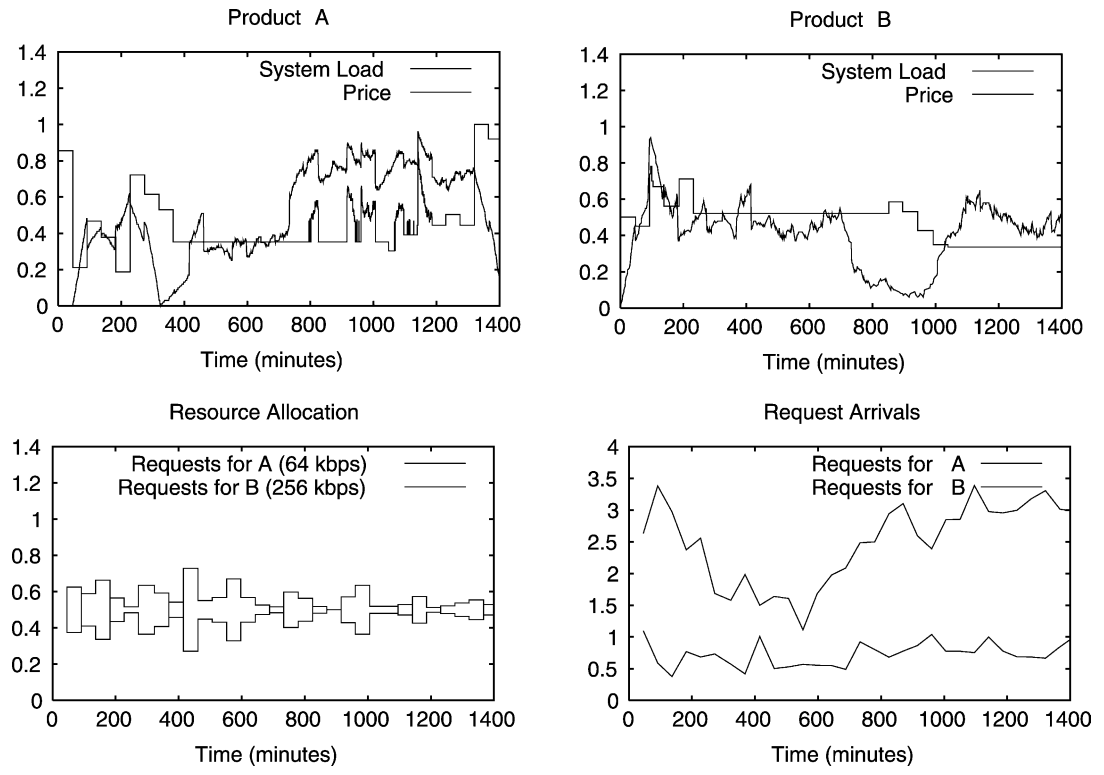

Fig. 8. Sudden change in customer behavior, sufficient resources.

We performed simulations for all the request arrival processes, all the customer valuation distributions, and all the pricing algorithms we described in Section 4. For each request arrival processes, each customer valuation distribution, and each pricing algorithm, we simulated two situations: (1) there is no change in customer behavior throughout the duration of the simulation and (2) customer behavior changes after 700 min of simulated time.

5.2.1. Simulation 1

Table 2 presents results for the set of simulations where there is no change in customer behavior throughout the simulation. As can be seen, the revenue earned by HYBRID is much higher than revenue earned by the other algorithms. On average, HYBRID earns between 75 and 200% more than the TEP and between 10 and 48% more than the DFP algorithms. However, there were some simulations in which both TEP and DFP algorithms earned revenues comparable to that earned by HYBRID. This happened when the initial price chosen was close to the ideal fixed price. Even in those simulations, the HYBRID algorithm earned as much or slightly more revenue. The DFP algorithm outperforms the TEP algorithm in all the cases shown here, mainly because it learns the customer behavior better than TEP. We also observe that for Workload 2 and 3, the service denial rate is very high for both TEP (0.24–0.30) and DFP (0.36–0.49) algorithms. This is because, they charge a low price and cannot accommodate all the requests. Such high service denial rates would be unacceptable in a commercial content delivery system. The HYBRID algorithm on the other hand, has no service denials for any of the request arrival processes. This is because of the exponentially increasing price that is charged when the server is heavily loaded. The exponentially increasing price function that we used in this article depends on the upper bound price. If the upper bound price chosen is much smaller than possible customer valuations, then HYBRID too will have service denials. But we believe that content providers can estimate upper bounds on valuations for commodity products with reasonable accuracy.

The revenues presented for the FIXED algorithm are the means across a number of equally likely prices. In our simulations, we observed that while some fixed prices generated very high revenues, other fixed prices generated zero revenue. One intuitive example is to charge $9.99 from every customer. For the customer valuation distributions we chose for these simulations, this price will generate zero revenue. We believe that, fixed prices are ideal because of their simplicity. However, finding the ideal fixed price is extremely difficult. Moreover, systems with fixed prices are not robust to heavy server loads. For example, the mean service denial rate for Workload 2 and 3 is between 0.24 and 0.36, which is clearly unacceptable.

Table 2
Simulation 1: no changes in customer behavior

	HYBRID	TEP	DFP	FIXED
Workload 1				
Uniform	⟨68, 0.00⟩	⟨28, 0.00⟩	⟨47, 0.00⟩	⟨45, 0.00⟩
Normal	⟨62, 0.00⟩	⟨25, 0.00⟩	⟨48, 0.00⟩	⟨33, 0.00⟩
Zipf	⟨48, 0.00⟩	⟨27, 0.00⟩	⟨43, 0.00⟩	⟨38, 0.00⟩
Workload 2				
Uniform	⟨220, 0.00⟩	⟨122, 0.30⟩	⟨193, 0.49⟩	⟨199, 0.36⟩
Normal	⟨190, 0.00⟩	⟨117, 0.28⟩	⟨149, 0.49⟩	⟨136, 0.30⟩
Zipf	⟨238, 0.00⟩	⟨122, 0.29⟩	⟨185, 0.36⟩	⟨196, 0.30⟩
Workload 3				
Uniform	⟨149, 0.00⟩	⟨74, 0.25⟩	⟨131, 0.40⟩	⟨124, 0.27⟩
Normal	⟨153, 0.00⟩	⟨68, 0.23⟩	⟨139, 0.42⟩	⟨89, 0.24⟩
Zipf	⟨126, 0.00⟩	⟨72, 0.24⟩	⟨122, 0.37⟩	⟨111, 0.24⟩
Workload 4				
Uniform	⟨124, 0.00⟩	⟨42, 0.00⟩	⟨84, 0.00⟩	⟨91, 0.00⟩
Normal	⟨100, 0.00⟩	⟨36, 0.00⟩	⟨89, 0.00⟩	⟨70, 0.00⟩
Zipf	⟨100, 0.00⟩	⟨41, 0.00⟩	⟨75, 0.00⟩	⟨78, 0.00⟩

5.2.2. Simulation 2

In the second set of simulations, we changed customer behavior distributions after 700 min of simulated time. The new distributions are as follows. The Uniform and Zipf distributions for product A were drawn from {20, 30, 40, 50}. For product B, the valuations were drawn from {40, 50, 60}. The $\langle \mu, \sigma \rangle$ for Normal distribution were $\langle 30, 2 \rangle$ for product A and $\langle 50, 2 \rangle$ for product B. The results of the simulations are presented in Table 3. Because the results are similar, we only present results for Workload 4. The HYBRID algorithm consistently generates high revenues in comparison to the other algorithms mainly because it learns the customer behavior by experimenting with different prices. All the results appear consistently similar to the results in the first set of simulations. Notice that the revenues are lower than in Simulation 1 because the customer valuations have decreased.

The reason why DFP and TEP do not perform well is that the algorithms do no consider resource constraints. They were primarily designed for a scenario where e-content can be delivered at leisure. We also ran other simulations to evaluate our choice of parameters for the algorithms. We only present a summary of our findings. We observed that performance of the TEP and DFP did not vary when we changed the interval after which prices are reassessed. This was because the jumping probabilities for TEP are very small, and in case of DFP, the algorithm itself is

Table 3
Simulation 2: changes in customer behavior

	HYBRID	TEP	DFP	FIXED
Workload 4				
Uniform	⟨105, 0.00⟩	⟨34, 0.00⟩	⟨70, 0.00⟩	⟨72, 0.00⟩
Normal	⟨79, 0.00⟩	⟨32, 0.00⟩	⟨70, 0.00⟩	⟨48, 0.00⟩
Zipf	⟨72, 0.00⟩	⟨33, 0.00⟩	⟨62, 0.00⟩	⟨63, 0.00⟩

independent of the interval. In case of the HYBRID algorithm, however, we observed that a small interval generates higher revenue but increases service denial rate. We found that an interval of 45 min was ideal in terms of revenue as well as service denial rate. We also observed that by increasing the big jump and small jump probability, the performance of the TEP was more erratic. The revenue did not increase or decrease in a consistent way with increasing jumping probability.

6. Conclusions

In this article we developed an approach for pricing delivery of e-content in a system with multiple LoS. The pricing scheme, called HYBRID, is dynamic because the price varied with time. The pricing scheme is based on observing customer reactions to price and provisioning of resources among the different levels of service. Resources are dynamically provisioned based on the amount of resources that requests for each LoS could consume. We presented a number of scenarios illustrating the robustness and scalability of our algorithm. We examined how price controls server load, and how it can be used effectively to increase robustness and scalability of servers. We also compared the performance of this scheme with two other simplistic dynamic pricing schemes adapted from work by other researchers, and a naive fixed pricing scheme. We performed simulations using a variety of request arrival processes and semi-realistic data to evaluate the performance of the algorithms. We observed that the HYBRID pricing scheme consistently generates high revenues across a range of customer and system profiles. We also observed that the HYBRID pricing scheme reduced the number of customers rejected service due to resource constraints mainly by charging high prices at times of peak load. We observed that the two other dynamic pricing schemes failed to generate higher revenues mainly because they do not consider resource restrictions for content delivery. Fixed pricing schemes appear to be very appealing because of their simplicity. However, they are not scalable and robust. This is because, there is no mechanism to prevent customers from purchasing during times of heavy server load.

References

[1] S. Jagannathan, J. Nayak, M. Hofmann, K.C. Almeroth, On pricing algorithms for batched content delivery systems, Journal on Electronic Commerce Research and Applications, Elsevier Science 1 (3) (2002).
[2] B. Stiller, P. Reichl, S. Leinen, Pricing and cost recovery for internet services: practical review, Netnomics—Economic Research and Electronic Networking 3 (2001) 149–171.
[3] M. Falkner, M. Devetsikiotis, I. Lambadaris, An overview of pricing concepts for broadband ip networks, IEEE Communications Review 3 (2) (2000).
[4] L.A. DaSilva, Pricing for qos-enabled networks: a survey, IEEE Communications Review 3 (2) (2000).
[5] J.O. Kephart, J.E. Hanson, J. Sairamesh, Price-war dynamics in a free-market economy of software agent, in: ALIFE VI, 1998, pp. 53–62.
[6] J. Sairamesh, J. Kephart, Price dynamics of vertically differentiated information markets, in: International Conference on Information and Computation Economics, 1998.
[7] C.H. Brooks, E.H. Durfee, R. Das, Price wars and niche discovery in an information economy, ACM Conference on Electronic Commerce, 2000, pp. 95–106.
[8] J. Mackie-Mason, J.F. Riveros, R. Gazzale, Pricing and bundling electronic information goods: field evidence, in: 17th Annual Telecom Policy Research Conference, 1999.
[9] J. Mackie-Mason, C.H. Brooks, R. Das, J.O. Kephart, R.S. Gazzale, E. Durfee, Information bundling in a dynamic environment, in: Proceedings of the IJCAI-01 Workshop on Economic Agents, Models, and Mechanisms, 2001.
[10] P. Basu, T. Little, Pricing considerations in video-on-demand systems, in: ACM Multimedia Conference, Los Angels, California, 2000.
[11] J. Wolf, M. Squillante, J. Turek, P. Yu, Scheduling algorithms for broadcast delivery of digital products, in: IEEE Transactions on Knowledge and Data Engineering, 2000.
[12] S. Chan, F. Tobagi, On achieving profit in providing near video-on-demand services, in: IEEE International Conference on Communications (ICC'99), 1999.
[13] A. Krishnamurthy, A Dynamic Resource Reservation and Pricing Policy for Scalable Video Delivery. PhD Thesis, Boston University, September 1995.
[14] S. Jagannathan, K.C. Almeroth, Price issues in delivering e-content on-demand, ACM Sigecom Exchanges 3 (2) (2002).
[15] S. Jagannathan, J. Nayak, M. Hofmann, K.C. Almeroth, A model for discovering customer value for e-content, in: ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2002.
[16] S. Jagannathan, K.C. Almeroth, The dynamics of price, revenue and system utilization, in: Management of Multimedia Networks and Services, 2001.
[17] S. Jagannathan, K.C. Almeroth, Pricing and resource provisioning for delivering e-content on demand with multiple levels-of-service, in: International Workshop on Internet Charging and QoS Technologies, 2002.