Client-Driven Price Selection for Scalable Video Streaming with Advertisements*

Musab S. Al-Hadrusi and Nabil J. Sarhan
{hadrusi,nabil}@wayne.edu

Department of Electrical and Computer Engineering, Wayne State University
5050 Anthony Wayne Drive, Detroit, MI 48202, USA

Abstract. This paper considers the scalable delivery framework of streaming video content with advertisements. In this framework, the revenues generated from the ads are used to subsidize the cost and thus attract more clients. We analyze a predictive scheme that provides clients with multiple price options, each with a certain number of expected viewed ads. The price depends on the royalty fee of the requested video, its delivery cost based on the current system state, the applied scheduling policy, and the number of viewed ads. The price is lower when the number of viewed ads is larger.

Keywords: Media streaming, periodic broadcasting, pricing, scheduling, stream merging, waiting-time prediction.

1 Introduction

Multicast-based delivery of video streams can be achieved by stream merging [15, 13, 19, 8] (and references within) or periodic broadcasting [20, 9, 21] (and references within). Stream merging reduces the delivery cost by aggregating clients into larger groups that share the same multicast streams while periodic broadcasting divides each media file into multiple segments and broadcasts each segment periodically on dedicated server channels. Successful video streaming services require incorporating effective business models. A plurality of business models are currently being utilized [14]. In this paper, we use advertisements for subsidizing the cost of premium content, such as new movies.

Incorporating advertisements in the multicast-based approach is challenging because requests (and/or streams) are aggregated to reduce the overall delivery costs. This paper utilizes the recently proposed framework for scalable delivery of media content with advertisements [2], which combines the benefits of stream merging and periodic broadcasting. Video ads are delivered on dedicated broadcasting channels, whereas stream merging is used for the primary video content. A client starts by joining an ads' broadcast channel for some time and then receives the requested video by stream merging. In [17], a waiting-time prediction algorithm has been proposed to estimate the ads' viewing period based on the requested video and the system's current state.

* This work is supported in part by NSF grant CNS-0834537.

Thus, a client is presented with the expected ads' viewing time and the associated price for streaming the requested video. The revenues generated from the ads are used to subsidize the price and thus attract more clients. Clients with larger ads' viewing time get lower prices. The algorithm considers the dynamic and complex natures of stream merging and request scheduling. The main limitation of that work is that the client is presented with only one choice of price and associated ads' viewing time.

This paper analyzes a solution that provides clients with the opportunity for selecting from multiple price options, thereby providing a better customer-centric and profit-making solution. The proposed *Client-Driven Price Selection* (CPS) approach provides clients with multiple price options, each with a certain number of expected viewed ads. The price depends on the royalty fee of the requested video, the video delivery cost (to be computed dynamically based on the current system state because of request aggregation), and the number of viewed ads (and thus the revenue generated from these ads). We modify the waiting-time prediction algorithm in [17] and request scheduling in [2] to allow for multiple options and then study the effectiveness of the overall solution in detail. As in prior work, we assume the system uses a mechanism to ensure that the clients do actually watch the ads (such as requiring their input to advance to the next ad, using interactive ads, etc.).

We study the effectiveness of the CPS approach and various scheduling policies through extensive simulation in terms of numerous performance metrics. These metrics include *waiting-time prediction accuracy*, *percentage of clients receiving waiting times*, *client defection* (i.e., turn-away) probability, *average waiting time*, *price*, *arrival rate*, and *profit*. The defection probability is the probability that customers defect because of waiting times exceeding their tolerance. The average deviation between the expected and actual times of service is used as a measure of accuracy. The accuracy decreases with the deviation. The reported average waiting time includes the time spent viewing ads and the initial waiting time to receive the ads. We consider the impact of various design and workload parameters. We combine the equation-based [2] and willingness-based [17] arrival rate models to assess the impacts of the price, purchasing capacity, and defection probability on the effective arrival rate. To account for the different ways the clients may select from the available options that meet their purchasing capacity (based on purchasing capacity and willingness model), we experiment with a variety of price selection criteria: *Random Price*, *Lowest Price*, *Median Price*, and *Highest Price*.

The rest of the paper is organized as follows. Section 2 discusses background information and related work. Section 3 presents the proposed approach. Section 4 discusses the performance evaluation methodology. Section 5 presents and analyzes the main results. Finally, conclusions are drawn.

2 Background Information

Resource sharing techniques [3, 4, 7, 10, 16, 11, 15] face the scalability challenge of multimedia streaming systems by utilizing the multicast facility. Stream merging techniques are resource sharing techniques that combine streams when possible to reduce the delivery cost. *Earliest Reachable Merge Target* (ERMT) [5, 6] is a near optimal hierarchical stream merging technique, which allows stream to merge multiple times. A

new client joins the closest reachable stream (target) and receives the missing portion by a new stream. After the merger stream finishes and merges into the target, the latter can get extended to satisfy the playback requirement of the new client(s), and this extension can affect its own merge target.

For stream merging, a scheduling policy is used to select a video to service when a *channel* becomes available. A channel is a set of resources needed to deliver a video stream. All requests in the selected video can be serviced using only one channel. The number of channels is called *server capacity*.

The main scheduling policies include *First Come First Serve* (FCFS) [4], *Maximum Queue Length* (MQL) [4], *Maximum Factored Queue Length* (MFQL) [1], and *Minimum Cost First* (MCF) [19]. MCF achieves the best overall performance by capturing the significant variation in stream lengths caused by stream merging. *MCF-P*, which is the preferred implementation, selects the video with the least cost per request.

For supporting video streaming with advertisements, a scalable delivery framework was proposed in [2]. This framework has the following main characteristics. (1) Clients start by joining an ads' broadcast channel for some time and then receive the requested video by stream merging. (In this paper, "requested video", "actual video", or "video", unless otherwise indicated, refer to one of a primary videos and not an ad.) (2) Ads are combined and broadcast on dedicated server channels. Hence, when beginning to listen to an ads' channel, the client views different ads until streaming of the desired video finishes. Multiple channels can be used with time-shifted versions of the combined ads, as shown in Figure 1, to reduce the waiting time for reaching the beginning of an ad. With N_{adCh} channels, the maximum value of this time is ad_len/N_{adCh}, where ad_len is one ad length. (3) Ads are only viewed prior to watching the actual video to allow for uninterrupted viewing and more enjoyable playback experience.

For the scalable delivery framework, two modified versions of MCF-P scheduling policy were proposed in [2] to ensure that ads are viewed by a large number of clients: *Any N* and *Each N*. *Each N* considers a video for scheduling only if each waiting request for it has viewed at least N ads, whereas *Any N* considers a video for service only if any one of its waiting requests has viewed at least N ads.

3 Client-Driven Price Selection

This paper explores the idea of presenting each client with multiple price options. Providing clients with multiple price options enhances customer satisfaction and system profitability. The proposed *Client-Driven Price Selection* (CPS) approach is targeted for the scalable video delivery framework with advertisements [2]. This approach performs waiting-time prediction and provides the client with the list of expected waiting times (or times of service) and their associated prices. The waiting-time prediction is done by modifying the prediction algorithm in [17]. The waiting-time is essentially the total time between the request arrival and the commencement of streaming of the requested video.

The idea of the CPS algorithm can be described as follows. When a new request is made, it is mapped to the ads' channel with the closest ad start (or end) time. The algorithm examines the ad start times on that channel in the order of closeness to the

current time for possible assignment as the expected time for that request. Because only multiple ads can be viewed by clients, the later ad start times represent the different possible service times. At each ad start time, the server estimates the number of available channels and predicts the videos that can be serviced at that time. When the requested video is the expected video to be serviced, the corresponding ads' viewing time and associated price is added to the list of choices to present to the client. The process continues until, the *prediction window* (Wp) is exceeded. This window provides a tradeoff between the implementation complexity and the number of price options the client receives as well a tradeoff between the prediction accuracy and the percentage of clients receiving expected times. If the video prediction does not return any expected time, the average waiting time for the requested video is used instead.

Figure 1 illustrates how CPS generally works with $W_p = 3$ ad lengths. When a new client makes a request at time T_{Now}, the algorithm examines the ad start times within the prediction window in the order T_0, T_1, T_2, and T_3. For each one of these times, if the requested video is selected in the prediction, the corresponding ads' viewing time and price is added to the list of client's choices. The list for example may include (1 ad, $\$2.2$) and (3 ads, $\$1.8$), assuming that the video is selected at times T_1 and T_3. The server has to make sure that the client will view at minimum the selected number of ads by placing an additional constraint on the scheduling policy.

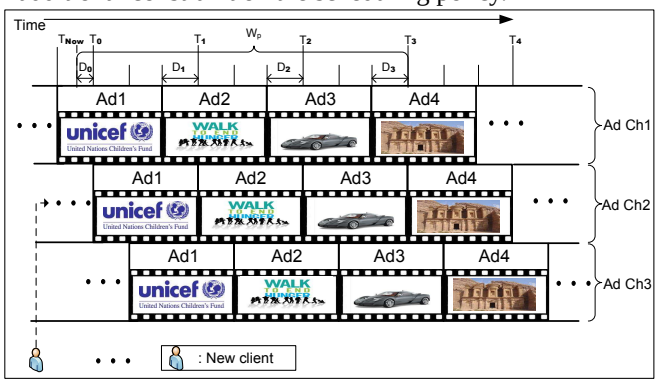


Fig. 1: Illustration of the CPS Algorithm

Figure 2 shows a simplified version of the algorithm, which is performed upon the arrival of request R_i to video v_j. This algorithm extends the algorithm proposed in [17] to allow multiple price options. Providing multiple price options is enabled by continuing to provide expected ads' viewing times and associated prices to the client even after the prediction algorithm determines an expected time for service for that request. Thus, if the request is expected to be serviced at time T_i, the CPS algorithm returns an expected number of ads and a matching price to the client and continues to find other expected service times until the prediction window is exceeded. When determining the latter expected numbers of ads and associated prices, the CPS algorithm acts as if the request has not been serviced earlier. As discussed earlier, the implementation of the scheduling policy (not shown in the Figure) has also to be changed in order to enforce that the client views the number of ads that he/she selected earlier. To make the paper as self-contained as possible, let us briefly discuss how to predict the videos to be served

for $(v = 0; v < N_v; v++)$ *// Initialize*
  $assigned\_time[v] = -1$; *// Not assigned expected time*
$adChNo$ *= Get label # of the ads' channel with the closest start time;*
T *= Get next ad's start time;* $T_0 = T$; $examined\_times = 0$;
// Find number of available channels at time T
$N_c = available\_channels + will\_be\_available(T_{Now}, T)$;
while $(T < T_{Now} + W_p)$ { *// Loop till prediction window is exceeded*
  for $(v = 0; v < N_v; v++)$ {
    if $(isQualified(v, T, adChNo))$ {
      if $(assigne\_time[v] == -1)$
        $expected\_qlen = qlen(v, adChNo) + \lambda[v] \times$
        $((T - T_{Now}) + examined\_times \times ad\_len/N_{adCh})$;
      else *// Video v has been assigned an expected time*
        $expected\_qlen = \lambda[v] \times (T - assigned\_time[v])/N_{adCh}$;
      $objective[v]$ *= find scheduling objective for video v;*
    } *// end if* $(isQualified(v, T, adChNo))$
    else $objective[v] = -1$; *// v is not qualified*
  } *// end for* $(v = 0; v < N_v; v++)$
  while $(c = 0; c \leq N_c; c++)$ { *//for every available channel*
    // Find the expected video to serve at time T
    $expected\_video$ *= find video with maximum nonzero objective;*
    if$(expected\_video == v_j)$ {
      Push T into expected time que Que_{ET} of request R_i;
      TempPrice = CalculatePrice(T,v_j);
      Push TempPrice into price que Que_{price} of request R_i;
      break; }
    else { $assigned\_time[expected\_video] = T$;
        $objective[v] = -1$; } *// -1 means can't be selected again*
  } *// end while* $(c = 0; c \leq N_c; c++)$
  $T = T + ad\_len$; *//Proceed to the next edge*
  // Find number of available channels at time T
  $N_c = left\_over + will\_be\_available(T - ad\_len/N_{adCh}, T)$;
  $examined\_times++$;
} *// end while* $(T < T_{Now} + W_p)$
Present Que_{Price} to client and then clear it.

Fig. 2: Simplified Algorithm for CPS [performed upon arrival of request R_i to Video v_j]

at any particular ad start time T. First, the algorithm determines the videos that will be qualified at that time based on any minimum ads' viewing constraints, imposed by Any N or Each N. For each video that qualifies, the algorithm estimates its waiting queue length at time T, based on the video arrival rates, which are to be computed periodically but not frequently. As in [17], the expected queue length for video v at time T can be found as follows:

$$expected\_qlen[v] = qlen(v, adChNo) + \lambda[v] \times ((T_0 - T_{Now}) + examined\_starts \times ad\_len/N_{adCh}), \quad (1)$$

where $qlen(v, adChNo)$ is the queue length of video v on channel $adChNo$ at the current time (T_{Now}), $\lambda[v]$ is the arrival rate for video v, ad_len is the ad length, N_{adCh} is the number of ads' channels, T_0 is the nearest ad start time, and $examined\_starts$ is the number of examined ad start times so far during the current run of the prediction algorithm. Equation (1) assumes that video v has not been identified before as the expected video to be serviced at an earlier ad start time. Otherwise, the expected arrivals will have to be found during the time interval between T and the latest assigned time ($assigned\_time[v]$) at which video v is expected to be served [17]. In Figure 2, N_c represents the number of available server channels at ad start time T. This number is equal to the number of channels that will be available prior to T plus a left-over value for the number of unused channels at prior ad start times [17].

The pricing depends on the royalty fee of the requested video, the video delivery cost (to be computed dynamically based on the current system state because of request aggregation), and the number of viewed ads (and thus the revenue generated from these ads). Note that the delivery cost varies with the current access rate. Although the prediction algorithm is highly accurate as will be shown later, clients who view more ads than expected should be provided with compensation credits, which can be used to reduce the number of ads to be viewed when accessing other primary media later on.

4 Performance Evaluation Methodology

Table 1 summarizes the workload characteristics. Like most prior studies, we generally assume that the arrival of the requests to the server follows a Poisson Process with an average arrival rate λ. Hence, the inter-arrival time is exponentially distributed with a mean $T_{avg} = 1/\lambda$. Additionally, we assume that the access to videos is highly localized and follows a Zipf-like distribution. With this distribution, the probability of choosing the n^{th} most popular video is $C/n^{1-\theta}$ with a parameter θ and a normalized constant C. The parameter θ controls the skew of video access. Note that the skew reaches its peak when $\theta = 0$, and that the access becomes uniformly distributed when $\theta = 1$. We assume a value of 0.271 [18].

The waiting tolerance of clients is characterized as follows: a client who chooses a certain expected time of service (and thus an associated number of expected ads and a price) waits for service until that expected times plus an added tolerance value Wad (a variable from 0 to 9 ad lengths with default value of 2). The waiting tolerance of all other clients follows an exponential distribution with mean μ_{tol}

Estimating the overall revenue and profit is challenging because the price influences the arrival rate and number of streams delivered. Subsidizing the price can attract more clients and can eventually increase the overall revenue and profit. By increasing the arrival rate, the delivery costs also decrease because of the higher degrees of stream merging. We combine *equation-based* and *willingness-based* arrival rate models. In the first, the arrival rate changes dynamically based on the defection probability [2]. The defection probability is the probability that clients leave without being serviced because of waiting times exceeding their tolerance. We experiment with the following function:

$$\lambda = \frac{c_1(1-d)}{c_2 + c_3 d^2} \tag{2}$$

Table 1: Summary of Workload Characteristics

Parameter	**Model/Value(s)**
Request Arrival	Poisson Process
Request Arrival Rate	Variable, Default = 40 Requests/min
Server Capacity	200-550
Video Access	Zipf-Like, Skew Parameter $\theta = 0.271$
Movie-Related Characteristics	80 120-min movies
Waiting Tolerance Model for clients without expected times of service	Poisson, min= 3 ads, mean= 5 ads, max= 8 ads
Waiting Tolerance Model for clients with expected times of service	Expected Service Time + Wad, Wad: Variable, Default= 2 Ad lengths
Ad-Related Characteristics	Ad Length= 30 sec, # different ads= 8, # ads channels= 3
Minimum Ads Constraint (N)	Variable, Default = 2
Prediction Window (W_p)	Variable, Default = 9
Qualification Threshold (Q_{Th})	Variable, Default = 0.25
Scale (b), Shape (α), Elasticity (δ)	Variables, Defaults= 1.0, 1, 7, resp.
Equation-Based Model Constants	$c1 = 60$, $c2 = 0.5$, $c3 = 1$

where d is the defection probability, and c_1, c_2, and c_3 are constants. In the second model, we utilize client purchasing capacity and willingness models. The capacity of a client to spend for a particular service or product can be modeled using Pareto distribution [12]. The Pareto probability density function can be given as follows in equation (3):

$$f_p(x) = \alpha \times b \times x^{-(\alpha+1)} \quad for \quad x \geq b, \tag{3}$$

where b (also called *scale*) represents the minimum value of x, and α represents the shape of the distribution. Most clients have capacities close to b. The distribution is more skewed for larger values of α. Hence, as α increases, fewer clients can pay much more than b. Clients with larger capacities are more likely to spend more. The willingness probability of a client with capacity y to pay for a product or service with price p can be given by

$$Prob(willingness) = \begin{cases} 1 - (\frac{p}{y})^{\delta} & 0 \leq p \leq y \\ 0 & p > y, \end{cases} \tag{4}$$

where δ is the *elasticity* [12]. As δ increases, more clients are willing to spend.

In prior work, the equation-based model [2] and the willingness-based model [17] were used separately. In this paper, we combine these two models as they are not exclusive. The equation-based model captures the impact of the current defection probability on the future arrival rate, whereas the equation-based model captures the impacts of the price and purchasing capacity on the willingness of the client to purchase the streaming service. Therefore, each one of these two models impacts the arrival rate in a different way. Figure 3 illustrates how the two models are combined.

We consider a commercial *Movie-on-Demand* system. Without loss of generality, we assume a *cost-plus* pricing. The price in the considered system covers the movie royalty fee, delivery fee, and operational cost minus subsidization credit. In the discussed example, the revenue per ad per client is 10 cents, the movie royalty fee is 70

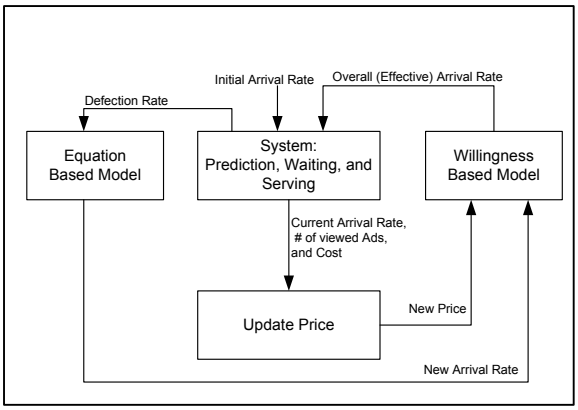


Fig. 3: Combining the Equation-Based and Willingness-Based Models

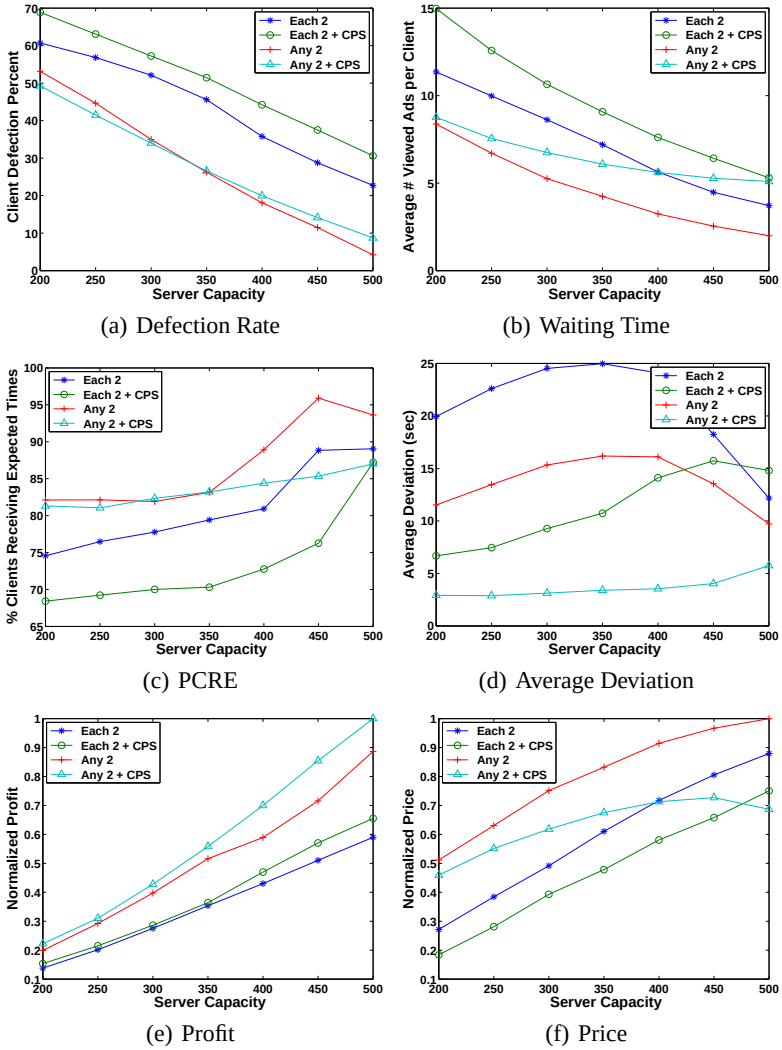
cents, and the delivery cost per GB is 50 cents. Based on service positioning analysis, the service provider seeks to get 70 cents per movie request to cover their operational cost and attain the sought profit. A fixed fraction of the 70 cents is used as a profit.

5 Result Presentation and Analysis

We conducted extensive simulation using many different workload and system parameters. For space limitation, we only show the main results. Only the results for ERMT are shown because it is the most efficient stream merging policy. We assume that the client selects the lowest price unless otherwise indicated. We also consider the impact of the used price selection scheme. The analyzed **performance metrics** include: *waiting-time prediction accuracy*, *percentage of clients receiving waiting times* (PCRE), *client defection* (i.e., turn-away) probability, *average number of viewed ads*, *price*, *arrival rate*, and *profit*. The average deviation between the expected and actual times of service is used as a measure of accuracy. The accuracy decreases with the deviation. The waiting time includes the time spent viewing ads and the initial waiting time to receive the ads. Profit is considered as the most important metric. All other metrics influence the profit. The average number of viewed ads, prediction accuracy, and PCRE are important measures of Quality-of-Service (QoS).

Figure 4 illustrates the effectiveness of CPS for "Any 2" and "Each 2" Scheduling with different server capacities. The value 2 is selected optimally based on extensive analysis. Note that increasing this value increases the ads' viewing time but leads to higher underutilization of resources. The results show that CPS may slightly increase the defection rate, but this is due to the higher arrival rate achieved by accepting more client requests, as shown in Figure 5(a). Figure 5(b) shows the same but with different prediction windows. CPS increases the profit for both scheduling polices. "Any 2" Scheduling with CPS achieves the best overall performance. The average deviation between the predicted and actual waiting times is always less than one ad length and is within 6 seconds with the best scheduling policy. The percentage of clients receiving expected times (PCRE) is always larger than 67%.

Figure 6 demonstrates the impact of various price selection schemes: random, lowest, median, and highest. Note that there is a tradeoff between price and waiting time.



(a) Defection Rate (b) Waiting Time

(c) PCRE (d) Average Deviation

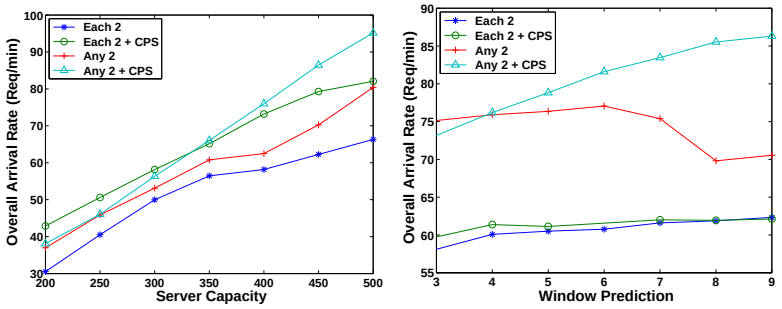
(e) Profit (f) Price

Fig. 4: Effectiveness of the Proposed CPS Scheme [ERMT, Least Price Selection]

The price is lower with a larger waiting time. In reality, different clients will have different criteria, and this motivates investigating random selection. Interestingly, although these schemes achieve significant variation in waiting time and price, they perform closely in terms of the profit.

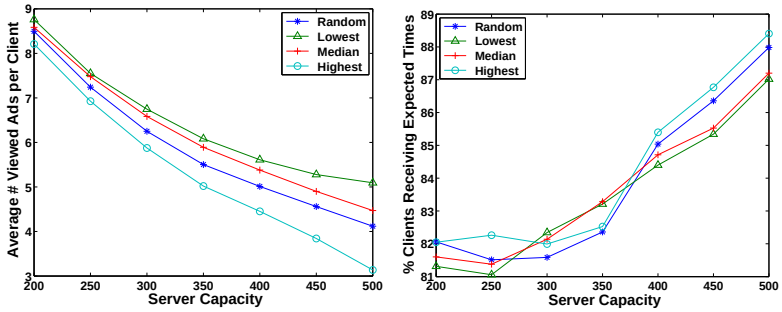
6 Conclusions

We have analyzed a predictive scheme, called *Client-Driven Price Selection* (CPS), which provides clients with multiple price options, each with a certain number of expected viewed ads. The main results can be summarized as follows. (1) The proposed

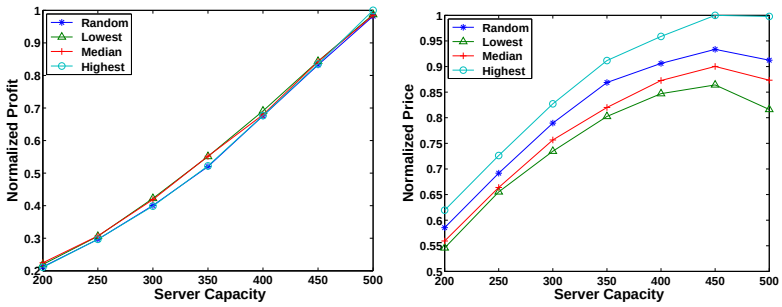


(a) Server Capacity (b) Window Prediction

Fig. 5: Overall Arrival Rate [ERMT, Any 2, CPS, Least Price Selection]



(a) Waiting Time (b) PCRE



(c) Profit (d) Price

Fig. 6: Impact of Various Price Selection Schemes

CPS approach enhances the revenue and profit by giving multiple price choices to the client, thereby attracting more clients. (2) CPS is best when combined with Any N scheduling and ERMT. (3) The achieved waiting time prediction accuracy with this combination is within 6 seconds (20% of an ad length).

References

1. Aggarwal, C.C., Wolf, J.L., Yu, P.S.: The maximum factor queue length batching scheme for Video-on-Demand systems. IEEE Trans. on Computers 50(2), 97–110 (Feb 2001)

2. Al-Hadrusi, M., Sarhan, N.J.: A scalable delivery framework and a pricing model for streaming media with advertisements. In: Proc. of SPIE/ACM Multimedia Computing and Networking Conference (MMCN) (Januray 2008)
3. Cai, Y., Hua, K.A.: An efficient bandwidth-sharing technique for true video on demand systems. In: Proc. of ACM Multimedia. pp. 211–214 (Oct 1999)
4. Dan, A., Sitaram, D., Shahabuddin, P.: Scheduling policies for an on-demand video server with batching. In: Proc. of ACM Multimedia. pp. 391–398 (Oct 1994)
5. Eager, D.L., Vernon, M.K., Zahorjan, J.: Optimal and efficient merging schedules for Video-on-Demand servers. In: Proc. of ACM Multimedia. pp. 199–202 (Oct 1999)
6. Eager, D.L., Vernon, M.K., Zahorjan, J.: Minimizing bandwidth requirements for on-demand data delivery. IEEE Trans. on Knowledge and Data Engineering 13(5), 742–757 (Sept 2001)
7. Gao, L., Kurose, J., Towsley, D.: Efficient schemes for broadcasting popular videos. In: Proc. of the Int'l Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV) (July 1998)
8. Gill, P., Shi, L., Mahanti, A., Li, Z., Eager, D.: Scalable on-demand media streaming for heterogeneous clients. ACM Transactions on Multimedia Computing, Communications, and Applications (ACM TOMCCAP) 5(1), 1–24 (Oct 2008)
9. Hu, A.: Video-on-Demand broadcasting protocols: A comprehensive study. In: Proc. of IEEE INFOCOM (April 2001)
10. Hua, K.A., Cai, Y., Sheu, S.: Patching: A multicast technique for true Video-on-Demand services. In: Proc. of ACM Multimedia. pp. 191–200 (1998)
11. Huang, C., Janakiraman, R., Xu, L.: Loss-resilient on-demand media streaming using priority encoding. In: Proc. of ACM Multimedia. pp. 152–159 (Oct 2004)
12. Jagannathan, S., Almeroth, K.C.: The dynamics of price, revenue, and system utilization. In: Proc. of the IFIP/IEEE International Conference on Management of Multimedia Networks and Services. pp. 329–344 (2001)
13. Ma, H., Shin, G.K., Wu, W.: Best-effort patching for multicast true VoD service. Multimedia Tools Appl. 26(1), 101–122 (2005)
14. Rayburn, D.: Streaming and Digital Media: Understanding the Business and Technology. Focal Press (2007)
15. Rocha, M., Maia, M., Cunha, I., Almeida, J., Campos, S.: Scalable media streaming to interactive users. In: Proc. of ACM Multimedia. pp. 966–975 (Nov 2005)
16. Sarhan, N.J., Das., C.R.: Caching and scheduling in NAD-based multimedia servers. IEEE Trans. on Parallel and Distributed Systems 15(10), 921–933 (Oct 2004)
17. Sarhan, N.J., Al-Hadrusi, M.: Waiting-time prediction and QoS-based pricing for video streaming with advertisements. In: Proc. of IEEE International Symposium on Multimedia (ISM) (December 2010)
18. Sarhan, N.J., Das, C.R.: A new class of scheduling policies for providing time of service guarantees in Video-On-Demand servers. In: Proc. of the 7th IFIP/IEEE Int'l Conf. on Management of Multimedia Networks and Services. pp. 127–139 (Oct 2004)
19. Sarhan, N.J., Qudah, B.: Efficient cost-based scheduling for scalable media streaming. In: Proc. of Multimedia Computing and Networking Conf. (MMCN) (January 2007)
20. Shi, L., Sessini, P., Mahanti, A., Li, Z., Eager, D.L.: Scalable streaming for heterogeneous clients. In: Proc. of ACM Multimedia. pp. 337–346 (Oct 2006)
21. Tantaoui, M.A., Hua, K.A., Do, T.T.: Broadcatch: A periodic broadcast technique for heterogeneous Video-on-Demand. IEEE Trans. on Broadcasting 50(3) (Sept 2004)