Noname manuscript No.
(will be inserted by the editor)

A Scalable Delivery Solution and a Pricing Model for Commercial Video-on-Demand Systems with Video Advertisements

Musab S. Al-Hadrusi · Nabil J. Sarhan

Received: date / Accepted: date

Abstract This paper presents a scalable delivery solution for commercial near on-demand video streaming systems with an associated pricing model. The proposed delivery solution combines the benefits of periodic broadcasting and stream merging, thereby enabling scalable video delivery. Video advertisements are delivered to the clients prior to viewing the requested videos. The revenues generated from the ads are used to subsidize the price of the requested videos. The pricing is determined based on the total ad viewing time. The proposed solution includes an efficient ad allocation scheme and a new constraint-based scheduling approach. In addition, the paper investigates how targeted advertisements can be efficiently supported. Furthermore, we investigate the effectiveness of the overall solutions and analyze and compare the effectiveness of various scheduling policies and ad allocation alternatives in terms of several metrics, including client defection probability, average number of viewed ads per client, price, channel utilization, revenue, and profit.

Keywords Periodic broadcasting · pricing · scheduling · stream merging · supporting advertisements · targeted advertisements · video streaming.

1 Introduction

Multimedia-capable devices have grown rapidly and have become more diverse, including mobile devices, home media centers, as well as regular computers. Video streaming is one of the main multimedia applications and is being used by many industries as the core of their businesses. These businesses span several areas, including social media, video communication, video-on-demand (VOD), and live streaming.

Unfortunately, the distribution of video streams faces a significant scalability challenge due to the high server and network requirements. Multicast-based video delivery can be used to address this challenge. This delivery can be done on a *client-pull* or a *server-push* fashion, depending on whether the channels are allocated on demand or reserved in advance, respectively. The first category includes *stream merging* techniques [2–7], which reduce the delivery cost by aggregating clients into larger groups that share the same multicast streams. The achieved degrees of resource sharing depend greatly on how the waiting requests are scheduled for service. The second category consists of *Periodic Broadcasting* techniques [8–12], which divide each video file into multiple segments and broadcast each segment periodically on dedicated server channels. They are cost-performance effective for highly popular content but lead to channel underutilization when the request arrival rate is not sufficiently high. This paper combines the advantages of both stream merging and periodic broadcasting.

The overwhelming majority of prior studies on multicast-based media delivery focused on regular content without any video advertisements. The use of ads, however, is important for many reasons, including the following. (1) Ads generate revenue, which can be used to pay for the service cost, generate profit, or subsidize the paid content. (2) A streaming solution that supports ads can essentially convert passive startup waiting times for service to active waiting times (i.e., watching ads while waiting for the playback of the desired media content). Today, even for short videos with medium quality, users of online video websites may ex-

M. Al-Hadrusi and N. Sarhan
5050 Anthony Wayne Dr.,
Department of Electrical and Computer Engineering
Multimedia Computing and Networking Research Lab
Wayne State University
Detroit, MI 48202
E-mail: {hadrusi,nabil}@wayne.edu
A preliminary version of this paper was presented at MMCN 2008 conference [1].

perience significant delays [13]. The transition, in the near future, to streaming long videos (such as full-length movies) at high quality may lead to even longer delays. (3) Many users like to watch some types of ads, such as movie trailers, to know about other interesting movies to watch. (4) Using ads results in better aggregation of the requests and thus can reduce the delivery costs.

This paper presents a scalable delivery solution and a pricing model for commercial VOD systems with video ads. The delivery solution combines the benefits of stream merging and periodic broadcasting. A client starts by joining an ad broadcasting channel for some time and then receives the primary (i.e., requested premium) video by stream merging. A client joins the ad channel only at the beginning of an ad and leaves it at the end of an ad. A client can be any device, such as computer, mobile device, multimedia box or smart TV. The overall solution includes a new ad allocation solution that allocates ads efficiently on ad broadcasting channels so as to reduce the initial waiting time before viewing the first ad. It also includes a new constraint-based request scheduling approach for the primary videos. We present four scheduling policies. For ad allocation, we present two schemes for ordering the ads on the broadcasting channels: *optimal* and *heuristic*. In addition, we discuss how to efficiently support *targeted ads* and present three alternatives. With targeted ads, clients receive ads matching their interests. In the overall solution, the revenues generated from the ads are used to subsidize the price, thereby attracting more clients and increasing the overall revenue.

We study, through extensive simulation, the effectiveness of the overall solutions and analyze and compare the effectiveness of various scheduling policies, ad ordering schemes, and alternative methods for supporting targeted ads. We consider numerous performance metrics, including client defection probability, average number of viewed ads per client, price, channel utilization, revenue, and profit. The defection probability is defined as the probability that clients leave the system without being serviced because of waiting times exceeding their tolerance. We capture the impacts of client purchasing capacity and willingness models [14] as well as client defection probability on the effective request arrival rate.

The rest of the paper is organized as follows. Section 2 discusses background information and related work. Section 3 presents an overview of the overall proposed solution and associated pricing model. Section 4 discusses the ad allocation solution, including the support for targeted ads. Section 5 presents the proposed constraint-based scheduling approach. Subsequently, Section 6 discusses the performance evaluation methodology and presents the main results.

2 Background Information and Related Work

Let us now discuss the three main aspects in video streaming systems that are related to the proposed solution: handling VOD scalability through multicast, request scheduling, and supporting video advertisements.

2.1 Handling VOD Scalability through Multicast

The main approaches that have been used to address the scalability challenge are *Content Distribution Networks* (CDNs) and *Peer-to-Peer (P2P)*. While the first approach requires maintaining a huge number of geographically distributed servers [15], the second still needs central servers [16,17]. Both of these approaches mitigate the scalability problem but do not eliminate it because the fundamental problem is due to unicast delivery [17]. Furthermore, using multicast delivery schemes in fully owned networks, such as cable TV, dish networks, and corporate/university networks is currently applicable.

In this paper, we use the multicast-based video delivery approach, which can be classified into two broad categories: *stream merging* and *periodic broadcasting*. Stream merging techniques [2–7] combine streams when possible to reduce the delivery cost. These techniques include *Earliest Reachable Merge Target* (ERMT) [2,18], which is a near optimal hierarchical stream merging technique. It requires two download channels and makes each stream sharable, leading to a dynamic merge tree. 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. Figure 1 illustrates how ERMT works. A special case of ERMT, called Patching [19] has been implemented and tested in an end-to-end environment [20]. Patching allows streams to merge only once. The implementation has been performed over the Real Time Streaming Protocol (RTSP) [21], with the multicast control being carried out using the Multicast Control Protocol (MLD) [22].

Whereas stream merging techniques deliver data in a *client-pull* fashion (i.e, on-demand), periodic broadcasting techniques [8,9,12,23] employ the *server-push* approach. Periodic broadcasting methods divide each supported video into multiple segments and broadcast them periodically on dedicated channels. Thus, they can service unlimited number of clients, but can be used only for the most popular videos, and they require clients to wait until the next broadcast times of the first segments. Moreover, server channels may become underutilized when videos are requested infrequently.

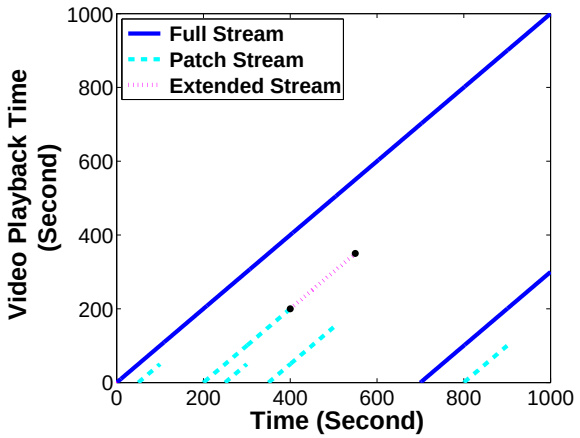


Fig. 1 An Illustration of ERMT

Periodic broadcasting can be used in Near VOD (NVOD) systems, whereas stream merging can be used in both NVOD and True VOD (TVOD) systems. In TVOD, all requests are serviced immediately. In most practical situations, however, requests may experience delays, leading to NVOD services [24]. This paper combines the advantages of stream merging and periodic broadcasting and deals with providing a NVOD service.

Multicast-based video delivery has faced major challenges, including implementation complexity and supporting heterogeneous receivers and interactive operations. These challenges have recently been addressed by many studies, including [5, 12, 17].

2.2 Request Scheduling

When stream merging is used, request scheduling becomes a major issue. In this case, the video streaming server maintains a waiting queue for every video and applies a scheduling policy to select an appropriate video for service whenever it has an available *channel*. A channel is a set of resources (network bandwidth, disk I/O bandwidth, etc.) needed to deliver a multimedia stream. All requests in the selected queue can be serviced using only one channel. The number of channels is referred to as *server capacity*.

The main scheduling policies include *First Come First Serve* (FCFS) [25], *Maximum Queue Length* (MQL) [25], and *Minimum Cost First* (MCF) [26]. These policies are described in Table 1. MCF achieves the best overall performance by capturing the significant variation in stream lengths caused by stream merging techniques through selecting the requests requiring the least cost. The length of a stream (in time) is directly proportional to the cost of servicing that stream since the server allocates a channel for the entire time the stream is active. In this paper, we use the preferred implementation of MCF, which selects the queue with the least cost per request and treats regular streams in a preferential manner because they are shared by future patches.

Table 1 Main Scheduling Policies

Scheduling Policy	Selects the Video with the
First Come First Serve (FCFS)	oldest waiting request
Maximum Queue Length (MQL)	longest waiting queue
Minimum Cost First (MCF)	least required cost

2.3 Supporting Video Advertisements

Using video ads has been adopted recently in many video streaming systems, including YouTube and Hulu. In YouTube, the video ads are shown prior to viewing the requested video. With Hulu, however, the client has the option of viewing a long period of ads at the beginning or viewing multiple ads while the requested video is being played back. In both systems, both the video ads and the requested videos are streamed to clients by unicast streams.

A plurality of business models are currently being utilized by video streaming companies for servicing premium content [27], including (i) *pay-per-view*, in which the client pays in advance for the requested media, (ii) *subscription-based*, in which the client pays a monthly fee, (iii) *free with advertisement*, in which the client watches the media for free in exchange of viewing advertisements in the beginning or during streaming the requested media, and (iv) *hybrid model*, in which the client has the option to pay for a subscription or to watch subsidizing ads. These models have been adopted by Amazon video-on-demand, Netflix, YouTube, and Hulu, respectively. While the first two models promise ad-free viewing, clients may like to view some types of ads, such as trailers for new movies. More importantly, incorporating ads has the advantage of providing content for free or at a reduced price. In this paper, we use ads for subsidizing the cost of the premium content.

Supporting ads has been discussed in only few research studies. In [28], the primary data and ads are delivered using piggybacking on the same channels. Piggybacking adjusts the movie playback rate so that two streams can merge with each other, thereby impacting the playback quality and suffering from technical challenges and complexities to change the movie playback rate. Moreover, ads are inserted randomly multiple times during the playback of the primary media. Study [29] provided a general discussion of pricing based on the techniques in [28]. It discussed the general relationships among the arrival rate, price, and ratio of ads to the total client viewing time. Ads targeting using videos is one of the most emerging topics. Many issues such as ads-clients matching, ads viewing time and ads insertion point are still under research and investigation [30].

3 Proposed Overall Video Delivery Solution and Associated Pricing Model

This paper considers the design of a commercial on-demand video streaming system with video advertisements. Figure 2 shows an overview of the system. The system consists of a video streaming server and clients (such as computers, mobile devices, and smart TVs), which are connected through a multicast-enabled network (such as corporate or university networks, Cable TV networks, Internet2, and eventually the Internet). The server employs channels for supporting stream merging and others for periodic broadcasting.

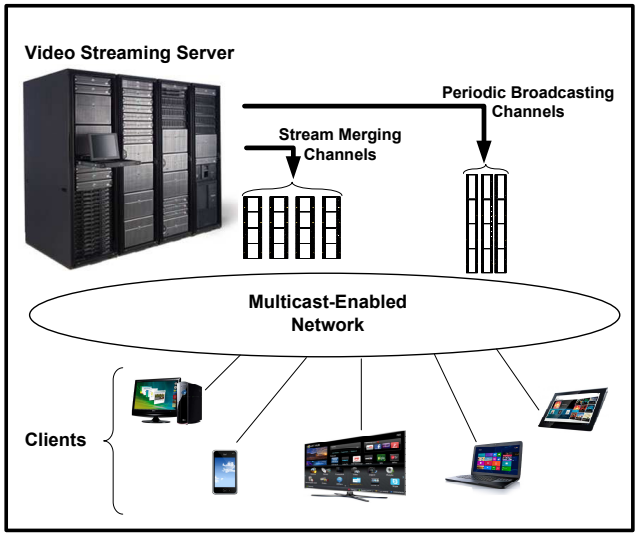


Fig. 2 System Overview

The main metrics in such system include revenue, profit, and client defection probability. The revenue and profit depend on the request arrival rate, defection probability, and the pricing of video contents. The profit additionally depends on the overall cost. The detailed quantitative analysis of revenue and profit based on various design and workload parameters is one of the unique contributions of this paper.

In our pricing model, the video advertisements are used to subsidize the cost and thus attract more clients. The price for streaming a video depends on the content royalty fee, the actual delivery cost, and the total ad viewing time by the client. The revenues generated from the ads are distributed to the clients proportionally to their total ad viewing times. Section 6 provides detailed information about the cost and pricing in addition to various system modeling aspects.

Our novel delivery solution combines the benefits of both stream merging and periodic broadcasting. A client starts by joining an ad broadcasting channel for some time and then receives the requested video by a stream merging technique. ("Video", "primary video", or "requested video", unless otherwise indicated, refers to one of the primary videos and not an ad.) When beginning to listen to an ad broadcasting channel, the client views different ads until the streaming of the requested video commences. The number of different ads or the total playback duration of unique ads should be large enough to ensure that the same ad is not viewed more than once by the same client when the system is heavily loaded. Periodic broadcasting is used for pushing the ads to the clients because the ads are potentially accessed more frequently than any individual primary video content, especially if each client is required to view a minimum number of ads. Thus, a specified number of server channels are allocated for the video ads, and the video ads are combined and broadcast periodically on these channels. The primary videos, however, are delivered on-demand using a scalable stream merging technique, such as ERMT. All server channels that are not assigned for periodic broadcasting are used for stream merging. The use of the multicast-based approach for delivering both the primary videos as well as the video ads leads to outstanding performance benefits.

In our delivery solution, the ads are viewed only prior to watching the primary videos. Uninterrupted viewing of the primary videos allows for a more enjoyable playback experience. This strategy is adopted by many video streaming systems, including YouTube. A client can join a broadcasting channel only at the beginning of an ad and can leave the channel only at the end of an ad, thereby partial viewing of any ad is not allowed. This feature, referred to here as *Only-Full*, is appealing to companies that seek to advertise their products and services. Thus, the client may experience a waiting time before beginning to listen to the ad broadcasting channel. Moreover, even when a server channel becomes available during the playback of an ad, the streaming of the primary video can begin only after the current ad has finished.

We propose an early snooping mechanism to reduce the delivery cost of stream merging. Since the primary videos are delivered using stream merging, up to two client download channels are required, each at the video playback rate. Thus, while listening to one ad broadcasting channel, the client uses the other channel for snooping on the closest preceding stream of the requested video so as to reduce the delivery cost for joining that stream. Clients will not snoop if there is no preceding stream for the requested video or if the time difference between the current time and the starting point of the closest preceding stream is beyond a certain threshold. This threshold controls the limit on the required buffer size, which is an important factor, especially for mobile devices.

Figure 3 illustrates the overall system operation and design.

The main problems we need to address are (i) how the ads can be allocated on the broadcasting channels in a way that reduces the average client waiting time before joining

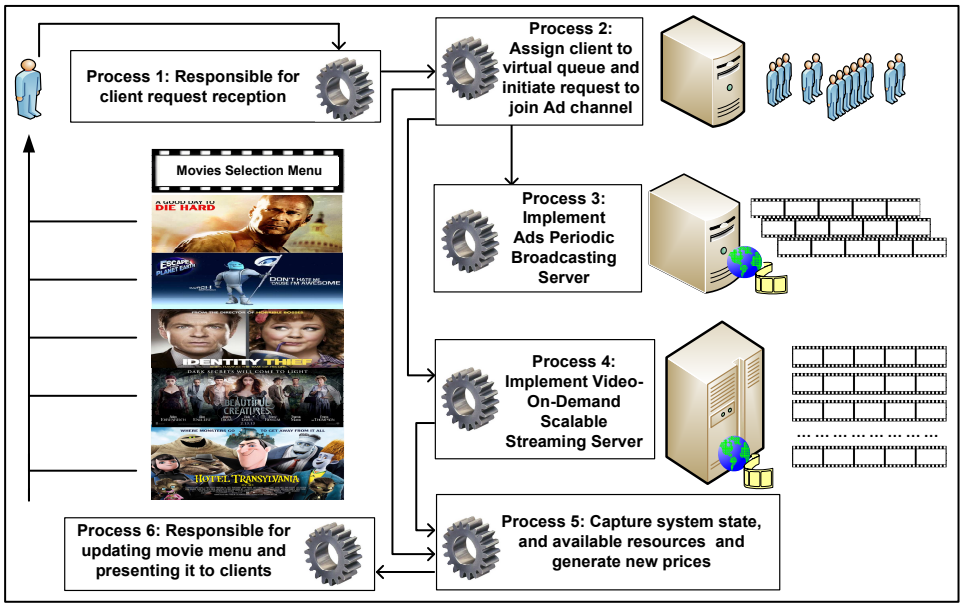


Fig. 3 An Illustration of the Overall System Design

an ad broadcasting channel and (ii) how the waiting requests for the primary videos can be efficiently scheduled for service. These issues are discussed in Section 4 and Section 5, respectively.

Table 2 describes the major notions used in the rest of the paper.

Table 2 Description of the Used Notions

δ	Time offset between two consecutive ad broadcasting channels (seconds)
Ad_{len}	Ad length (seconds)
N_{adCh}	Number of ad broadcasting channels
$AdCh$	Ad Channel
P_{brdcst}	Ad broadcasting period (seconds)
M_{ad}	Number of ads used in one ad broadcasting channel
G	Number of different ad groups
D	Number of ad channels in each ad group
λ	Request arrival rate (requests/second)
t_{Δ}	Time interval between successive ads on different ad channels (second)
t_{max}	Maximum client waiting time before joining an ad broadcasting channel (second)
t_{avg}	Average client waiting time before joining an ad broadcasting channel (second)
C	Number of possible combinations of allocating ads on broadcasting channels
$Delay$	Client waiting time before joining an ad channel (in the unit of time interval (t_{Δ}))
$Match$	Level of similarity between the client's interest and an ad group
θ	Skew parameter in Zipf-like distribution
$f_p(x)$	Pareto probability density function for purchasing capacity
b	Scale parameter of the Pareto probability density function
α	Shape parameter of the Pareto probability density function
w	Willingness probability
Δ	Elasticity parameter in the willingness function
p	Price

4 Allocation of Video Advertisements on Broadcasting Channels

We discuss next the ad allocation solution (which is one part of the overall delivery solution) and then present the proposed support for targeted ads.

4.1 Proposed Ad Allocation Solution

A central issue in the proposed delivery solution is how the ads can be allocated on the dedicated broadcasting channels. One of the main problems is that a client may experience a waiting time before beginning to listen to the ad broadcasting channel because it can join that channel only at the beginning of an ad and can leave it at the end of an ad. To mitigate this problem, we use multiple ad broadcasting channels with time-shifted versions of the combined ads, as shown in Figure 4. In this figure, four ads are broadcast on three dedicated broadcasting channels. As an example, if the ad length is 30 seconds, the maximum time for a new request to reach the beginning of an ad is $30/3 = 10$ seconds with three broadcasting channels compared to 30 seconds with only one broadcasting channel. The average initial waiting time is $10/2 = 5$ seconds with three broadcasting channels compared to $30/2 = 15$ seconds with only one broadcasting channel. As done in most prior studies, we assume throughout this paper a Poisson distribution for the request arrivals

[6,31,32]. Note that the system has an ad insertion point every $30/3 = 10$ seconds with three broadcasting channels, compared with 30 seconds when only one broadcasting channel is used. At each insertion point, multiple clients can join the same ad broadcasting channel.

The ad allocation problem arises because of using multiple ad broadcasting channels, each with multiple ad slots. The problem involves two aspects: (i) how to determine the offset from one ad channel to the next, and (ii) how to order the ads on each channel. The offset from one channel to the next, referred to here as *channel offset* (δ), is the time shift or skew between one ad channel and the next. The configuration in Figure 4 has two channel offsets: δ_1 and δ_2. These offsets are between Ad Channel 1 and Ad Channel 2 and between Ad Channel 2 and Ad Channel 3, respectively.

The ad allocation problem is simple in the case of uniform ad lengths. It is reduced to only how to determine the offset from one channel to the next. In general, the initial waiting time is simply the time to reach the beginning of the broadcast of the next closest ad. Thus, it depends on the time interval between the beginning of the broadcast of the latest ad and the beginning of the broadcast of the closest following ad, which is on the next ad broadcasting channel. In Figure 4, the time intervals are $t_{\Delta 1}$, $t_{\Delta 2}$, ..., $t_{\Delta 12}$. $t_{\Delta 1}$ and $t_{\Delta 2}$ are equal to δ_1 and δ_2, respectively. The maximum initial waiting time is equal to the length of the longest interval, whereas the average initial waiting time is the weighted sum of all intervals divided by 2. The weighted sum is required because the intervals are not of equal length. Note that longer intervals should have larger weights because more new requests are likely to be initiated during them. Because of the repeating nature of ads, the average can be found during the time of one ad broadcasting period (P_{brdcst}). In Figure 4, this period is the total length of the ads (Ad 1, Ad 2, Ad 3, and Ad 4). The optimal channel offset is basically the offset that leads to uniform intervals between successive ads and thus can be given by

$$\delta_{opt} = \frac{Ad_{len}}{N_{adCh}}, \tag{1}$$

where Ad_{len} is one ad length and N_{adCh} is number of ad broadcasting channels.

For variable-length ads, the problem is more challenging because the initial waiting time depends not only on the channel offset but also on the order of ads in each channels. Assuming a fixed order in all channels simplifies the problem but is generally inefficient because it may lead to large initial waiting times. Thus, a solution that finds the best order of ads in each channel is required. Because the best solution is the one that makes the intervals between successive ads as uniform as possible, the optimal channel offset can be found as follows:

$$\delta_{opt} = \frac{P_{brdcst}}{N_{adCh} \times M_{ad}}, \tag{2}$$

where P_{brdcst} is the ad broadcasting period, N_{adCh} is the number of ad broadcasting channels, and M_{ad} is number of ads used in one ad broadcasting channel. As discussed earlier, the average initial waiting time is the weighted sum of the intervals between successive ads within one broadcasting period. Assuming that $t_{\Delta i}$ is the i^{th} interval, the maximum and average initial waiting times can be determined as follows:

$$t_{max} = \max_{i}^{n} t_{\Delta i}, \tag{3}$$

and

$$t_{avg} = \sum_{i}^{n} [(t_{\Delta i}/2) \times \frac{\lambda \times t_{\Delta i}}{\sum_{j}^{n} \lambda \times t_{\Delta j}}] = \frac{\sum_{i}^{n} t_{\Delta i}^2}{2 \times \sum_{i}^{n} t_{\Delta i}}, \tag{4}$$

where t_{max} is the maximum initial waiting time, t_{avg} is the average initial waiting time, λ is the request arrival rate, and n is the number of intervals. Note that $\lambda \times t_{\Delta i}$ is the number of request arrivals within interval $t_{\Delta i}$. Because of the square of $t_{\Delta i}$ in the equation, reducing the maximum initial waiting time tends to reduce the average initial waiting time. The initial waiting time depends on the interval lengths, which in turn depends on the order of ads in each channel.

Let us now discuss how ads can be ordered in broadcasting channels. An optimal ad allocation minimizes the average initial waiting time t_{avg} but has a prohibitive time complexity. To find the minimum t_{avg}, a brute-force procedure can be used to search the domain of all possible combinations. The number of these combinations can be given by

$$C = (M_{ad}!)^{N_{adCh}}, \tag{5}$$

where N_{adCh} is the number of ad broadcasting channels, and M_{ad} is number of ads per channel. To achieve a low time complexity while providing close performance in terms of the initial waiting time, we propose a simple *heuristic ad ordering scheme*. This scheme works as follows for a maximum of five channels.

- On channel 1, place the longer ads closer to the ends.
- On channel 2 (if it is not the last channel), reverse the order of channel 1.
- On channel 3 (if it is not the last channel), place the second half of the ads of channel 1 first, followed by the ads of the first half of those on channel 1.
- On channel 4 (if it is not the last channel), reverse the first half of the ads of channel 1 and then reverse the second half and place them together.
- For the last channel, compute the best combination leading to the least initial waiting time given the prior ordering of the other channels.

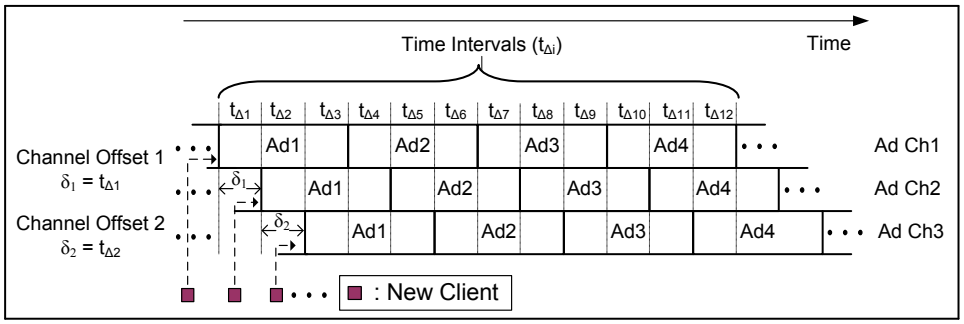


Fig. 4 Configuration of Ad Broadcasting Channels

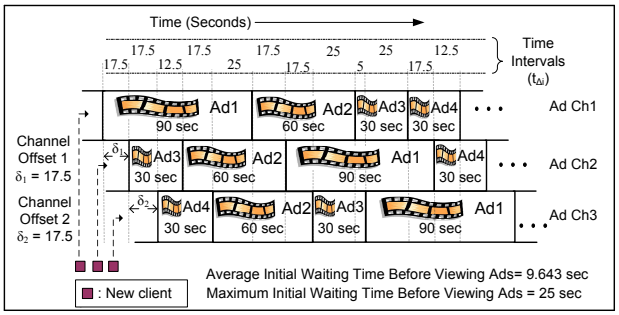


Fig. 5 Ad Allocation with Optimal Ad Ordering

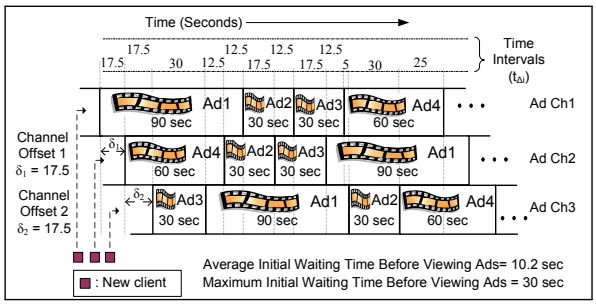


Fig. 6 Ad Allocation with Heuristic Ad Ordering

The scheme can be extended to a larger number of channels, but generally there is no need for the extra channels as the decrease in the initial waiting time would be too small with further channels. For example, for five ad channels with each channel streaming the same ad lengths such as 30 second, the initial waiting time will be $30/5 = 6$ seconds for six ad channels it will be $30/6 = 5$ seconds. Figures 5 and 6 show the ad allocation with optimal ad ordering and heuristic ad ordering, respectively for four specific ads and three channels. Note that in this example the average initial waiting time is only 6.2% longer than the optimal. The worst allocation (not shown) yields an average initial waiting time of approximately 19 seconds, which is about twice that of the optimal.

4.2 Support for Targeted Advertisements

Let us now discuss how *targeted advertisements* can be supported. With targeted advertisements, clients receive ads that are well suited to their interests. Since the support for targeted ads impacts ad allocation, we modify the proposed ad allocation solution to capture the interests of various clients. In particular, the ads are divided into various groups or categories. The classifications can be broad (such as sports, entertainment, and nutrition/food) or more specific (such as soccer, basketball, drama, horror, etc.). The interests of the client can be manually provided by the client or determined automatically by the system based on the history of the viewed contents or the group of the currently requested video. A client can join any ad category, but the advertising company pays in full only for the ads that perfectly match that client. Otherwise, a partial payment is assessed based on the level of similarity between the best matching group and the one to which the client is assigned. The similarity of ad group A to Ad group B can be defined as the likelihood of a client who is interested in ad group A to be interested in ad group B. For example, a client who has a main interest in cars may also be interested in sports, and a client who is interested in sports may also be interested in nutrition. Different ad groups exhibits varying levels of similarity. For example, soccer games and soccer products videos have higher similarity to one other than that of foods and nutrition. In this paper, we assume that the similarity levels between various ad groups are already known and properly quantified. Assessing the similarity is left for a future study.

We define a $G \times D$ ad configuration as the configuration that has G different ad groups, and each group has D ad broadcasting channels. The groups are numbered according to their similarity. Hence, groups with greater similarity have closer numbers. The system should try to map a new client to a specific ad channel that achieves the best initial delay ($Delay$) and interest match ($Match$) tradeoff. We propose three alternative ad allocation schemes:

- *No Group Interleaving with Multiple Ad Channels Per Group*,
- *Group Interleaving with Multiple Ad Channels Per Group* and
- *Group Interleaving with One Ad Channel per Group.*

A fourth option with no group interleaving and only one ad broadcasting channel per group is possible but is not analyzed here because of its clearly evident poor performance. Figure 7 illustrates these three alternatives. The system here supports three different ad groups (cars, sports, and food with nutrition). Figures 7(a) and 7(b) assume a 3×2 configuration, whereas Figure 7(c) has a 3×1 configuration. In the first alternative, various ad groups are not interleaved, but each group consists of interleaved ad broadcasting channels. Thus, different groups are aligned in time, but the ad channels of each group are time shifted. As discussed earlier, time shifting helps in reducing the initial delay (or waiting time) before viewing the first ad. This alternative can always achieve the best interest match with an average initial delay of $(Ad_{len}/D)/2$. In the second alternative, however, all ad broadcasting channels are interleaved (whether they belong to the same or different groups). In this alternative, an incoming request joins an ad channel that results in the best interest match and delay tradeoff. The delay is specified in the unit of time interval (t_{Δ}). Since the delay and interest match may have different ranges, they should be normalized (by dividing each with its largest value). Subsequently, the client will be admitted to the ad broadcasting channel achieving the best delay and interest match tradeoff. Hence, we need to find the ad channel number i^* that maximizes $Match(i)/Delay(i)$:

$$i^* = \arg \max_{i=1}^{N_{adCh}} \frac{Match(i)}{Delay(i)}, \tag{6}$$

where $Match(i)$ is the level of similarity between the client's interest and the ads in Ad Channel i, and $Delay(i)$ is the initial delay that client will experience if it joins Ad Channel i. Note that different weights can be assigned to $Delay$ and $Match$, but for simplicity we assume uniform weights. The average initial waiting time is $(Ad_{len}/(G \times D))/2$, assuming a uniform access distribution of the ad groups. The third alternative is the same as the second but has only one ad broadcasting channel per group and thus can be viewed as a special case. The average initial delay is given by $(Ad_{len}/G)/2$. Since this alternative uses fewer channels for ads, more server channels are utilized in streaming the requested video content.

A high level of ad targeting can be achieved by increasing the number of ad groups. Since large severs may have hundreds of channels, the number of ad groups can be increased easily without having any significant overhead. Furthermore, because the Group Interleaving with One Ad Channel per Group alternative performs the best as shown in Section 6.4, only one server channel is required for each ad group.

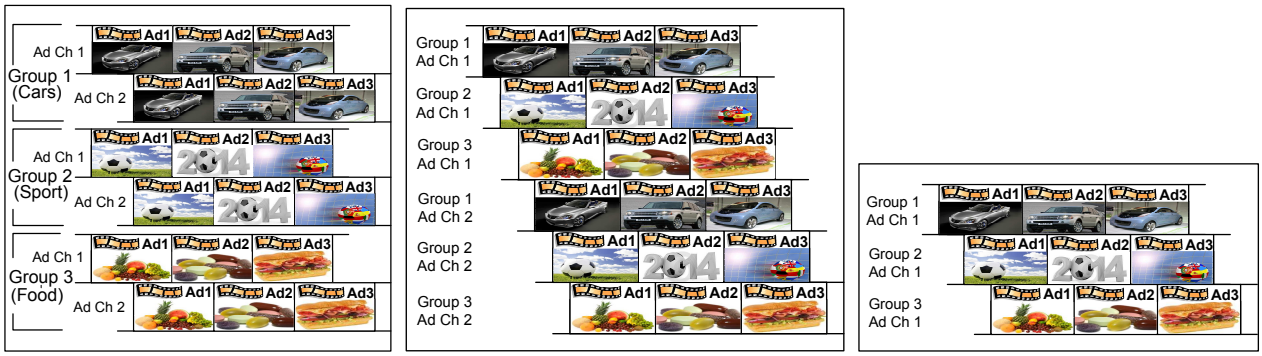
5 Proposed Constraint-Based Scheduling Policies

Let us now discuss the scheduling of the primary videos, which are serviced by stream merging. Existing scheduling policies, such as Maximum Queue Length (MQL) and Minimum Cost First (MCF), try to service requests as soon as possible and thus will not fit the video streaming system with video advertisements. The viewing of ads is significantly reduced when these policies are used.

We, therefore, propose a scheduling policy, called *Maximum Ads Time* (MAT), which schedules requests based on the ad viewing times. It selects for service the video with the longest total ad viewing time.

In addition, we present three modifications of MCF to ensure that ads are viewed by a large number of users: *Each N*, *Any N*, and *Majority 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, and Majority N considers a video for service only if 50% or more of the waiting requests have viewed at least N ads. Let us illustrate the differences among the polices when a waiting queue for a certain video has five requests. Any 2 qualifies these requests for scheduling only if any one of the requests has viewed at least 2 ads, whereas Each 2 qualifies them only if each one of them has viewed at least 2 ads, and Majority 2 qualifies them only if three or more of them have viewed at least 2 ads. Each N increases the ad viewing times but can cause servicing the most popular videos to be delayed for extended periods of time as new requests are made frequently. Majority N serves as a compromise between Each N and Any N. These policies determine the videos that are qualified for service and then the scheduling of these qualified videos is performed based on the MCF criterion. MQL can be used instead of MCF, but we choose only MCF because it is the best performer among all existing scheduling policies [26]. Since Each N, Any N, and Majority N consider the costs of servicing the video streams, they are expected to achieve higher degrees of resource sharing than MAT. Moreover, the value of N should be selected carefully. It can be dynamically assessed by the server using a probing-based mechanism, whereby the server tries different values of N over different periods of time and then adopts the value that yields the highest profit. We should mention that ads are assumed to have equal lengths and revenues.

Figure 8(a) illustrates the operation of Any 2 Scheduling by an example. In this example, the scalable delivery system periodically broadcasts three specific ads ($Ad1, Ad2, and Ad3$) on three dedicated broadcasting channels. In addition, the system has also four other channels for servicing the requested videos using stream merging. Incoming requests join the the next closest ad channels. Client



(a) No Group Interleaving with Multiple Ad Channels Per Group (b) Group Interleaving with Multiple Ad Channels Per Group (c) Group Interleaving with One Ad Channel Per Group

Fig. 7 Different Ad Allocation Alternatives for Supporting Targeted Ads

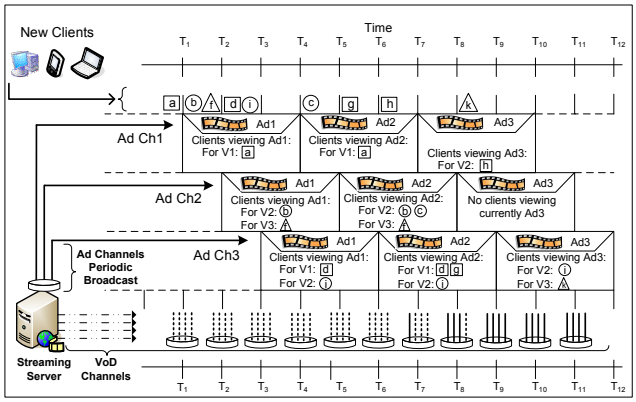
a is admitted to channel $AdCh1$ at time $T1$, while clients b and f are admitted to channel $AdCh2$ at time $T2$. Client c is also admitted to channel $AdCh2$ but at time $T5$. At time $T7$, client a for video $V1$ has already watched two ads, and thus it meets the qualification criterion of Any 2. The server has four available channels for stream merging at time $T7$, and thus video $V1$ is served at time $T7$. At time $T8$, both clients b for video $V2$ and f for video $V3$ have already watched two ads and are qualified for service. The server still has three available channels for stream merging, and thus both clients are served at time $T8$, leaving only one available channel for streaming merging. Since video $V2$ is qualified for service according to the Any 2 criterion, client c for video $V2$ is served with client b using the same server channel although it watched only one ad.

Figure 8(b) illustrates the operation of Each 2 Scheduling with a similar example. In this example, client a is admitted to channel $AdCh1$ at time $T1$, while clients b and f are admitted to channel $AdCh2$ at time $T2$. Client c is admitted to channel $AdCh2$ but at time $T5$. At time $T7$, client a for video $V1$ has already watched two ads, and thus it meets the qualification criterion of Each 2. The server has four available channels for stream merging at time $T7$, and thus video $V1$ is served at time $T7$. At time $T8$, both client b for video $V2$ and client f for video $V3$ have watched two ads but only client f is qualified for service because video $V2$ has another client (client c) who joined channel $AdCh2$ and has not yet watched two ads. The server still has three available channels for stream merging, and thus only client f is served at time $T8$, leaving two available channels for stream merging. Note that

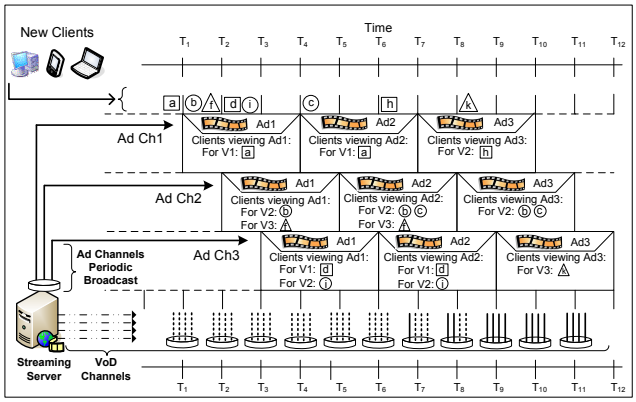
6 Performance Evaluation Methodology and Results Analysis

6.1 Simulation Platform

We have developed an event-driven simulator in C++ for a video streaming system implementing all aspects of the



(a) Any 2



(b) Each 2

Fig. 8 Illustrations of Any 2 and Each 2 Scheduling Policies [The system has four regular channels, three ad channels, and nine client requests (a-k)]

proposed solution. The simulator supports various stream merging techniques and scheduling policies. We have validated the simulator by reproducing several situations in previous studies [7,18,19,25,26,33]. The simulation stops after a steady state analysis with 95% confidence interval is reached.

The analyzed *performance metrics* are client defection probability, average number of ads viewed per client, average and maximum initial waiting time before viewing ads, price, channel utilization, arrival rate, delivery cost, profit, and revenue.

The effectiveness of the ad ordering schemes in terms of the initial waiting time is investigated by using another dedicated simulator, which we also developed in C++ for this purpose.

6.2 Workload Characteristics

Table 3 shows the workload characteristics used to evaluate the effectiveness of the proposed heuristic ad ordering scheme. Different configurations (i.e., combinations of the number of ad channels and the number of ads per channel) are examined. For each configuration, 750 runs are conducted. In each run, the ad lengths are chosen randomly in a weighted fashion to form a new combination. The used ads lengths are 15, 30, 45, and 60 seconds, each with a probability of 35%, 50%, 10%, and 5%, respectively. These probabilities are based on the popularities of ad lengths in practice [34]. For example, an ad length of 30 seconds is usually the most common.

Table 4 summarizes the workload characteristics used for all other parts of this study. Table 5 shows the additional workload characteristics when the proposed support of targeted ads is used. As discussed earlier, we assume that the arrival of the requests to the server follows a Poisson process with an average arrival rate λ. Moreover, we assume that the access to the primary videos (not the video ads) is highly localized and follows a Zipf-like distribution. The probability of choosing the n^{th} most popular video is $A/n^{1-\theta}$ with a parameter θ and a normalized constant A. The parameter θ controls the skew of video access. The skew reaches its peak when $\theta = 0$, and the access becomes uniformly distributed when $\theta = 1$. We analyze the impact of this parameter, but we generally assume a value of 0.271 [35–37]. In [38], it was shown that the Zipf-like distribution fits the video access better than the other distributions, such as the Stretched Exponential distribution. Furthermore, we characterize the waiting tolerance of clients in terms of the number of ads by a Poisson distribution [7,35].

The overall revenue is challenging to estimate, although it can be given simply as the product of the volume sold and the price. The volume here is the total number of streams delivered to the clients, which directly depends on the client defection probability. The complication happens because the price influences the arrival rate and number of streams delivered. Thus, subsidizing the price can attract more clients and can eventually increase the overall revenue. By increasing the arrival rate, the delivery costs also decrease because of the higher degrees of request aggregation and stream merging. Finding the profit also exhibits similar complications as the revenue. In this paper, we study the system under a dynamic arrival rate model. In this model, illustrated in Figure 9, the request arrival rate depends on the client purchasing capacity and willingness as well as client defection probability. To characterize purchasing capacity, we utilize economic theory, which suggests that the allocation of wealth is highly skewed and follows Pareto distribution. Similarly, the capacity of a client to spend for a particular service or product can be modeled using that distribution [14]. The Pareto probability density function can be given as follows: $f_p(x) = \alpha \times b \times x^{-(\alpha+1)}$ for $x \geq b$, where b (called *scale*) represents the minimum value of x, and α represents

Table 3 Summary of Workload Characteristics [Used for Only Heuristic Ad Ordering]

Parameter	Model/Value(s)
Number of Ads Channels	2, 3, 4
Number of Ads per Channel	2, 3, 4, 5, 6
Ad Lengths	15, 30, 45, 60 (sec)
Number of Iterations	750

Table 4 Summary of Workload Characteristics [General]

Parameter	Model/Value(s)
Request Arrival	Poisson Process
Request Arrival Rate	Variable, Default = 40 Requests/min
Server Capacity	200-750
Video Access	Zipf-Like, Skew Parameter $\theta = 0.271$
Number of Movies	120
Movie Length	120 min
Waiting Tolerance Model	Poisson, min = 3 ads, mean= 5 ads, max = 8 ads
Scale, Shape, Elasticity	$b : 1.0, \alpha : 1, \Delta : 7$
Ad Length	30 sec (uniform ad lengths case)
Number of Different Ads	8
Number of Ads Channels	Variable, Default = 3

Table 5 Summary of Workload Characteristics [Used for Only Targeted Ads]

Parameter	Model/Value(s)
Number of Ad Groups	5
Number of Ad Channels per Group	1 for Group Interleaving with One Ad Channel per Group 2 for the other configurations
Ad Group Configuration (# Groups x # Channels per group)	5x1 for Group Interleaving with One Ad Channel per Group 5x2 for the other configurations

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

$$w = \begin{cases} 1 - (\frac{p}{y})^{\Delta} & 0 \leq p \leq y, \\ 0 & p > y, \end{cases} \quad (7)$$

where Δ is the *elasticity* [14]. As Δ increases, more clients are willing to spend.

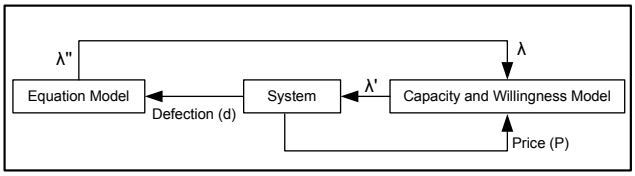


Fig. 9 Arrival Rate Dependency Flowchart, λ and Price(P) are Initialized at the System Start

The practical importance of the impact of the defection probability on the arrival rate is due to the fact that decreasing the price will not always, in the long run, continue to increase the arrival rate without adequate scaling of system capacity (upload bandwidth). We also assume that the download bandwidth at the client side is always available to accommodate their requests. The defection probability is an important Quality of Service (QoS) metric because clients who leave due to excessive delay will not likely return in the future. Note that the waiting time is already factored in the defection probability. We use the following model to capture the impact of defection probability:

$$\lambda = \frac{(1-d)}{c_1 + c_2 d^2}, \quad (8)$$

where d is the defection probability, and c_1 and c_2 are constants. Our analysis reveals that the overall behavior does not change with c_1 and c_2. The default values are 0.5 and 1, respectively.

6.3 Considered System

We consider here a commercial *Movie-on-Demand* system with 120 titles, each of which is 120-minute long with a constant bit rate. Stream merging is done using ERMT. Without loss of generality, we assume here a *cost-plus* model for the price. The price covers the movie royalty fee, delivery fee, and operational cost minus subsidization credit. All revenues from the ads are distributed to the clients proportionally to their total ad viewing times. The revenue per ad per user is 10 cents in the regular solution. With targeted ads, however, it is 10 cents plus a fraction of 5 cents proportional to the level of similarity between the group supplied and the best matching group, with $(10+0)$ cents for the worst match and $(10+5)$ cents for the best match. The movie royalty fee is 70 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 [39].

6.4 Main Results

Next, we discuss and analyze the main results. Since constraint-based scheduling has a big impact on the overall solution, we will start by analyzing various scheduling policies, assuming uniform ad lengths and no target advertisements. Subsequently, we analyze and compare the effectiveness of different ad ordering schemes when ads have varying lengths. Finally, we analyze the proposed support for targeted ads and compare various ad allocation alternatives.

6.4.1 Comparing the Effectiveness of Constraint-Based Scheduling Policies

Let us now compare the effectiveness of the proposed constraint-based scheduling policies. We generally assume, unless otherwise indicated, a total of 3 ad channels and 30-second ad lengths.

Figure 10 plots the defection rate versus the average number of viewed ads for various policies. The curve for each policy is generated by varying the server capacity. Since stream merging is primarily impacted by request scheduling, only the number of server channels used for stream merging is varied while fixing the number of ad broadcasting channels at 3. Obviously, higher server capacities produce lower defection rates. Each 3 leads to high defection rates and thus would be the least favorable. In general, Each N increases the ad viewing times but can cause servicing the most popular videos to be delayed for extended periods of time, especially for larger values of N. This behavior is because the requests for popular videos arrive at high rates, and each request for the video must view at least N ads for that video to be qualified for service. In contrast, Any N has a less restrictive qualification criterion, and Majority N serves as a compromise between Each N and Any N. Majority 2 performs close to Any 2 because the numbers of requests in the waiting queues are generally small and their average is close to 2. MAT has a wide range of operation and tends to increase the defection rate only moderately with the average number of viewed ads. It can be controlled, however, only by changing the server capacity because it does

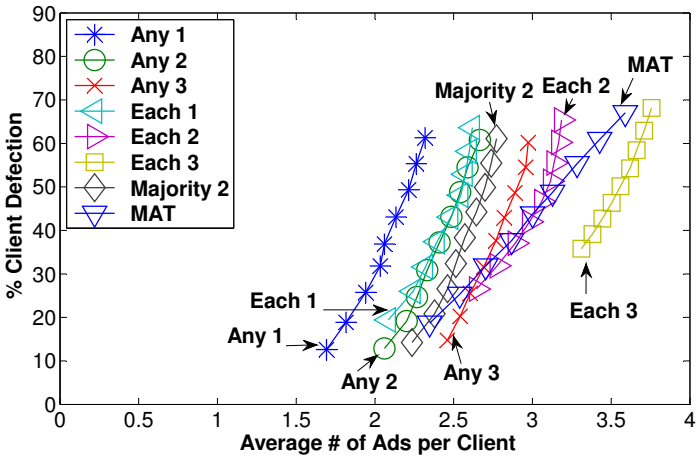


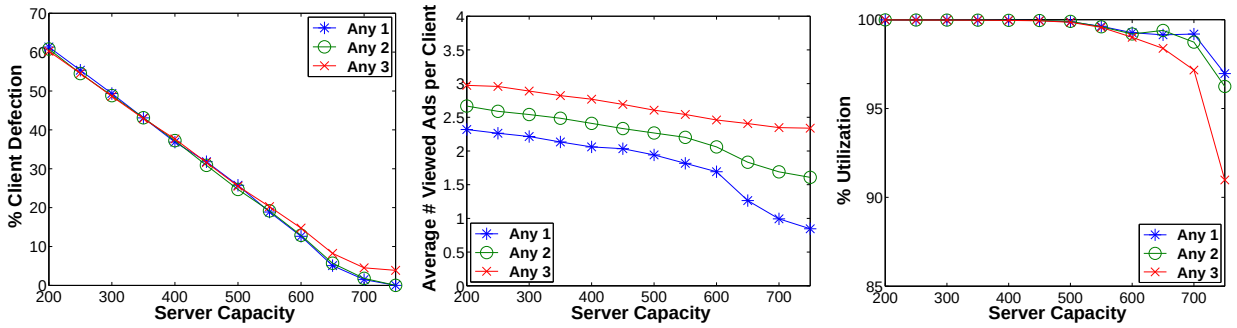
Fig. 10 Comparing Effectiveness of Various Scheduling Policies

not have a constraint on the minimum number of viewed ads.

Before conducing detailed comparisons of various scheduling policies, let us first analyze the impact of the constraint on the minimum number of viewed ads (N), and then use the best value in further comparisons. Figures 11 and 12 illustrate the impacts of N in Any N and Each N, respectively, on the defection probability, average number of viewed ads, and utilization of server channels. As expected, the ad viewing times increase with N at the expense of exceeding the client waiting tolerance for the requested video and thus increasing the defection rate. In addition, the utilization of server channels decreases slightly with N, and the variation in utilization is much more significant in the case of Each N because of its tighter constraint on the ad viewing times. A value of 2 leads to a balanced impact.

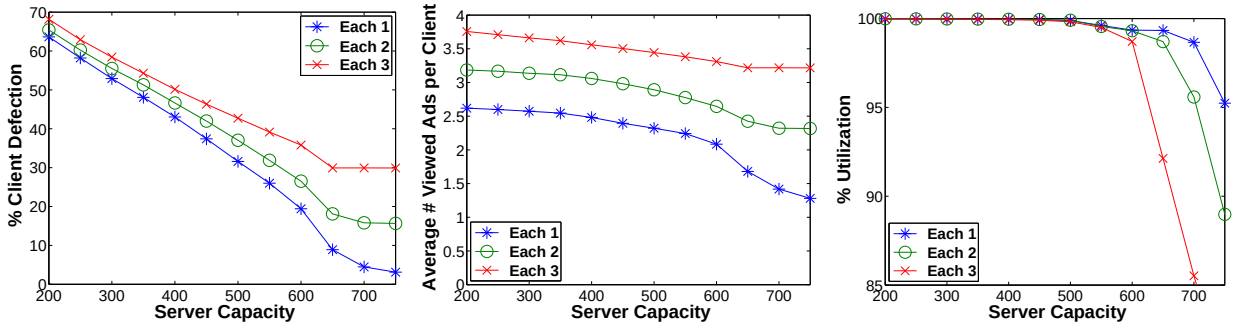
Based on the prior results, let us now compare Any 2, Each 2, Majority 2, and MAT in detail, considering eight performance metrics: client defection percentage, average number of viewed ads, utilization, delivery cost, price, arrival rate, revenue, and profit. The results are shown in Figure 13. Any 2 achieves the best overall performance in terms of the defection rate, revenue, and profit. It leads to the highest prices because it requires the shortest ad viewing times. Despite that higher prices discourage some clients from purchasing the service, Any 2 yields the lowest defection rates because of reducing the probability of exceeding the waiting tolerance of clients. The profit and revenue are also the highest with Any 2 because they depend on the product of the effective arrival rate and the defection rate. Due to the their longer ad viewing times, Each 2, Majority 2, and MAT lead to lower prices than Any 2, as shown in Figure 13(e). The increase in price with server capacity is due to the lower number of ads being watched by the clients. Administrators and business owners may choose to increase the price, or keep it the same and have lower increase in profit. In this case the profit increase is generated mainly from cost reduction. In conclusion, Figure 13 shows the operation curves for the system. Administrator or business owner may choose from different operation modes depending on the metrics' values that suite their target. Interestingly, the average delivery cost decreases with the server capacity in all policies despite of the increase in system resources. This behavior is attributed to the increase in the effective arrival rate as more clients will be aggregated and serviced by the system using these shared resources. As expected, all policies achieve almost full utilization of server channels except at high server capacities. When resources become abundant, some of the server channels may not be fully utilized. We experimented with different values of c_1 and c_2 in Equation (8), but the relative behavior remains unchanged. These results are not shown to reduce the number of figures.

Finally, let us discuss the impact of the number of ad broadcasting channels. Figure 14 shows the impact of the number of ad channels when the server capacity is fixed at 500 in terms of the three main metrics: client defection percentage, the average number of viewed ads, and profit. As we increase the number of ad channels, the initial client waiting time before joining an ad channel decreases (as discussed in Section 4). In addition, the ad viewing times increase as the number of stream merging channels (which are dedicated to servicing the requested videos) decrease. The behavior in terms of the client defection rate is not as straightforward. In all policies except for Each N, the defection rate increases with the number of ad channels. This increase is due primarily to two factors: (i) the reduced level of request aggregation and (ii) the reduced number of stream merging channels. The degree of request aggregation decreases with the number of ad channels because the requests for each primary video will be divided among the waiting



(a) Client Defection Rate (b) Average Number of Viewed Ads (c) Utilization

Fig. 11 Impact of the Constraint on the Minimum Number of Ads (N) in Any N Scheduling



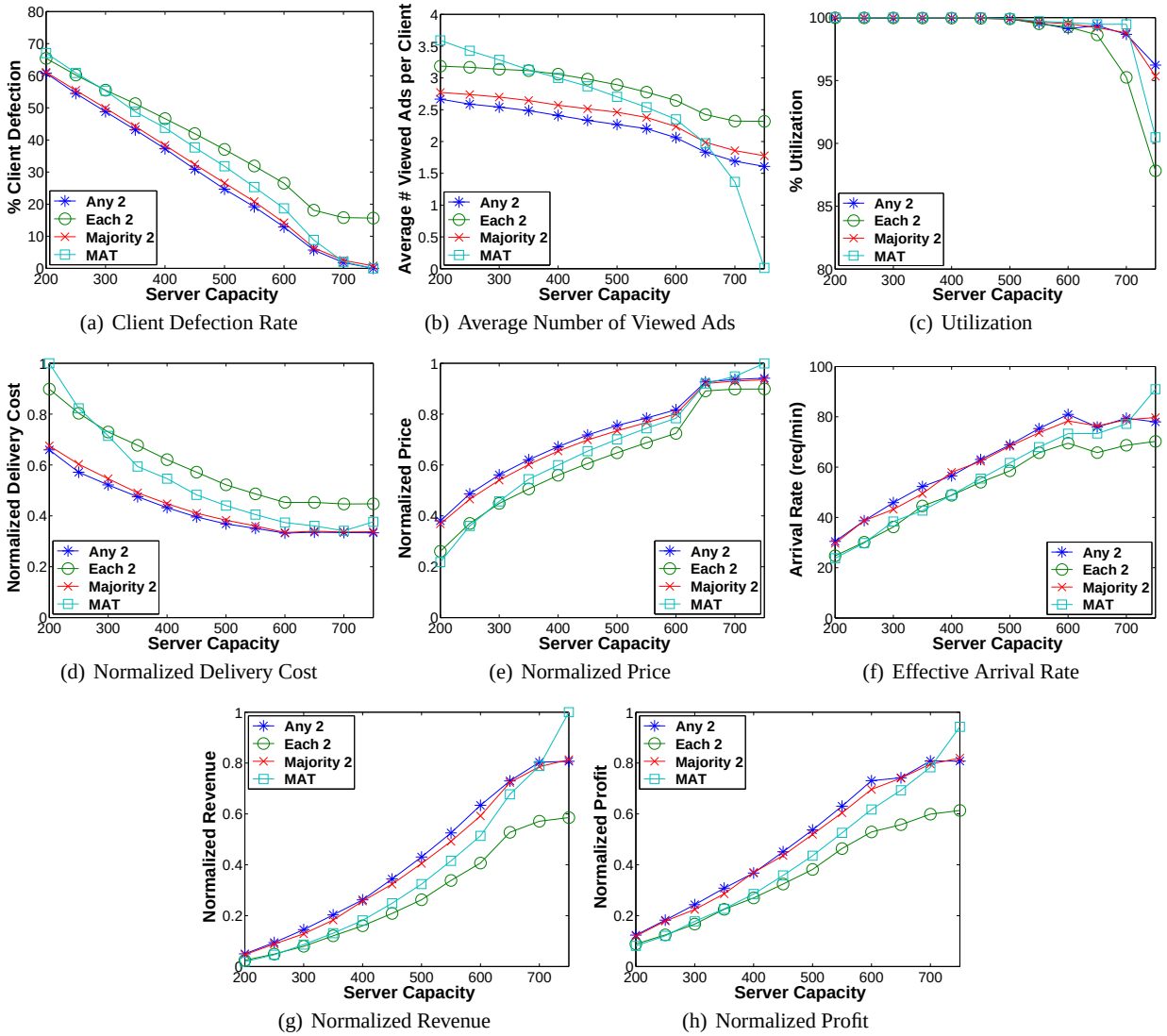
(a) Client Defection Rate (b) Average Number of Viewed Ads (c) Utilization

Fig. 12 Impact of the Constraint on the Minimum Number of Ads (N) in Each N Scheduling

queues on more ad channels, thereby reducing the number of requests that can be serviced at any time using the same stream merging channel. Although these factors are still applicable in Each N, its qualification criterion plays a more dominate role, pushing the defection rate in the opposite direction. As discussed earlier, Each N qualifies a video for service only if all its waiting requests have viewed at least N ads. Thus, any new request for a video causes the waiting requests for that video to wait longer until the new request becomes qualified. Therefore, as we increase the number of ad channels, more different waiting queues will be available for the same primary video, and thus the probability that a new request joins a specific waiting queue of any specific video becomes lower. Note that separate waiting queues are maintained for each primary video on different ad channels. These results indicate that Any 2 is still the best performer. Moreover, with Any 2, it is best to keep the number of ad channels as small as possible in order to increase profit and decrease defection rate. The initial waiting time before viewing the first ad, however, is a significant human-related metric. Using 3 ad channels serves as a good compromise. With 3 ad channels, the average waiting time before viewing the first ad is 5 seconds, assuming 30-second ads.

6.4.2 Comparing the Effectiveness of Ad Ordering Schemes

For variable ad lengths, let us now compare the effectiveness of the optimal and heuristic ad ordering schemes. As discussed earlier, the optimal scheme searches the domain of all possible combinations. For comparative purposes, we also analyze a random scheme that orders the ads on the broadcasting channels in a random fashion. Figure 15(a) compares the three schemes with various configurations (i.e., combinations of the number of ad channels and the number of ads per channel) in terms of the average initial waiting time before viewing the first ad. The figure indicates that the heuristic ad ordering performs close to the optimal. On average, it differs in only 5.9% from the optimal scheme, whereas the random differs in 15.8% from the optimal. Figure 15(b) shows the results of the maximum initial waiting time before starting to view ads. In this case, the heuristic scheme differs in 22% from the optimal, whereas the random has a difference of 52% from the optimal. Although the optimal ordering achieves the least initial waiting times, it has prohibitive time complexity. The heuristic schemes yields near optimal average waiting times while drastically reducing the time complexity.

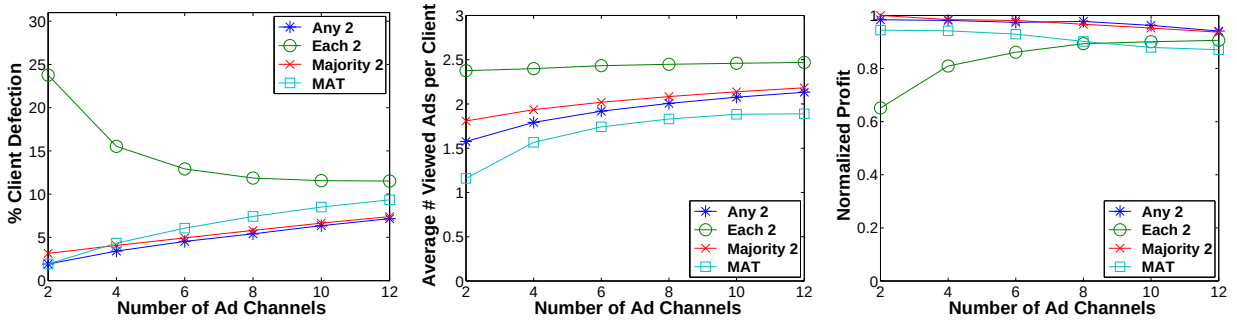


(a) Client Defection Rate (b) Average Number of Viewed Ads (c) Utilization

(d) Normalized Delivery Cost (e) Normalized Price (f) Effective Arrival Rate

(g) Normalized Revenue (h) Normalized Profit

Fig. 13 Comparing Effectiveness of Various Scheduling Policies

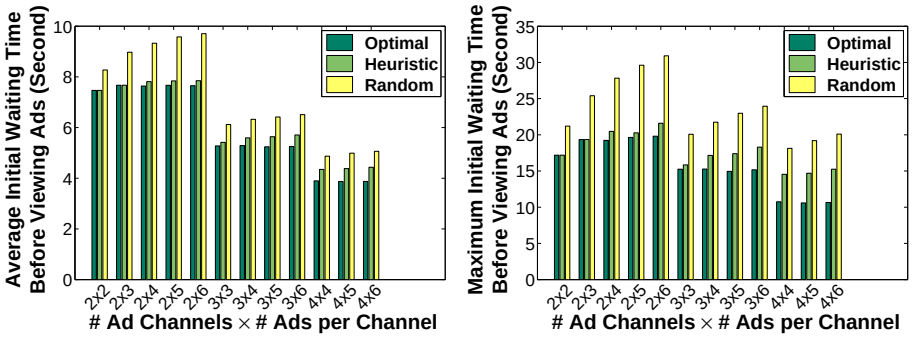


(a) Client Defection Rate (b) Average Number of Viewed Ads (c) Normalized Profit

Fig. 14 Impact of the Number of Ad Channels [Server Capacity = 700]

6.4.3 Comparing the Effectiveness of Various Alternatives for Supporting Targeted Ads

Finally, let us investigate the three proposed ad allocation alternatives for supporting targeted ads: *No Group Interleaving with Multiple Ad Channels Per Group*, *Group Interleaving with Multiple Ad Channels per Group*, and *Group Interleaving with One Ad Channel per Group*. Figure 16 compares the effectiveness of the three alternatives in terms of



(a) Average Waiting Time (b) Maximum Waiting Time

Fig. 15 Comparing the Effectiveness of Ad Ordering Schemes in the Initial Waiting Time before Viewing the Ads

the client defection percentage, average initial waiting time before viewing ads, and profit. Only the results with Any 2 are shown because it achieves the best performance. The first two alternatives have a 5×2 configuration (i.e., 5 different groups each with 2 ad channels). The third alternative still has 5 groups, but each has only one ad channel. The results show that the third alternative performs the best in terms of defection percent and profit. This behavior can be attributed to having the smallest number of ad channels and thus the largest number of channels that are dedicated to stream merging. Its average initial waiting time is between the other two alternatives but closer to the one that achieves the least initial waiting time (Second Alternative). It is the best overall performer despite that it leads to the least ad viewing time and thus the highest prices (not shown). Although higher prices discourage some clients from purchasing the service, this alternative yields the least defections rates by allocating more channels for stream merging.

Figure 17 shows the effectiveness of using targeted ads. Targeted ads increase the overall request arrival rates by increasing the price subsidization. Obviously, the defection rate increases with the number of requests as the server capacity remains unchanged. Hence, Figure 17(c) shows the profit versus the defection rate (instead of server capacity) to provide fair comparisons. These results demonstrate that targeted ads yield higher profits. They also indicate that the profit decreases with the defection rate and thus the system should be sized appropriately (in terms of the number of server channels).

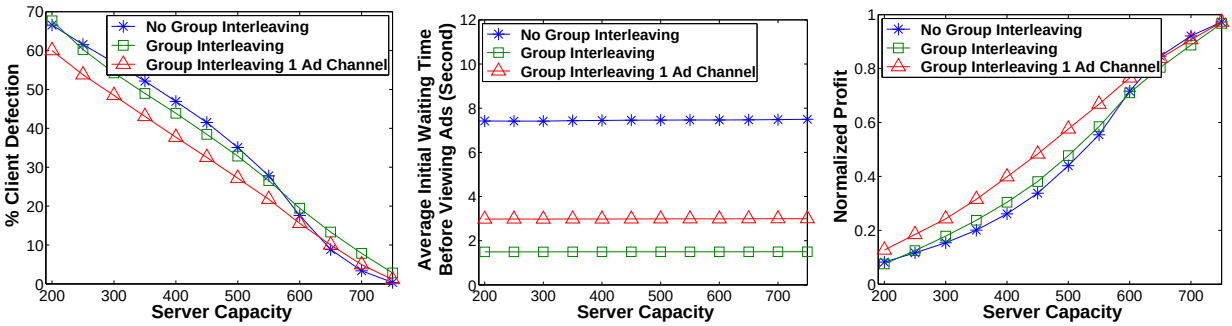
7 Conclusions and Future Work

We have presented a scalable delivery solution and a pricing model for commercial near VOD systems with video ads. The delivery solution combines the benefits of stream merging and periodic broadcasting. The overall solution includes an ad allocation solution that allocates ads efficiently on ad channels so as to reduce the initial waiting time before receiving the first ad. It also includes a new constraint-based request scheduling approach of the primary videos. We have presented four scheduling policies. For ad allocation, we have presented two schemes for ordering the ads on the broadcasting channels: optimal and heuristic. In addition, we have presented three alternatives for supporting targeted ads. In the overall solution, the revenues generated from the ads are used to subsidize the price, thereby attracting more clients and increasing the overall revenue. Pricing depends on the experienced QoS, specifically the total waiting before watching the desired media content. In particular, clients with longer ad viewing times get lower prices.

We have studied, through simulation, the effectiveness of the overall solutions and analyzed and compared the effectiveness of various scheduling policies, ad ordering schemes, and alternative methods for supporting targeted ads. We have considered numerous performance metrics and have captured the impacts of client purchasing capacity and willingness models as well as client defection probability on the effective request arrival rate.

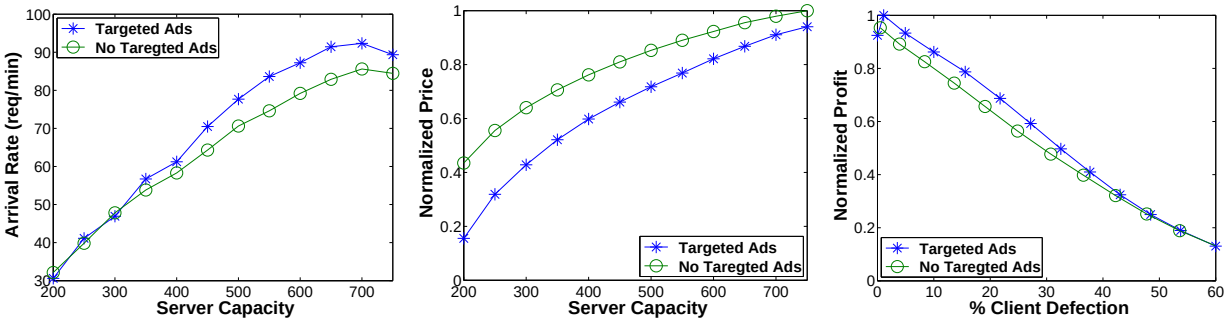
The main results can be summarized as follows.

- By being moderately restrictive in the ad viewing times, Any N achieves the best overall performance in terms of the client defection rate, profit, and revenue. Moreover, it can control the average ad viewing time by adjusting N. The best value of N in the studied workload is 2. Increasing N forces the clients to view more ads but increases the defection rate and decreases system utilization.
- The number of ad channels should be as small as possible so as not to hurt performance in terms of the defection rate, revenue, and profit. Increasing the number of ad channels, however, reduces the waiting time before viewing the first ad, which improves the experienced QoS. In our studied workload with 30-second ads, using three ad channels provides the best compromise.
- When the ads are of varying lengths, ordering the ads on broadcasting channels has a strong impact on the initial



(a) Client Defection Rate (b) Average Waiting Time Before Viewing Ads (c) Normalized Profit

Fig. 16 Comparing the Effectiveness of Ad Targeting Alternatives [Any 2]



(a) Arrival Rate (b) Normalized Price (c) Normalized Profit

Fig. 17 Impact of Ad Targeting [Any 2]

waiting time before viewing the first ad. The proposed heuristic allocation scheme provides near optimal results while drastically reducing the implementation time complexity.

– Supporting targeted ads improves revenue and profit, with ad allocation having a significant impact on performance. Ad allocation is best to be performed by interleaving the channels of various ad groups/categories while having only one ad channel per group.

In future work, we plan to study how the similarities among various ad groups/categories can be assessed. Future work also includes providing expected times for viewing ads. As the clients need to know the price before purchasing the service, the system should estimate the expected viewing time of ads for each request dynamically based on the system's current state and the requested video. This estimation can be achieved by using waiting-time prediction [7, 40, 41]. A comprehensive solution, utilizing prediction and including enhanced support for targeted ads, is required.

References

1. M. Al-Hadrusi, N.J. Sarhan, in *Proc. of SPIE/ACM Multimedia Computing and Networking Conference (MMCN)* (2008), pp. 68,180G–68,180G
2. D.L. Eager, M.K. Vernon, J. Zahorjan, IEEE Trans. on Knowledge and Data Engineering **13**(5), 742 (2001)
3. H. Ma, K.G. Shin, Computer Communication Review **32**(1), 31 (2002)
4. N. Carlsson, D.L. Eager, M.K. Vernon, Perform. Eval. **63**(9-10), 864 (2006)
5. M. Rocha, M. Maia, I. Cunha, J. Almeida, S. Campos, in *Proc. of ACM Multimedia* (2005), pp. 966–975
6. H. Ma, G.K. Shin, W. Wu, Multimedia Tools Appl. **26**(1), 101 (2005)
7. N.J. Sarhan, M.A. Alsmirat, M. Al-Hadrusi, ACM Transactions on Multimedia Computing, Communications, and Applications (ACM TOMCCAP) **6**(2), 1 (2010)
8. K.A. Hua, S. Sheu, in *Proc. of ACM Special Interest Group on Data Communication (SIGCOMM)* (1997), pp. 89–100
9. L. Juhn, L. Tseng, IEEE Trans. on Broadcasting **43**(3), 268 (1997)
10. J.F. Pâris, S.W. Carter, D.D.E. Long, in *Proc. of the Int'l Symp. on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)* (1998), pp. 127–132
11. C. Huang, R. Janakiraman, L. Xu, in *Proc. of ACM Multimedia* (2004), pp. 152–159
12. P. Gill, L. Shi, A. Mahanti, Z. Li, D. Eager, ACM Transactions on Multimedia Computing, Communications, and Applications (ACM TOMCCAP) **5**(1), 1 (2008)
13. X. Cheng, C. Dale, J. Liu, in *in Proc. of IEEE 16th International Workshop on Quality of Service (IWQoS)* (2008)
14. S. Jagannathan, K.C. Almeroth, in *Proc. of the IFIP/IEEE International Conference on Management of Multimedia Networks and Services* (2001), pp. 329–344
15. G. Pallis, A. Vakali, Communications of the ACM **49**, 101 (2006)
16. C. Wu, B. Li, S. Zhao, S. Zhao, in *INFOCOM* (2009), pp. 2731–2735

17. V. Aggarwal, R. Caldebank, V. Gopalakrishnan, R. Jana, K. Ramakrishnan, F. Yu, in *Proc. of ACM Multimedia* (2009), pp. 421–430
18. D.L. Eager, M.K. Vernon, J. Zahorjan, in *Proc. of ACM Multimedia* (1999), pp. 199–202
19. K.A. Hua, Y. Cai, S. Sheu, in *Proc. of ACM Multimedia* (1998), pp. 191–200
20. J.P. O'Neill, J. Dukes, J. Dukes, in *INTENSIVE* (2009), pp. 39–46
21. H. Schulzrinne, A. Rao, R. Lanphier, R. Lanphier, in *Internet Engineering Task Force (IETF), RFC2326* (1998)
22. S. Deering, W. Fenner, B. Haberman, B. Haberman, in *Internet Engineering Task Force (IETF), RFC2710* (1999)
23. L. Gao, J. Kurose, D. Towsley, in *Proc. of the Int'l Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)* (1998)
24. W.D. Chien, Y.S. Yeh, J.S. Wang, J.S. Wang, in *IEEE International Conference on Multimedia and Expo (ICME)* (2004), pp. 1843–1846
25. A. Dan, D. Sitaram, P. Shahabuddin, in *Proc. of ACM Multimedia* (1994), pp. 391–398
26. N.J. Sarhan, B. Qudah, in *Proc. of Multimedia Computing and Networking Conf. (MMCN)* (2007), pp. 327–334
27. D. Rayburn, *Streaming and Digital Media: Understanding the Business and Technology* (Focal Press, 2007)
28. P. Basu, A. Narayanan, W. Ke, T.D.C. Little, A. Bestavros, in *Proc. of International Conference on Computer Communications and Networks* (1999), pp. 104–109
29. P. Basu, T.D.C. Little, in *Proc. of ACM Multimedia* (2000), pp. 359–361
30. T. Mei, X.S. Hua, S. Li, IEEE Trans. Cir. and Sys. for Video Technol. **19**(12), 1866 (2009)
31. C.K.d.S. Rodrigues, R.M.M. Leão, Comput. Netw. **51**(3), 569 (2007)
32. Z. Ge, P. Ji, P. Shenoy, Multimedia Systems **13**, 235 (2007)
33. Y. Cai, K.A. Hua, in *Proc. of ACM Multimedia* (1999), pp. 211–214
34. R.E. Bruner, J. Singh. Video ad benchmarks: Average campaign performance metrics. Online White Paper (2007). URL `http://www.doubleclick.com/insight/pdfs/dc_videobenchmarks_0702.pdf`
35. A.K. Tsiolis, M.K. Vernon., in *Proc. of ACM Special Interest Group for the computer systems performance evaluation community (SIGMETRICS)* (1997), pp. 285–297
36. N.J. Sarhan, C.R. Das, in *Proc. of the 7th IFIP/IEEE Int'l Conf. on Management of Multimedia Networks and Services* (2004), pp. 127–139
37. F. Thouin, M. Coates, M. Coates, (2007), vol. 21, pp. 42–48
38. J.R. Ostrowski, N.J. Sarhan, in *Proc. of SPIE/ACM Multimedia Computing and Networking Conference (MMCN)* (2009)
39. D. Rayburn. CDN pricing data: What the CDNs are actually charging for delivery. Online Article (2007). URL `http://blog.streamingmedia.com/the_business_of_online_vi/2007/08/cdn-pricing-dat-2.html`
40. N.J. Sarhan, M.S. Al-Hadrusi, in *Proc. of IEEE International Symposium on Multimedia (ISM)* (2010)
41. M.S. Al-Hadrusi, N.J. Sarhan, in *Proc. of IEEE the International MultiMedia Modeling Conference (MMM)* (2012)