1-1-2013

Design And Analysis Of Scalable Video Streaming Systems

Musab S. Al-Hadrusi
Wayne State University,

DESIGN AND ANALYSIS OF SCALABLE VIDEO STREAMING SYSTEMS

by:

MUSAB SALEM AL-HADRUSI

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2013

MAJOR: COMPUTER ENGINEERING

Approved By:

_____

Advisor Date

_____

_____

_____

DEDICATION

To my great parents (Salem & Najah), my late mother (Hasna) and grandmother (Zahia)

To my beloved wife Fatema

To my little ones Salem and Aws

ACKNOWLEDGMENTS

This research could not have been possible without the support of many people. I would like to show my sincere appreciation for my adviser Dr. Nabil Sarhan. His guidance and directions were of extreme help during my research. He always walked me through the extra mile and shared with me his knowledge and experience. I also would like to thank my committee members Dr. Cheng-Zhong Xu, Dr. Song Jiang, and Dr. Hongwei Zhang for their precious feedback.

I would like to extend my heartfelt gratitude to my parents who kept encouraging and supporting me on this track. Most especially to my wife Fatema for her patience and support, and to my kids Salem and Aws who never understood why I used to spend long time on my computer. Finally, to my brothers, sisters and all my friends who were around me during this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

Recent advances in video capturing, processing, storage, and delivery technologies have driven the evolution of multimedia streaming systems dramatically. Video streaming systems can be classified into two main categories based on the source of the streamed video, *Video-on-Demand* (VOD) and *Live Video Streaming*. In VOD, a prerecorded media file is played from a storage device, and the client has the ability to view the video at any time. An example of a VOD application is YouTube which is currently ranked as the third most popular website worldwide [1]. YouTube has about 48 hours of uploaded videos every minute, with over 3 billion videos viewed by more than 26.7 million different users daily [2]. In contrast to VOD, live video streaming offers simultaneous streaming and capturing of the video. Common applications include streaming of news, sports, live events, video communication and conferencing, and video surveillance. Many popular news networks and sport channels broadcast live events over the Internet. For example, Super Bowl was streamed online for the first time in 2011, with telecasts totaling 21 million and an advertisement revenue of $200 million [3]. Video surveillance is another application in which videos are streamed live. Video surveillance systems may have one or more networked cameras. World largest surveillance system is installed in Britain to monitor major cities. It has more than 4.2 million Closed Circuit Television System (CCTV) cameras, one for every 14 people [4].

1.2 Overview

Despite advancement in multimedia streaming technology, many multimedia applications are still facing major issues, including provision of Quality-of-Service (QoS), system scalability, limited resources, and cost. QoS differs according to the system and its goals. System scalability with respect to

a certain factor is an indicator of the system's ability to keep or to gracefully degrade its performance to a certain threshold at different scale values of that factor. The problem of limited resources happens when the number or the size of tasks exceeds the available resources. However, addressing these issues and enhancing the performance without affecting the cost remain a major hindrance.

Due to the strong correlation, addressing one problem may lead to a positive or negative impact on another problem. Specific metrics must be (introduced and) defined before proposing any solution, so that the effectiveness of these solutions can be assessed. Metrics are either related to the system performance or to the clients receiving the service. Different systems may have different metrics. For example, in interactive video streaming systems, video and sound qualities during a session are important metrics, but from the system point of view, the number of concurrently supported clients is more important. In VOD systems, the client waiting time for receiving a service is an important metric, but from a system perspective, the client defection (i.e., turn-away) probability is more important.

In this dissertation, we address the aforementioned challenges. Particularly, we develop and analyze a new set of metrics based on two video streaming systems. In addition, we design video streaming systems with improved performance according to these metrics. We focus primarily on two video streaming systems: VOD with video advertisements and Automated Video Surveillance (AVS).

VOD systems face the same previous challenges. Particularly, (1) high demand for QoS, which can be quantified by some metrics, including client waiting time and service price, (2) system scalability, defined as the system ability to maintain a certain level of performance at different request arrival rates, (3) limited resources, and (4) keeping the service cost at minimum while addressing all these issues.

AVS has been used for many applications, such as monitoring, detection, tracking, and recognition [5, 6, 7, 8, 9, 10]. AVS systems face the same challenges as in VOD. However, scalability here has a different definition according to the surveillance objective. In monitoring applications, where the area coverage is the main issue, scalability refers to the ability to cover wider areas with the same number of cameras. In tracking and recognition applications, the main issue is related to the subjects. In this

case, scalability refers to the ability to deal with more subjects using the same number of cameras while keeping the same level of performance. In an AVS system that is designed for identifying subjects, a problem arises when the number of subjects exceeds the number of cameras. Capturing all subjects before leaving the scene becomes hard, and managing the system to get the best performance becomes necessary. Moreover, in recognition systems, identifying subjects by face recognition is sensitive to the image quality. Additionally, in a dynamic scene, predicting the best time to capture a subject frame (image) with a good quality is a hard task. Finally, providing successful solutions to these problems at low cost remains as a main challenge. While video streaming is an important aspect of AVS system, we will consider another tightly related aspect, namely the management of cameras in AVS.

By managing the cameras in AVS, we can (1) capture images with the best quality, thereby maximizing the subject recognition probability with the minimum number of frames, (2) group more than one subject in a single frame to reduce the number of captures, (3) and reject any captures that do not contain recognizable faces by predicting the quality of the image.

The main objectives of this dissertation can be summarized as follows.

(1) To develop a scalable delivery framework for VOD with video advertisements. This framework includes an accurate prediction algorithm and an envisioned pricing model. We consider the following performance metrics: *customer defection* probability, *average ads viewing time*, *price* paid by customer, *profit*, *revenue*, *waiting time prediction accuracy*, *system utilization*, and *arrival rate*.

(2) To design a framework for scalable AVS systems that can be used to accurately recognize subjects by efficiently controlling various networked cameras. We study the different factors that affect the practical implementation of the system. In addition, we investigate the major system modules and analyze the performance of the system. The two major performance metrics used are *recognition probability* and *system time complexity*.

Next, the two main objectives are described in more detail.

1.3 Designing a Scalable Delivery Framework for VOD with Advertisements

In this project, we address the main issues in the design of commercial VOD systems: scalability and support of advertisements. The scalability problem due to high server and network requirements has been addressed by Content Distribution Networks (CDNs), Peer-to-Peer (P2P) and Multicast approaches. In the first approach a wide number of geographically distributed servers are used and managed [11]. The second approach depends greatly on central seeders (servers) and stream reliability is still an issue [12, 13]. In both approaches, videos are delivered using unicast streams, and thus the problem is not completely solved. Multicast is naturally built towards serving one content to multiple clients at once. It can effectively deliver high-usage content and handle flash crowds. High-quality on-demand video distribution, IPTV, and many major video streaming applications fit nicely in a framework delivery built over native multicast [13, 14, 15]. In this research, we adopt the multicast approach.

Numerous techniques have been proposed to deal with the scalability challenge, especially in the areas of *resource sharing* and *request scheduling*. Resource sharing techniques can be classified into *client-pull* and *server-push* techniques, depending on whether the channels are allocated on demand or reserved in advance, respectively. The first category includes *stream merging* techniques [16, 17, 18, 19, 20, 21], which reduce the delivery cost by aggregating clients into larger groups that share the same multicast streams. The degree of resource sharing achieved by client-pull media delivery technique depends greatly on how the waiting requests are scheduled for service.

The overwhelming majority of prior studies on media delivery focused on regular content without any media advertisements. The use of ads 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

experience significant delays. 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. (4) Using ads results in better aggregation of requests and thus can reduce delivery costs.

Most media streaming companies use one of the following business models [22]: (i) pay-per-view, in which a client pays in advance for the requested media, (ii) subscription-based, in which a client pays a monthly fee, (iii) free with advertisement, in which a 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 a 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. The first two models are usually free of ads. However, many subscribers may not mind watching certain kinds of ads, such as trailers of new and interesting movies.

The main objectives of this work can be summarized as follows.

1. To develop a scalable delivery framework for streaming media content with video advertisements. The delivery framework combines the benefits of stream merging and periodic broadcasting.
2. To propose new scheduling policies that are well-suited for the proposed delivery framework.
3. To propose a new prediction scheme of ads' viewing time.
4. To propose an enhanced business model. The revenue generated from advertisements is used to subsidize the price.
5. To investigate the support of targeted advertisements, whereby clients receive ads that are well-suited for their interests and needs.
6. To provide the clients with the ability to select from multiple price options, each with an associate expected number of viewed ads.

To evaluate these objectives on a representative large scale workload, we have developed a detailed and accurate VOD system simulator.

1.4 Designing a Scalable Automated Video Surveillance System

In the second part, we discuss the design of an AVS system oriented towards subject-recognition applications. We focus on the management and the control of various Pan, Tilt, Zoom (PTZ) video cameras. PTZ cameras have appealing characteristics, such as (1) large field of regard (FoR), defined as all reachable view by all camera settings and (2) enhanced multiresolution views which are achieved by a controllable field of view (FoV), defined as the reachable view from the current camera settings. Currently, PTZ cameras are either used in pre-programmed tours or are manually controlled using special joysticks and keyboards [23, 24, 25].

The control of various video cameras has been a significant research problem [26, 27, 28, 29, 30]. Unfortunately, previous studies did not seek to optimize the most important performance metric, which is the subject/threat recognition probability. In this project, we investigate how various PTZ cameras can be controlled in such a way that maximizes the overall subject recognition probability. In the proposed system, we utilize image sets of potential threat (watch list). We consider the impact of various factors on the recognition probability, namely, (1) resolution, (2) pose, (3) occlusion, and (4) zoom-distance noise. As in [31, 32], we investigate the advantage of subject grouping whereby more than one subject are mapped to the same camera.

The objectives of this work can be summarized as follows.

1. To develop a camera management solution that provides the best tradeoff between the subject recognition probability and time complexity. We will consider both subject grouping and clustering mechanisms.

2. To characterize the impact of various factors on recognition probability. These factors include

resolution, pose, and zoom-distance noise.

3. To provide exhaustive analysis of camera management solutions, considering realistic workload, systems and design parameters.

As in the VOD, to evaluate these objectives on a representative large scale workload, we have developed a detailed and accurate AVS system simulator.

CHAPTER 2

RELATED WORK

2.1 Related Work on Video-on-Demand (VOD)

2.1.1 Resource Sharing

Numerous techniques have been proposed to deal with the scalability challenge, especially in the areas of *resource sharing*. Resource sharing techniques utilize the multicast facility. They can be classified into *client-pull* and *server-push* techniques, depending on whether the channels are allocated on demand or reserved in advance, respectively. The first category includes *stream merging* techniques [16, 17, 18, 19, 20, 21], which reduce the delivery cost by aggregating clients into larger groups that share the same multicast streams. The degrees of resource sharing achieved by client-pull media delivery techniques depend greatly on how the waiting requests are scheduled for service. The second category consists of *Periodic Broadcasting* techniques [33, 34, 35, 36, 37], which divide each media file into multiple segments and broadcast each segment periodically on dedicated server channels. These techniques are cost-performance effective for highly popular content but lead to channel underutilization when the request arrival rate is not sufficiently high.

Stream merging techniques combine streams when possible to reduce the delivery cost. They include *Patching* [38, 16], *Transition Patching* [17], and *Earliest Reachable Merge Target* (ERMT) [18, 39]. Patching expands the multicast tree dynamically to include new requests. A new request joins the latest *regular* (i.e., full) stream for the object and receives the missing portion as a *patch*. Hence, it requires two download channels (each at the video playback rate) and additional client buffer space. When the playback of the patch is completed, the client continues the playback of the remaining portion using the data received from the multicast stream and already buffered locally. To avoid the continuously increasing patch lengths, regular streams are retransmitted when the required patch length for a new request

exceeds a pre-specified value called *regular window* (Wr). Figure 2.1 illustrates how Patching works. Transition Patching allows some patch streams to be sharable by extending their lengths. Specifically, it introduces another multicast stream, called *transition stream*. The threshold to start a regular stream is Wr as in Patching, and the threshold to start a transition stream is called *transition window* (Wt). By contrast, ERMT is a near optimal hierarchical stream merging technique. It also requires two download channels, but it makes each stream sharable and thus leads 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 later can get extended to satisfy the playback requirement of the new client(s), and this extension can affect its own merge target. Figure 2.2 illustrates how ERMT works.

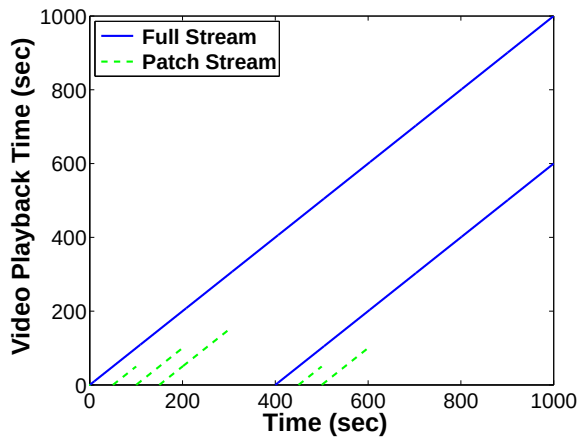


Figure 2.1: Patching

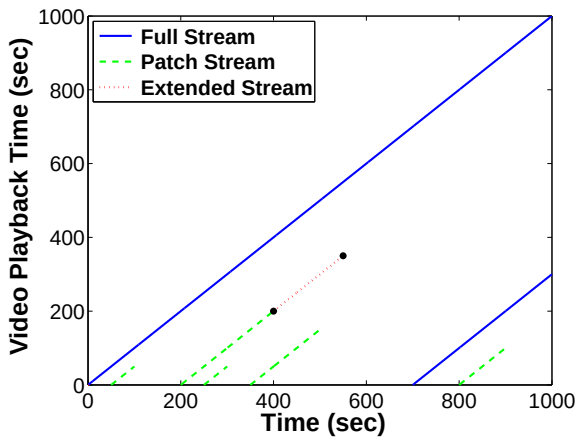


Figure 2.2: ERMT

Whereas stream merging techniques deliver data in a *client-pull* fashion, periodic broadcasting techniques [40, 33, 34, 37] employ the *server-push* approach. In particular, they divide each supported video into multiple segments and broadcast them periodically on dedicated channels. Thus, they can service virtually unlimited numbers of customers. However, these techniques can be used only for the most popular videos, and they require customers to wait until the next broadcast times of the first segments. Moreover, server channels may become underutilized when videos are requested infrequently.

2.1.2 Request Scheduling

A video streaming server maintains a waiting queue for every video and applies a scheduling policy to select an appropriate queue for service whenever it has an available *channel*. A channel is a set of resources (network bandwidth, disk I/O bandwidth, etc.) that is 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*.

Table 2.1: 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
Maximum Factored Queue Length (MQL)	longest factored queue length
Minimum Cost First - Total (MCF-T)	least total required cost
Minimum Cost First - Per (MCF-P)	least required cost per waiting request

The main scheduling policies include *First Come First Serve* (FCFS) [41], *Maximum Queue Length* (MQL) [41], *Maximum Factored Queue Length* (MFQL) [42], and *Minimum Cost First* (MCF) [43]. These policies are described in Table 2.1. MCF achieves the best overall performance. It captures the significant variation in stream lengths caused by stream merging techniques through selecting the requests requiring the least cost. The time length of a stream is directly proportional to the cost of servicing that stream as the server allocates a channel for the entire time the stream is active. *MCF-P (RAP)* is the preferred implementation of MCF. It selects the queue with the least cost per request and treats regular streams in a preferential manner because they are shared by later patches.

2.1.3 Using Advertisements and Business Models

The overwhelming majority of prior studies on media delivery focused on regular content without any media advertisements. Supporting ads has been discussed in only few studies. In [44], 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 others, 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. The study [45] provided a general discussion of pricing based on the techniques in [44]. It discussed the general relationships among arrival rate, price, and ads' ratio (to the total user viewing time).

2.2 Related Work on Automated Video Surveillance (AVS)

2.2.1 Face Recognition and Frame Quality

In the last decade many face datasets have been generated: FERET [46], FRVT [47] and [48], YaleA and YaleB [49], PIE [50], SCface [51], and UTK-LRHM [52]. Subjects with different ages, genders and ethnicities are pictured. Face images with different poses, illuminations, resolutions, occlusions, expressions and distances are acquired. Different cameras are also investigated in some sets. With the exception for the unpublished UTK-LRHM dataset, nothing has been covered on the effect of noise generated by zoom on the face recognition probability. On the other hand, many face recognition algorithms [53] (and references within), and applications [54] have been proposed and tested on these datasets. Most of the algorithms are proposed to recognize faces in controlled environment. Many solutions have been investigated to deal with specific problems like: illumination, pose, resolution, occlusion, expression. Most of these solutions require one or more of the following: (1) a pre-enhancement step using image processing algorithms, (2) training on different images for the same subject, (3) more than one entry on the dataset for the same subject and (4) extensive run time to execute. Face recognition applications [54, 55] have many limitations too, and work on a pre-specified environment. To mitigate the quality problem, these applications utilize a quality algorithm to measure the *faceness* of the image (how much the face in the image is appropriate for recognition).

2.2.2 Camera Scheduling and Assignment

Surveillance systems have limited number of cameras. A problem appears when the scene has more subjects than available cameras. Recent research has opened wider doors to PTZ cameras by utilizing their potential power and flexibility [26]. Connecting PTZ cameras in a network using master-slave framework has a huge potential. Distant Human Identification (DHID) system [56] illustrated a typical master-slave system configuration. A fixed wide angle camera is used to observe a scene and send information to a server, which in turn analyzes the scene and sends commands back to PTZ cameras. The PTZ cameras capture finer frames for targeted subjects. Each camera is assigned to exactly one subject at a time.

Choosing which subject to capture next is determined by the scheduling policy. Different scheduling policies adopt different objectives. In [57] the camera records one subject at a time, then the nearest subject to the current being recorded is scheduled for next capture. Inspired by network packet scheduling, authors in [27] examined the First Come First Serve (FCFS+), Earliest Deadline First (EDF+), and Random polices. They considered cameras as routers and subjects as packets. Another approach adopted by [30] was to take analogy from scheduling in operating system. It examined an online scheduling policy in which cameras considered as processors and subjects were the jobs. In this work, two approaches were investigated: (1) a preemptive approach in which a camera can be interrupted while recording and assigned to another subject and (2) a non-preemptive approach in which a camera is left to release the subject when the objective of its captures is fulfilled. The work in [28] scheduled subjects by pre-calculated plans, each plan has a list of subjects to record at different times. Plan building and assignment were addressed along with a detailed utility function used to order subjects inside the plan. The used utility function incorporated the pose, camera-subject distance and subject time-to-leave in the probabilistic utility function. Similarly, master-slave framework is adopted and the processing system generates detailed plans for all cameras at one time but each camera follows its plan separately. The

grouping was not discussed in this work and the results were shown for people tracking.

2.2.3 Subject Grouping

Subject grouping is another technique used to tackle the scalability problem. In this method, the system groups more than one subject to the same frame and assign it to one PTZ camera. The work in [31] used a master-slave system with a wide angle camera employed to collect scene information and pass it to the processing system. The processing system scans the data and runs a *Lattice-Based Grouping* (LBG) algorithm to generate a set of frames that represent the maximum "Objective Satisfaction". The algorithm checks for minimum overlap between selected frames. The satisfaction computed mainly based on weighted subject resolution and coverage. The algorithm run time is guaranteed to finish in a certain interval, and no other options provided if the algorithm failed to do so. PTZ cameras capture the assigned frames for a certain recording interval (each camera keeps watching single frame during one recording interval). The work assumed that all PTZ cameras have exactly the same FoR (located at the same place) and can cover the scene equally likely. This assumption oversimplifies the problem and reduces the candidate frames search domain by magnitude of number of PTZ cameras. In addition, the frames in the search domain are generated from a 2D environment. While the frame center (x,y) is copied from the scene and the zoom level is changed to populate different frame resolutions. Moreover, the quality of the captured frames did not include information about subjects' pose or occlusion, which makes face recognition task hard to achieve. The work addressed a generic objective and showed the results for grouping in a tracking application. The "subject satisfaction" was calculated in linear additive fashion. Thus, if subject S has satisfaction x_1 at time t_1 and x_2 at time t_2, the total acquired satisfaction after t_2 will be $x_1 + x_2$. This model does not take into account the probabilistic nature of the recognition process.

Paper [32] is another work that considered subject grouping. The work modeled the surveillance system as a bipartite graph. The subject camera assignment was obtained by applying maximum matching

algorithm. If the number of subjects exceeded number of cameras, the 2D frame projections are used to group subjects into sets. The grouping was based on the spatial scene and it was controlled by modifying the radius of a disc centered around each subject. After that, the sets were mapped to cameras. Subject occlusion was discussed and considered for later frame recording. The effect of grouping on number of covered subjects and algorithm running time was evaluated. However, the work did not mention any specific function to arbitrate between frames or groups in the scheduling process. It assigned a generic weight or cost to each node or edge in the generated graph. The final results discussed only the coverage issue.

CHAPTER 3

A SCALABLE DELIVERY FRAMEWORK FOR COMMERCIAL VIDEO-ON-DEMAND SYSTEMS WITH VIDEO ADVERTISEMENTS

3.1 Introduction

In this chapter, we present a framework for scalable delivery of media content with advertisements including different pricing models. The delivery framework combines the benefits of stream merging and periodic broadcasting. Ads are delivered on dedicated broadcasting channels, whereas stream merging is used for the primary media. A client starts by joining an ads' broadcast channel for some time and then receives the requested media by stream merging. We discuss two ad delivery options: *Partial-OK* and *Only-Full*. With Partial-OK, a client can join and leave the ads' channel at any time, and thus may watch some ads partially. With Only-Full, however, a client joins the ads' channel only at the beginning of an ad and leaves it at the end of an ad. Moreover, we propose a heuristic ad allocation algorithm. It arranges ads efficiently to reduce the average request waiting time. The revenue generated from the ads is used to subsidize the price. Subsidizing the price helps attract more clients, thereby increasing the overall revenue [45]. Pricing depends on the experienced quality-of-service (QoS). In particular, clients with longer ads' viewing times get lower prices. We also address the support of *targeted advertisements*. Additionally, we present four constrained scheduling policies for the proposed delivery framework. Furthermore, we develop a waiting-time prediction algorithm to estimate the expected ad viewing times in the scalable delivery framework.

We study, through extensive simulation, the effectiveness of various scheduling policies, the two ad delivery options, the heuristic ad allocation algorithm, and the support for targeted ads. We experiment with two stream merging techniques. The considered performance metrics include *customer defection* (i.e., turn-away) probability, *average ads' viewing time*, *price*, *system utilization*, *arrival rate*, *profit*, and *revenue*. We analyze the impacts of important workload and system parameters. Generally, customers

are more likely to buy a product or a service if it is offered at a lower price. In this chapter, we investigate the relation among arrival rate, price, and customer defection probability, defined as the probability that customers leave the system without being serviced because of waiting time exceeding their tolerance. 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 (i.e., upload bandwidth). The defection probability is an important Quality-of-Service (QoS) metric because customers 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 experiment with three different arrival rate models and their various parameters. The first model is an equation-based model, whereas the second utilizes the purchasing capacity and willingness models [58], and the third is a combination of the two.

The rest of the chapter is organized as follows. Section 3.2 presents the proposed delivery framework, ad allocation scheme, scheduling modifications, support for targeted ads. Section 3.3 presents the proposed pricing and waiting-time prediction scheme. Subsequently, Section 3.4 discusses the performance evaluation methodology. The main results are discussed and analyzed in Section 3.5. And finally we draw the conclusion in Section 3.6.

3.2 Delivery Framework

The proposed delivery framework combines the benefits of stream merging and periodic broadcasting. Periodic broadcasting is used for delivering the ads because they are potentially accessed more frequently than any individual primary media content, especially if each client is required to view a minimum number of ads. However, the primary media content is delivered using a scalable stream merging technique and can benefit from a high degree of aggregation because of the use of ads.

The main characteristics of the framework can be summarized as follows. (1) Clients start by joining an ad broadcast channel for some time and then receive the requested media by stream merging.

("Media" or "requested media", unless otherwise indicated, refers to one of the primary videos and not an ad.) (2) Ads are combined and broadcast on dedicated server channels. Hence, when beginning listening to an ads' channel, the client views different ads until streaming of the requested media content commences. (3) Ads are only viewed prior to watching the actual media content. Uninterrupted viewing of the primary media allows for a more enjoyable playback experience. (4) Scalable stream merging is used for the primary media content. (5) Clients snoop on preceding media streams while viewing the ads so as to reduce the delivery costs for joining the preceding streams. We refer to this feature as *early snooping*. Stream merging techniques require two client download channels at the video playback rate. Thus, while listening to one ad channel, the client uses the other channel for snooping.

3.2.1 Ad Allocation and Delivery

Ads are combined and broadcast periodically on dedicated server channels. Hence, when beginning listening to an ad channel, the client views different ads until streaming of the requested media content commences. The number of 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. The ads may or may not be uniform in length. Two options are possible as to whether to allow partial ad-viewing or only full-ad viewing. In the first option, called *Partial-OK*, a client can join and leave the ad channel at any time and thus may watch some ads partially. With the second option, called *Only-Full*, a client joins the ad channel only at the beginning of an ad and leaves it at the end of an ad. Thus, with this option, the client may experience a waiting time before beginning to listen to the ad channel. Moreover, even when a channel becomes available during the ad's playback, the streaming of the primary media can begin only after the current ad has finished. The maximum waiting time is equal to the longest ad duration. The Only-Full, however, is more appealing to companies that seek to advertise their products and services. Hence we will focus primarily on the Only-Full option.

The major drawback in Only-Full option is its imposed initial waiting time. To solve this problem

multiple ad channels can be used with time-shifted versions of the combined ads, as shown in Figure 3.1. In this figure, a total of four ads are supported and broadcast on three channels. Assuming the ad length is 30 seconds as in [59], with these three channels, the maximum time for a new request to reach the beginning of an ad is $30/3 = 10$ seconds, and the average time is $10/2 = 5$ seconds. With only one channel, the maximum waiting time to reach an ad is 30 seconds and the average is $30/2 = 15$ seconds. Recall that with the Partial-OK option, only one ad channel is required because clients view ads immediately without waiting time, except for the initial buffering time to smooth out the delay jitter.

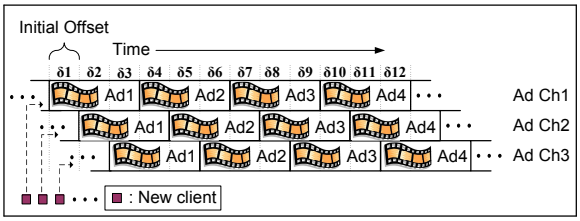


Figure 3.1: Ad Broadcast Channels

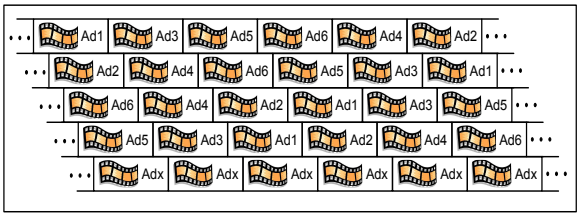


Figure 3.2: Heuristic Allocation

The ad allocation problem arises in the Only-Full option. It involves two aspects: how to order the ads on each channel and how to determine the offset from one channel to the other, which is simply referred to here as *channel offset* (δ). The channel offset is the time interval between the beginnings of two ad groups in two successive channels. In Figure 3.1, the offset is $\delta = \delta_1 = \delta_2$. Next, we discuss the allocation problem in the case of uniform ad lengths and variable ad lengths.

Uniform Ad Lengths

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 other. In general, the waiting time is simply the time to reach the beginning of the broadcast of the next closest ad. Thus, the waiting times depend on the spacing (or intervals) between (the beginnings of) successive ads. In Figure 3.1, the intervals between successive ads are δ_1, δ_1, ..., δ_{12}. The maximum waiting time is equal to the length of the largest interval, whereas the average waiting time is the weighted sum of all intervals divided by 2. Because of the repeating nature

of ads, the average can be found during the time of one ad group. The weighted sum is required if 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. The uniform ad lengths lead to uniform intervals. The optimal channel offset is basically the offset that leads to uniform intervals between successive channels and thus can be given by

$$\delta_{opt} = \frac{ad\_len}{N_{adCh}}, \tag{3.1}$$

where ad_len is one ad length and N_{adCh} is number of ad channels.

Variable Ad Lengths

For variable-length ads, the problem is more challenging. Assuming a fixed order in all channels simplifies the problem but is generally inefficient because it may lead to longer 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:

$$\delta_{opt} = \frac{G_{ad\_len}}{N_{adCh} \times M_{ad}}, \tag{3.2}$$

where G_{ad_len} is length of one ad group (channel), N_{adCh} is number of ad channels, and M_{ad} is number of ads used in one ad channel.

As discussed earlier, the average waiting time is the weighted sum of the intervals between successive ads within one ad group length. Assuming that δ_i is the i^{th} interval, the maximum and average waiting times can determined as follows:

$$t_{max} = \max_{i}^{n} \delta_i, \tag{3.3}$$

and

$$t_{avg} = \sum_{i}^{n} [(\delta_i/2) \times \frac{\lambda \times \delta_i}{\sum_j^n \lambda \times \delta_j}] = \frac{\sum_i^n \delta_i^2}{2 \times \sum_i^n \delta_i}, \tag{3.4}$$

where t_{max} is maximum waiting time, t_{avg} is average waiting time, λ is the request arrival rate and n is the number of intervals. Note that $\lambda \times \delta_i$ is the number of request arrivals within interval δ_i. Because of the square of δ_i in the equation, reducing the maximum waiting time tends to reduce the average waiting time.

Using Heuristic Ad Allocation Algorithm

Finding the optimal ad allocations is an NP hard problem. We propose a *heuristic ad allocation algorithm* that achieves a close performance to the optimal solution. The heuristic works as follows for a maximum of five channels. (1) On channel 1, place the longer ads closer to the ends. (2) On channel 2 (if it is not the last channel), reverse the order of channel 1. (3) 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. (4) 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. (5) For the last channel, compute the best combination leading to the least waiting time given the prior ordering of the other channels. The algorithm can be extended to a larger number of channels, but generally there is no need for the extra channels as the decrease in the waiting time would be too small with further channels. Figure 3.2 clarifies the heuristic allocation. The ads are numbered in decreasing order of length. Figures 3.3 and 3.4 show the optimal allocation (found using exhaustive test) and the heuristic allocation, respectively for four specific ads and three channels. Note that the average waiting time t_{avg} is only 6.2% longer than the optimal. The t_{avg} with the worst allocation (not shown) is approximately 19 seconds, which is about twice that of the optimal.

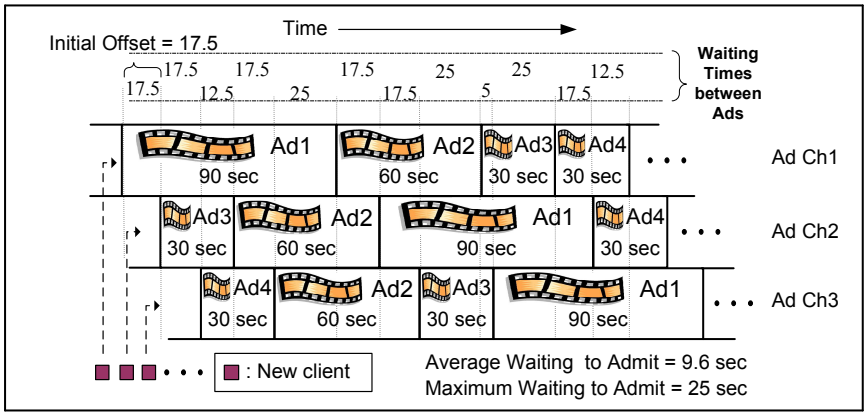


Figure 3.3: Optimal Ad Allocation

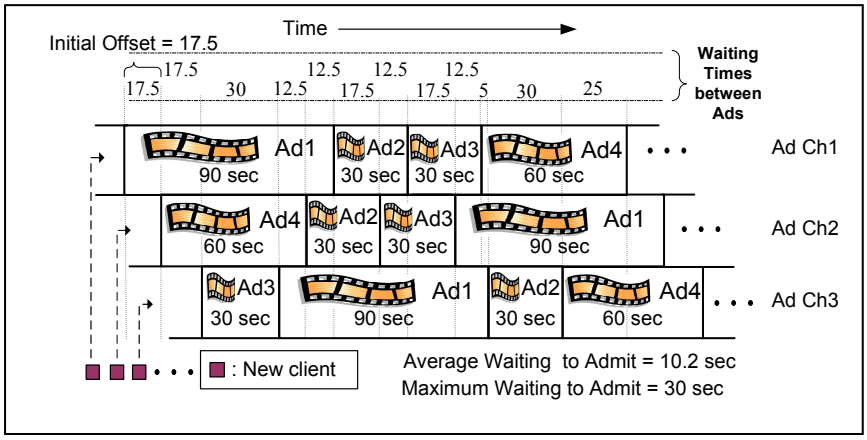


Figure 3.4: Heuristic Ad Allocation

3.2.2 Proposed Constraint-Based Scheduling Policies

Most existing scheduling policies are not well suited for the proposed delivery framework. For example, MCF, MQL, and MFQL attempt to serve requests as soon as possible, thereby reducing significantly the ad viewing time. Thus, we present three modifications of MCF-P (RAP) to ensure that ads are viewed by a large number of users: *Each N*, *Any N*, and *MST 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 MST N considers a video for service only if *most* of its waiting requests have viewed at least N ads. The value of N can be dynamically defined by the server using a probing-based mechanism. These modified versions can work with MQL and MFQL but we primarily use them for MCF-P (RAP) because it is the best performer.

Moreover, we propose a fourth policy, called *Maximum Ads Time* (MAT), which selects for service the video whose waiting requests have the longest total ad viewing time.

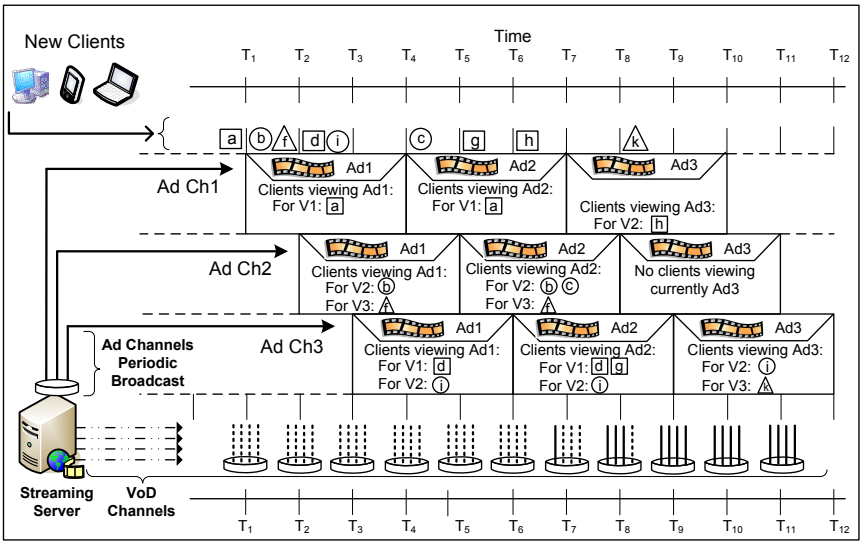


Figure 3.5: Illustrations of Any 2 Scheduling Policy [The system has four regular channels, three ad channels, and nine client requests (*a*-*k*)]

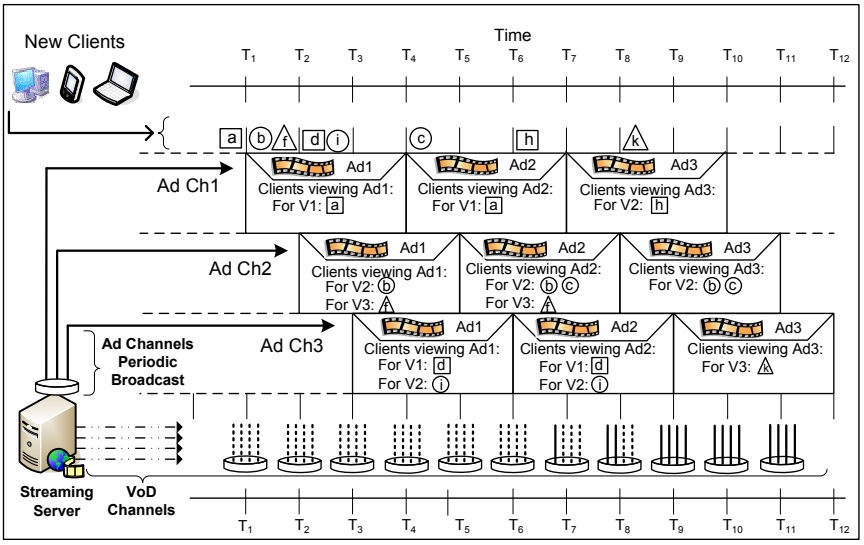


Figure 3.6: Illustrations of Each 2 Scheduling Policy [The system has four regular channels, three ad channels, and nine client requests (*a*-*k*)]

Figure 3.5 illustrates the operation of Any 2 Scheduling by an example. In this example, the scalable delivery system periodically broadcasts three ad channels, each having three specific ads (*Ad*1, *Ad*2, *and Ad*3). The server has also four channels dedicated for stream merging and used to stream requested videos. Arriving clients requests start listening to the next closest ad channel. 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 also 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 for Any 2. The server has four available channels 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. The server still has three available channels, and thus both clients are served at time $T8$, leaving one available channel for the streaming server. Since video $V2$ is qualified for admission according to 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 3.6 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 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 because video $V2$ has another client (client *c*) who joined channel $AdCh2$ and has not watched two ads yet. The server still has three available channels, and thus only client *f* is served at time $T8$, leaving two available channels for the streaming server.

3.2.3 Supporting Targeted Advertisements

In this section, we develop mechanisms for *targeted advertisements*, whereby clients receive ads that are well suited for their interests and needs. Support for targeted ads impacts ad allocation as well as resource and data sharing. Thus, we propose a new delivery approach that captures the interests of various clients. Ads are divided into various groups (sports, entertainment, food, etc.). The interests of clients can be manually provided by them or determined automatically by the system based on the history of viewed multimedia contents or the category of the currently requested video. A client can

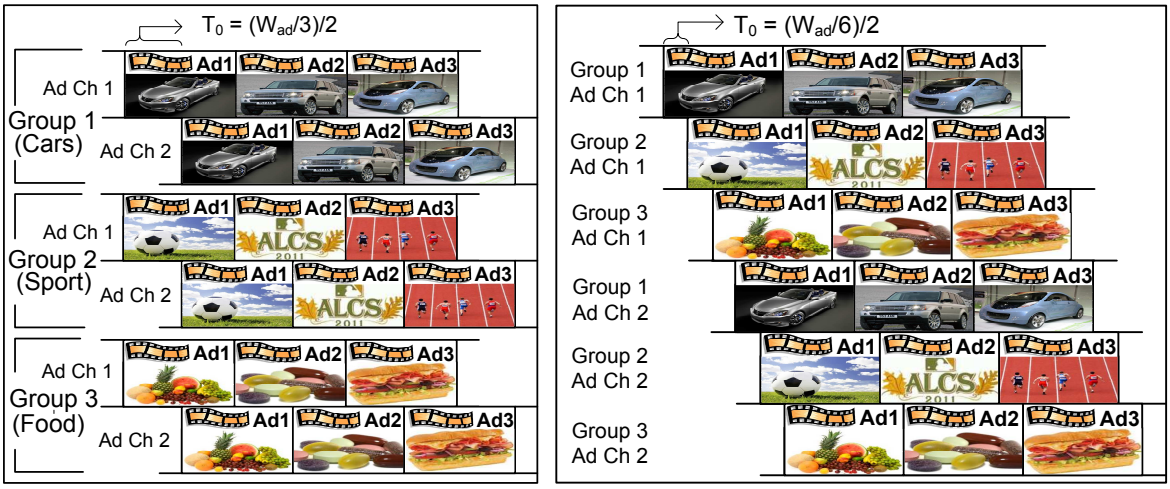
join any ad's group, but the ad's company pays in full only for the ads that perfectly match the client's interest. Otherwise, a partial payment is assessed based on the similarity between best matching group and the one to which the client is assigned. Similarity between two ad groups measures the likelihood of clients being interested in one ad group to show interest in the other one. 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 have some interest in nutritions and food. In our paper we assume this information is provided in advance by ad providers.

In a $G \times D$ configuration, G different ad groups exist and each group has D ad channels. The groups are numbered according to their similarity. Hence, groups with greater similarity have closer numbers. The system tries 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 Ads Channels Per Group*, and *Group Interleaving with One Ad Channel per Group*. Figure 3.7 illustrates these three alternatives. The system here supports three different ad groups (cars, sports, and food/nutrition). Figures 3.7(a) and 3.7(b) assume a 3×2 configuration, whereas 3.7(c) has a 3×1 configuration. In the first alternative, various ad groups are not interleaved but each group consists of interleaved ad 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 waiting time for viewing the first ad. This alternative can always achieve the best interest match with an average delay of $(Ad_{len}/D)/2$. In the second alternative, however, all ad channels are interleaved (whether they belong to the same or different groups). 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 channel offset (δ). Since the delay and interest match may have different ranges, they should be normalized. Subsequently, the client will be admitted to the ads' channel achieving the best delay and interest match tradeoff. Hence, we need to find i that maximizes the objective

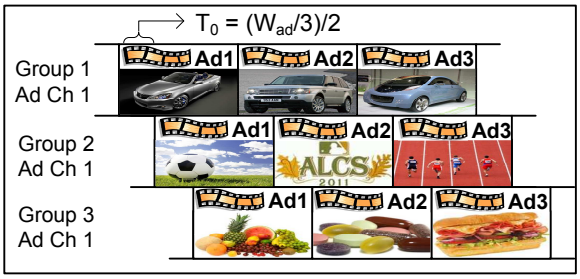
function

$$AdCh_i^* = \arg\max_{i=1}^{N_{adCh}} \frac{Match_i}{Delay_i}. \tag{3.5}$$

In the second configurations, the system average waiting time is $delay = Ad_{len}/(G \times D))/2$, assuming a uniform distribution for the interest groups. The third alternative is the same as the second but has only one ad channel per group and thus can be viewed as a special case. The initial delay is given by $delay = (Ad_{len}/G)/2$. Since this alternative uses fewer channels for ads, more server channels are utilized in streaming the requested multimedia content.



(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

Figure 3.7: Different Ad Allocation Alternatives for Supporting Targeted Ads

3.3 Proposed Pricing and Waiting-Time Prediction Schemes

We have developed a waiting-time prediction algorithm to estimate the expected ad viewing times in the scalable delivery framework. We have also proposed a pricing scheme based on the expected ad viewing time, royalty fee, and the current delivery cost of the requested video. (When desirable, it is also possible to remove the last from the price function. Note that the delivery cost varies with the current access rate. Videos with larger concurrent requests have lower delivery costs because of the better resource and data sharing achieved by stream merging.) In the pricing model, all ad revenues are used to subsidize the price, thereby attracting more clients. Additionally, the amount of subsidization the client receives is determined based on the predicted ad viewing time. The subsidization increases with the expected ad viewing time (or expected waiting time) because of the lower expected QoS level and the larger contribution to the generated ad revenues. Although this pricing scheme involves some sort of price prediction, this is justified by the following two reasons. (1) As will be shown later, the waiting-time prediction algorithm is highly accurate. (2) Inaccurate prediction involves no risk because only how the ad revenue is allocated to clients is affected and not the actual service cost. The total service cost (which includes the delivery cost) is determined dynamically based on the requested video.

In the envisioned system, the user is first presented with a menu showing the list of supported videos with their corresponding expected ad viewing times and prices. It may be desirable that the system allows the clients to stop the service without charge while receiving ads but before the streaming of the requested media starts. This can help in the unlikely situation when the client views much more ads than expected. To encourage clients who wait longer than expected to use the system again in the future, the system offers them *bonus points*, which can be applied later to increase their subsidization allocation.

The idea of waiting-time prediction for media streaming has recently been proposed in [21]. That study has presented a prediction algorithm that considers the applied scheduling policy and utilizes information about the current server state, including the current waiting queue lengths, the completion

times of running streams, and statistics such as the average request arrival rate for each video. It uses the completion times of running streams to know when channels will become available, and thus when waiting requests can be serviced. The algorithm, however, works only for stream merging techniques and cannot be used in the scalable delivery framework. Moreover, [21] did not consider advertisements or pricing.

Waiting-time prediction in the scalable delivery framework is complicated for the following reasons. (1) Both periodic broadcasting and stream merging techniques are used concurrently. (2) Different clients may join different ad channels and of course have to be dealt with in a differentiated manner. (3) Clients should receive multiple numbers of ads. Partial viewing of ads complicates pricing and may not be an attractive choice to advertisement companies. (4) Constraints on minimum ad viewing times must be met, such as those of Any N and Each N scheduling. (5) The number of available channels at a future time is somewhat hard to predict because of the interleaving of ad channels. (6) Multiple ad channels may become available simultaneously.

We present a highly accurate waiting-time prediction algorithm, called *Assign Closest Ad Completion Time* (ACA), for the scalable delivery framework. The main idea can be summarized as follows. As illustrated in Figure 3.8, when a new request is made at time T_{Now}, it is mapped to the ad channel with the closest ad start (or end) time (Ad Ch 2 in this example). The algorithm then examines the future ad start times on that channel in order (T_0, T_1, ...) for possible assignment as the expected time for that request. These ad start times represent possible expected times for serving the requested video. The request is tied to a specific ad channel because we assume here that partial-ad viewing is not permitted and thus requests can be served only at ad boundaries. At each ad start time, the algorithm estimates the number of available channels and predicts the videos that can be served at that time. The process continues until the expected video to be served is the same as the currently requested video or the *prediction window* (Wp) is exceeded. Wp controls the implementation complexity and introduces a tradeoff between prediction accuracy and the percentage of clients receiving expected times. As Wp increases,

the algorithm gives expected times to more incoming requests but at the expense of increasing the time complexity (a function of W_p × number of videos) and more importantly reducing the accuracy. In the considered example, $W_p = 3$ ad lengths and thus the algorithm examines the ad start times in the order T_0, T_1, T_2, and T_3, until an expected time is determined for the requested video.

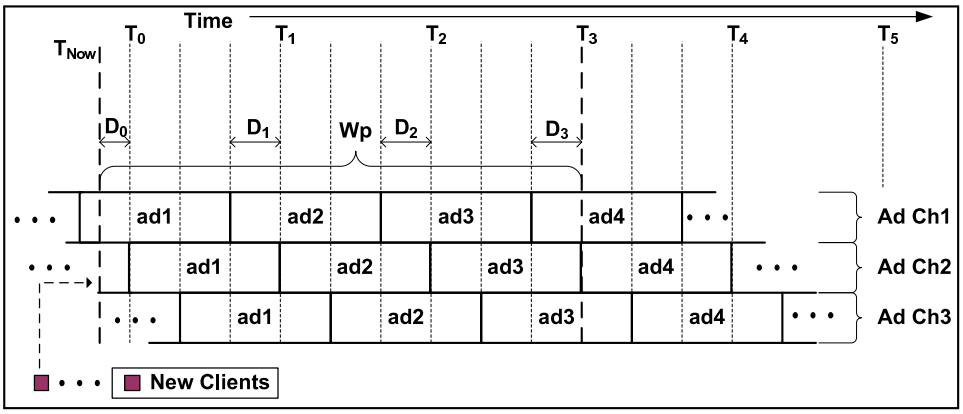


Figure 3.8: Illustration of the ACA Algorithm

Figure 3.9 shows a simplified version of the algorithm, which is performed upon the arrival of request R_i for video v_j if the server is fully loaded. Otherwise, the request can be serviced immediately. The algorithm has two major loops: the outer one (Lines $7-38$) goes over successive ad start times on ad channel $adChNo$ (corresponding to the new client), and the inner (Lines $21-33$) goes over all available channels at that time. Note that at any ad start time T, a video may be assigned only once. Thus, when a video is selected, its objective function is reset to -1, specifying that it should not be selected again.

Figure 3.10 illustrates the operation with an example. Each ad channel is split here into three logical channels, one for each video.

3.3.1 Video Selection Prediction

Let us now discuss how to predict the videos to be served at any particular ad start time T. First, the algorithm (Line 9) determines the videos that will be qualified at that time based on any minimum ad viewing constraints, such as those imposed by Any N and Each N. The qualification algorithm will be discussed later on. For each video that qualifies, the algorithm (Lines $10-15$) estimates its waiting

queue length at time T, if the scheduling policy requires so (such as MCF-P, Any N, and Each N), based on the video arrival rates, which are to be computed periodically but not frequently. The expected queue length for video v at time T can be found as follows:

$$expected\_qlen[v] = qlen(v, adChNo) + \lambda[v] \times ((T_0 - T_{Now}) + examined\_starts \times ad\_len/N_{adCh}), \quad (3.6)$$

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 ad 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. Dividing by N_{adCh} in Equation (3.6) is because not all arrivals belong to the considered ad channel. Note that the same video may be serviced again at a later ad start time. Equation (3.6) assumes that video v has not been identified before (while running the ACA algorithm for the new request) 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 the latest assigned time ($assigned\_time[v]$) at which video v is expected to be served and T. In that case, the expected queue length for video v at time T is given by

$$expected\_qlen[v] = \lambda[v] \times (T - assigned\_time[v])/N_{adCh}. \quad (3.7)$$

Finally, the algorithm (Line 16) finds the objective function for each qualified video based on the scheduling policy and selects (Line 23) the video with the maximum objective function.

3.3.2 Channel Availability Prediction

The number of available server channels (N_c) is computed as follows. For the closest ad start time (T_0 in Figure 3.8), the number of available channels is equal to the number of currently available channels plus the number of channels that will be available in the interval (D_0) between T_{Now} and T_0, inclusive. The currently available channels at time T_{Now} may not be used for later ad start times (T_1, T_2, etc.) because they may be consumed by requests on other ad channels ($AdCh1$ and $AdCh3$). Instead, an average leftover value can be computed dynamically to estimate the number of available server channels that are carried over from one ad start time to the immediately following ad start time on the next ad channel. Thus, the number of available channels at a later ad start time T is computed (Line 36) as the number of channels that will be available in the immediately preceding ad_len/N_{adCh} duration (D_1, D_2, or D_3) plus the average leftover value.

3.3.3 Scheduling Qualification Prediction

Scheduling policies imposing minimum ad viewing times (such as Any N and Each N) require the determination whether a video will be qualified for service at a considered ad start time. This is done by the proposed *qualification algorithm*. It determines whether video v on ad channel $adChNo$ will be qualified for service at time T. For Any N, this algorithm can be explained as follows. If the video has waiting clients on ad channel $adChNo$ (the ad channel to which the new request Ri is mapped to), then qualify it only if at least one of these clients has viewed at least N ads. If the video, however, does not have any waiting clients, or it has already an expected time during the current run of the prediction algorithm (for request Ri), then the algorithm determines the probability ($QProb$) that it will have future clients and that at least one of them will have viewed at least N ads by the time T. The function $Prob(1, v, t_d)$ returns the probability of at least one arrival for video v during duration t_d. The video will only be qualified if $QProb$ is larger than a certain threshold, called the *qualification threshold* (Q_{Th}). To clarify the concept, let us discuss how $QProb$ for video v at time T can be computed upon the arrival

of a new request (for another video) at time T_{Now} for the case when video v has no waiting requests and has not been assigned an expected time during the current run of the ACA algorithm. Referring to Figure 3.8, $QProb$ in this case is the probability that new expected requests for video v will view at least N ads and can be given by

$$QProb = P_1 + (maxViewedAds - N) \times P_2, \tag{3.8}$$

where P_1 is the probability of at least one arrival for video v during duration D_0 (whose length is $T_0 - T_{Now}$), P_2 is probability of at least one arrival during a duration of length ad_len/N_{adCh} (corresponding to D_1, D_2, etc.), and $maxViewedAds$ is the number of ad intervals between T_{Now} and T.

The main difference in the case of Each N is that the video is qualified only if each waiting client (real or expected) has viewed at least N ads. Ensuring the satisfaction of the Each N constraint for expected clients introduces some complications, which have been addressed in the algorithm. MAT, MCF-P, MQL, and MFQL do not require qualification.

3.3.4 Alternative Prediction Scheme

As discussed earlier, the ACA algorithm predicts the scheduling outcomes during a prediction window (Wp), which is constrained to reduce the implementation complexity and ensure satisfactory prediction accuracy. Hence, in practical situations, ACA may not be able to find an expected time for every incoming client. This issue can be addressed by using a hybrid scheme: if ACA does not return an expected time, the average waiting time for the video is used. This hybrid scheme is called *ACA+APW*. APW stands for *Assign Per-Video Average Waiting Time* (APW).

3.3.5 Multiple Price Option: The Client-Driven Price Selection

This section extends the idea of price prediction by presenting each client with multiple price options. Providing clients with multiple price options enhances customer satisfaction and system profitability.

The idea of the CPS algorithm can be described as follows. When a new request is made, it is mapped to the ad 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 ad 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 3.11 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 ad 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.

3.4 Performance Evaluation Methodology and Results

We have developed a simulator for a media server that supports various video delivery techniques and scheduling policies. We have validated the simulator by reproducing several graphs in previous studies. The simulation stops after a steady state analysis with 95% confidence interval is reached. The analyzed performance metrics here are *customer defection probability*, *average number of ads viewed per client*, *prediction accuracy*, *percentage of clients receiving expected times of service* (PCRE), *price*, *channel utilization*, *arrival rate*, *profit*, and *revenue*.

The proposed heuristic ad allocation algorithm is investigated by extensive simulation using another dedicated simulator developed for this purpose. Table 3.2 shows the main parameters used to examine various configurations (i.e., various combinations of the number of ad channels and the number of ads per channel). For each configuration, 750 runs are conducted. In each run, ad lengths are chosen randomly in a weighted fashion to form a new combination. The different ad 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. For example, an ad length of 30 seconds is usually the most common. The heuristic algorithm is compared with the optimal algorithm and random algorithm. The optimal algorithm searches the whole domain of possible combinations, whereas the random algorithm chooses an allocation randomly. Table 3.3 shows the main parameters used for assessing the performance of the support of targeted ads.

Table 4.1 summarizes the workload characteristics used for the other parts of this chapter as well as targeted ads. Like most prior studies, we assume that the arrival of the requests to the server follows a Poisson Process with an average arrival rate λ and that the access to videos is highly localized and follows a Zipf-like distribution with skew parameter $\theta = 0.271$. We characterize the waiting tolerance of customers in terms of the number of ads by a Poisson distribution.

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 clients, which directly depends on the customer 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 chapter, we study our system under three different models of the arrival rate: *Equation-Based*, *Willingness-Based*, and *Hybrid*. The first model captures the fact that the arrival rate will not only depends on the price but also the customer defection probability. To characterize this behavior, we analyze two main functions:

$$\lambda = \frac{c_1(1-d)(1-p/p_{max})}{c_2 + c_3 d^2 + c_4(p/p_{max})^2} \quad and \quad \lambda = \frac{c_1(1-d)(1-p/p_{max})}{c_2 + c_3 d + c_4 p/p_{max}}, \tag{3.9}$$

where d is the defection probability, p is the price, p_{max} is the maximum price at which no customer will be interested in the service, and c_1, c_2, c_3, and c_4 are constants. The default values of these parameters in this chapter are 1, 0.5, 1, and 1, respectively. Dividing p by p_{max} serves to normalize the price. These two equations are referred to as *Function 1* and *Function 2*, respectively. Figures 3.12 and 3.13 depict these two functions, respectively.

In the other hand, the willingness-based model utilizes client purchasing capacity and willingness models. Economic theory 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 [58]. 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 (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

$$Prob(willingness) = \begin{cases} 1 - (\frac{p}{y})^{\delta} & 0 \leq p \leq y, \\ 0 & p > y, \end{cases} \tag{3.10}$$

where δ is the *elasticity* [58]. As δ increases, more clients are willing to spend. Note that the arrival rate is an input when the willingness-based model is used and an output with the equation-based arrival rate model. In the willingness-based model, the arrival rate represents the maximum rate the server can attain and properly handle.

Finally, the hybrid model combines the equation-based and willingness-based models. It captures the impacts of client purchasing capacity, willingness models, and customer defection probability Figure 3.14.

$$\lambda = \frac{(1-d)}{c_1 + c_2 d^2}, \tag{3.11}$$

where as in the first model, d is the defection probability, c_1, and c_2 are constants. The default values of these parameters in this chapter are 0.5, 1 respectively.

We consider here a commercial *Movie-on-Demand* system with 120 titles, each of which is 120-minute long. Stream merging is done using Patching and ERMT. We analyze the impacts of both server capacity (i.e., number of server channels) and number of ad channels. 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 framework. With targeted ads, however, it is 5 cents plus a fraction of 10 cents proportional to the level of similarity between the group supplied and the best matching group, with $(5 + 0)$ cents for the worst

Table 3.1: Summary of Workload Characteristics [VOD, General]

Parameter	**Model/Value(s)**
Request Arrival	Poisson Process
Request Arrival Rate	Variable, Default = 40 Requests/min
Server Capacity	200-600
Video Access	Zipf-Like, Skewness Parameter $\theta = 0.271$
Number of Movies	120
Movie Length	120 min
Waiting Tolerance Model A	Poisson, min = 3 ads, mean= 5 ads, max = 8 ads
Waiting Tolerance Model B	Poisson, Expected Service Time + Wad, Wad: Variable, Default= 2 Ad lengths, if client doesn't have expected time: use Model A
Pricing Model	Equation-Based, Willingness-Based, Combined
Prediction Window (W_p)	Default: 4
Qualification Thresh. (Q_{Th})	Default: 0.25
Scale, Shape, Elasticity	$b : 1.0, \alpha : 1, \delta : 7$
Ad Length	30 sec
Number of Different Ads	8
Number of Ads Channels	Variable, Default = 3

Table 3.2: Summary of Workload Characteristics [Heuristic Algorithm]

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 (seconds)
Number of Iterations	750

match and $(5 + 10)$ 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.

3.5 Result Presentation and Analysis

In this section, we generally assume a total of three ad channels, unless otherwise indicated. We will focus primarily on the Only-Full option. Our experiments show that both options approach each other in terms of performance metrics for high server capacities. For low and moderate server capacities, however, Partial-OK reduces the defection rate by approximately 7% and reduces the waiting time by approximately half an ad length, compared with Only-Full option, considering that only one ads channel

Table 3.3: Summary of Workload Characteristics [Ad Targeting]

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

is needed for the Partial-OK option.

3.5.1 Effectiveness of Heuristic Ad Allocation Algorithm

Figure 3.15(a) compares the effectiveness of the heuristic ad allocation algorithm with the optimal and random algorithms for various configurations (i.e., combinations of the number of ad channels and the number of ads per channel) in terms of the average customer waiting time before viewing the first ad. The figure indicates that the proposed heuristic algorithm perform close to the optimal. On average, it differs by 5.9% from the optimal algorithm, whereas the random differs by 15.8%.

3.5.2 Results under the Equation-Based Model

Effectiveness of Existing Scheduling Policies

Figure 3.16 compares the performance of existing scheduling policies in the proposed delivery framework environment when ERMT is used for delivering the primary media content. The performance of FCFS, MQL, and MCF-P (RAP) are plotted in terms of the defection rate, average number of views ads, and unfairness. MFQL is not shown because it does not perform well in the stream merging environment. The results here demonstrate that MCF-P achieves the best overall performance in terms of the two most important metrics. As expected, FCFS has the best fairness. It also reduces the defection probability better than MCF-P for very high server capacities. Unfortunately, the average viewing time with the overall better policies (MCF-P and MQL) is rather small. With MCF-P, a large number of customers did not view any ad at all, which reduces the price subsidization. These results motivate the

new variants of MCF-P: Each N and Any N. These variants can also be used for MQL but we choose only MCF-P because of its higher performance.

Effectiveness of Constraint-Based Scheduling Policies

Let us now discuss the effectiveness of the proposed scheduling policies. Figure 3.17(a) and 3.17(b) plot the defection rate versus the average number of viewed ads with all new policies for ERMT and Patching, respectively. With Patching, the results exhibit the same behavior, but ERMT is superior in terms of the defection rate and average waiting time. Detailed comparisons between Patching and ERMT are shown later in Figure 4.15. The curve for each policy is generated by varying the server capacity. Higher server capacities produce lower defection rates. MCF-P (Each 3) occupies the least interesting area of operation since its leads to very high defection rates at very high ad viewing times. MCF-P (Any 2), MCF-P (Any 1) and MCF-P (Each 1) have the best regions of operations. MCF-P (MST 2) performs close to MCF-P (Each 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 but can be controlled only by changing the server capacity.

Figures 3.18 and 3.19 illustrate the impact of N in MCF-P (Any N) and MCF-P (Each N) on the defection probability, average number of viewed ads and utilization respectively. As expected, the defection probability increases with N because of the tighter constraints on scheduling. These constraints increase the ad viewing times and also the probability of exceeding the customer waiting tolerance for the desired content. Note that N has two conflicting impacts on defection rate. Increasing N increases the waiting time for the primary media content and also increases resource and data sharing. Thus, MCF-P (Any 2) has a more balanced impact and achieves the least defections. MCF-P (Any N) always achieves better utilization because of its less restrict constraints on the number of clients who must view N ads.

Next, we primarily focus on MCF-P (Any 2), and MCF-P (Each 2), MCF-P (MST 2), and MAT, all

operating over ERMT. The same value of N is chosen for the two MCF-P variants to ensure a realistic comparison.

Figures 3.20 compares MCF-P (Any 2), MCF-P (Each 2), MCF-P (MST 2), and MAT in eight different performance metrics, when assuming arrival rate Function 1. MCF-P (Any 2) remains the best overall performer in terms of the defection rate, profit, and revenue. We have experimented with different c_3 and c_4 values. MCF-P consistently gives the highest profit and revenue although the relative performance of MAT, MCF-P (MST 2), and MCF-P (Each 2) was not always the same. MCF-P (Any 2) has high unfairness, but this metrics carries less importance. Due to the largest ad viewing time, MCF-P (Each 2), MCF-P (MST 2), and MAT lead to lower prices than MCF-P (Any 2), as shown in Figure 3.20(f). They, however, yield lower profit and revenue because the increase in the defection rate is more significant. Interestingly, the average delivery cost with MCF-P (Any 2) and MCF-P (Each 2) slightly decease then remains nearly constant with the server capacity. One would expect the delivery cost to increase with server capacity because of lower resource and data sharing. The arrival rate, however, also increases with server capacity when these two policies are used as shown in Figure 3.20(g). Increasing the arrival rates balances the impact of increasing the server capacity on delivery cost. As expected the server channel utilization is the highest with MAT because it does not impose any minimum ad's viewing time requirement. MCF-P (Each 2), MCF-P (Any 2), and MCF-P (MST 2) have lower utilization because they force some requests to wait even when channels are available. MCF-P (Each 2) imposes more waiting and thus results in lower utilization than MCF-P (Any 2).

Impact of Number of Ads Channels

Figure 3.21 shows the impact of the number of ad channels when server capacity is fixed at 500. It illustrates that MAT and MCF-P (Any 2) remain the best performers with ERMT regardless of the number of ad channels. In Patching (not shown), MCF-P (Any 2) is still the best choice. As expected, the number of viewed ads and thus the defection rate increases with the number of ad channel because

of the fewer number of channels used for delivering the desired (primary) media content. Increasing the number of ad channels reduces the initial waiting time for receiving the ads, but a value larger than four in the studied workload is not beneficial because the waiting time decreases only slightly.

Comparison between Patching and ERMT

Figure 4.15 compares the performance of Patching and ERMT in the proposed delivery framework when MCF-P (Any 2). These results demonstrate that ERMT achieves significantly better than Patching in all performance metrics (except utilization at high server capacity and average viewed ads per customer at low server capacity). This is due to the hierarchical stream merging nature of ERMT. The utilization is lower with ERMT for high server capacities because it services (almost) all requests without using all the channels. The only disadvantage is that ERMT is much more complex to implement. The high performance benefits, however, could justify its application.

Impact of Arrival Rate Function

Figure 3.23 shows the results for the equation-based arrival rate model with Function 2 (i.e., the second function represented in Equation 3.9). These results exhibit a similar trend as that of Function 1, but with lower customer defection rates, due to the nature of the function.

Supporting Targeted Ads

Figure 4.14 compares the effectiveness of the three ad allocation alternatives: *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*. The first two alternatives have a 5×2 configuration (i.e., five different groups each with two ad channels). The third alternative still has five groups but each has one ad channel only. The results show that the third alternative performs the best in terms of revenue and profit. It has a moderate customer defection ratio, but it attracts more clients than the other models

(as indicated by the increased arrival rate) since it has fewer channels dedicated to ad broadcasting and more for servicing the requested videos. Patching exhibits a similar behavior and thus the results are not shown.

3.5.3 Results under the Willingness-Based Model

Figure 4.16 shows the results under the willingness-based model. We define the effective arrival rate as the arrival rate of customers who meet the capacity criterion and are willing to purchase the service. As shown in Figure 3.25(d), the arrival rate is high at low server capacities, causing the customers to compete more for the limited resources and leading to higher defection rates. Since only those customers who are willing to accept the price are admitted, we observe higher revenues and profits compared with the equation-based model. Although MCF-P (Any 2) leads to the highest prices for low and moderate server capacities, it still achieves the best revenue and profit and the least defection rate and average waiting time.

3.5.4 Results under Hybrid Equation and Willingness Model

Figure 3.26 shows the results under the hybrid equation and willingness model. In this model, both the willingness and defection rate are factored in the arrival rate control. Hence, we observe a behavior lying between the equation-based and willingness-based models. The arrival and defection rates are mid-ranged between the two individual models. MCF-P (Any 2) achieves the best performance among all scheduling policies. It leads to the best revenue, profit, arrival rate, and customer defection rate, at a slightly higher price.

3.5.5 Effectiveness of Waiting-Time Prediction

Let us start by analyzing the performance of the proposed ACA prediction algorithm in terms of average deviation and PCRE. Two straightforward approaches are used to evaluate its effectiveness:

Assign Overall Average Waiting Time (AOW) and *Assign Per-Video Average Waiting Time* (APW), which work exactly as named. Figure 4.13 compares the average deviation results under Any 2, Each 2, and MAT, respectively, when stream merging is done using ERMT. Patching and Transition Patching exhibit a similar behavior but achieve slightly lower deviations (as shown in Figures 3.28 and 4.15). These results demonstrate that ACA is highly accurate, especially when combined with Any 2. The deviation in that case is within 7 seconds, which is less than 25% of the ad length.

Figures 3.28 and 4.15 show the impacts of the qualification threshold (Q_{Th}), prediction window (W_p), and minimum number of ads constraint (N) on the average deviation and PCRE, respectively. The results are shown for different stream merging techniques or server capacities. The effect of dynamic Q_{Th} is not shown because it does not perform well. As expected, both these metrics increase with Q_{Th} and W_p, which suggests that they should be chosen based on a good compromise. In contrast, the average deviation decreases with N (in the Any policy) up to a certain point ($N = 2$ or 3) and then starts to increase. Although PCRE always decreases with N, its value at 2 is close to that at 1. As will be shown later, N should be chosen based on the overall system performance and not only the average deviation.

```
1 for (v = 0; v < N_v; v++) // Initialize
2   assigned_time[v] = −1; // Not assigned expected time
3 adChNo = Get # of the ads' channel with the closest start time;
4 T = Get next ad's start time; T_0 = T; examined_times = 0;
5 // Find number of available channels at time T
6 N_c = available_channels + will_be_available(T_Now, T);
7 while (T < T_Now + W_p) { // Loop till prediction window is exceeded
8   for (v = 0; v < N_v; v++){
9     if (isQualified(v, T, adChNo)) {
10      if (assigned_time[v] == −1)
11        expected_qlen = qlen(v, adChNo) + λ[v]×
12        ((T_0 − T_Now) + examined_times × ad_len/N_adCh);
13      else // Video v has been assigned an expected time
14        expected_qlen = λ[v] × (T − assigned_time[v])/N_adCh;
16      objective[v] = find scheduling objective for video v;
17    } // end if (isQualified(v, T, adChNo))
18    else
19      objective[v] = −1; // v is not qualified
20  } // end for (v = 0; v < N_v; v++)
21  while (c = 0; c ≤ N_c; c++){ //for every available channel
22    // Find the expected video to serve at time T
23    expected_video = find video with maximum objective;
24    // -1 objectives are discarded
25    if(expected_video == v_j) {
26      Assign T as the expected time to request R_i;
27      return;
28    }
29    else {
30      assigned_time[expected_video] = T;
31      objective[v] = −1; // Can't be selected again
32    }
33 } // end while (c = 0; c ≤ N_c; c++)
34  T = T + ad_len; //Proceed to the next edge
35  // Find number of available channels at time T
36  N_c = left_over + will_be_available(T − ad_len/N_adCh, T);
37  examined_times++;
38} // end while (T < T_Now + W_p)
```

Figure 3.9: Simplified Algorithm for ACA [performed upon arrival of request R_i for Video v_j]

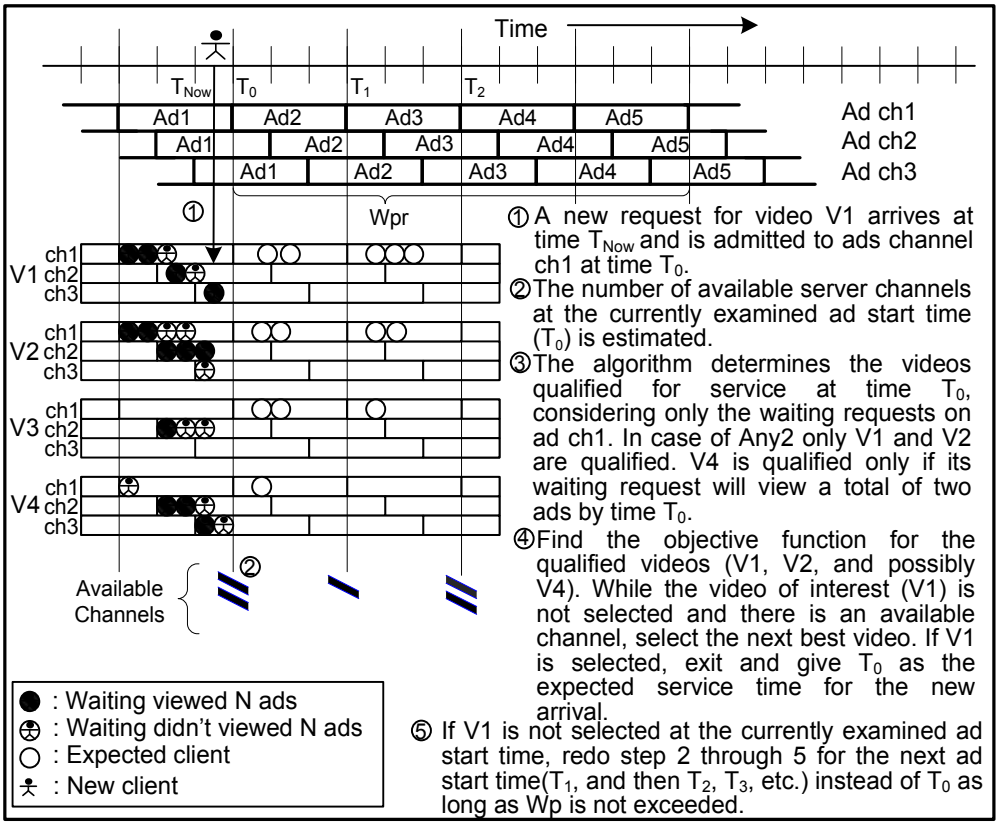


Figure 3.10: An Example Illustrating the Operation of ACA

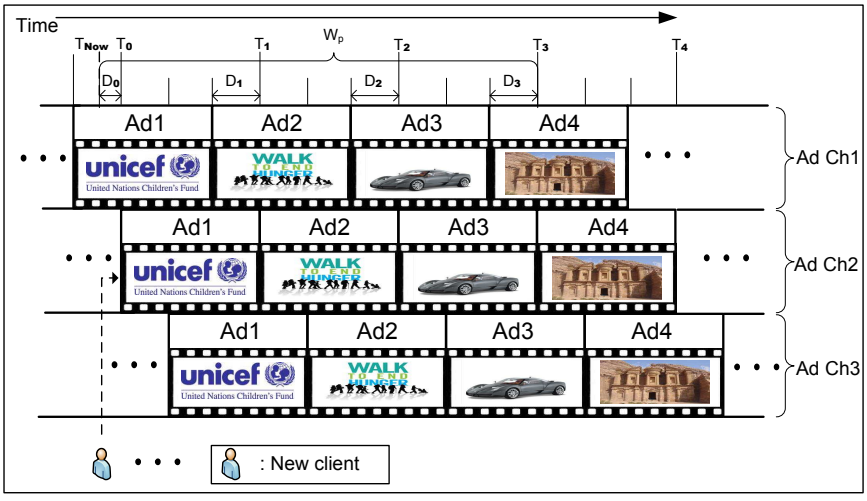


Figure 3.11: Illustration of the CPS Algorithm

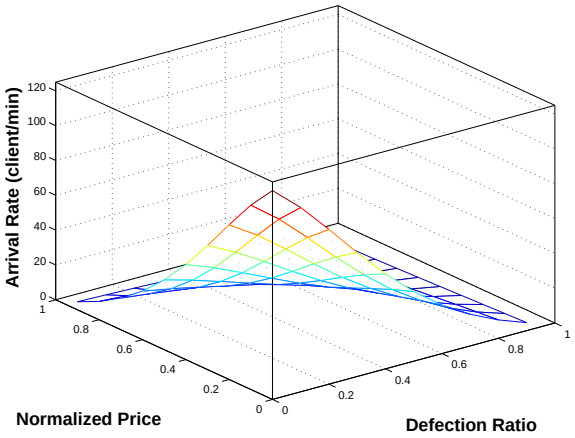


Figure 3.12: Arrival Rate Function 1 [$c_1 = 1$, $c_2 = 0.5$, $c3 = c_4 = 1$]

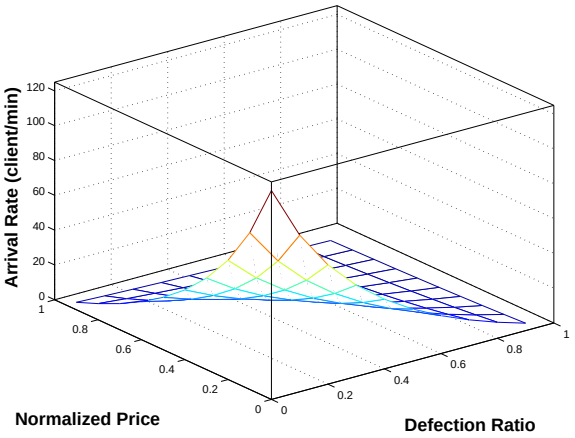


Figure 3.13: Arrival Rate Function 2 [$c_1 = 1$, $c_2 = 0.5$, $c3 = c_4 = 1$]

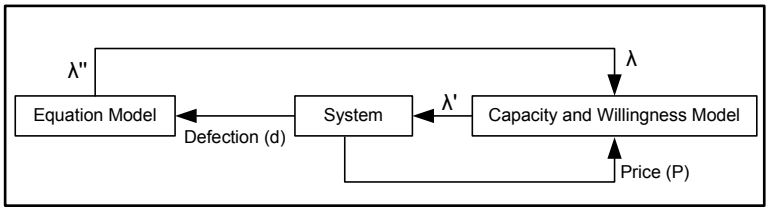
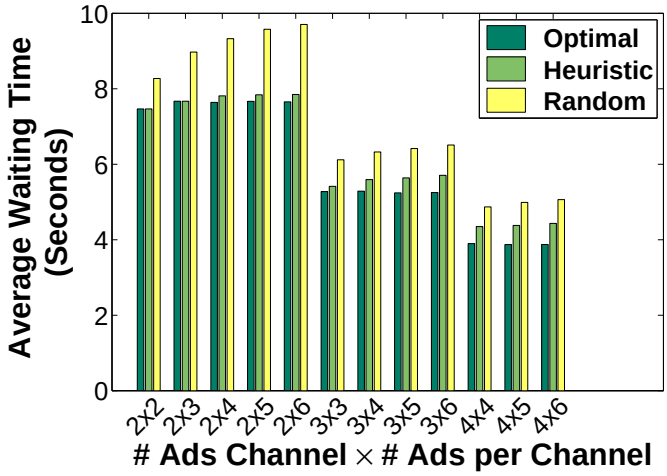
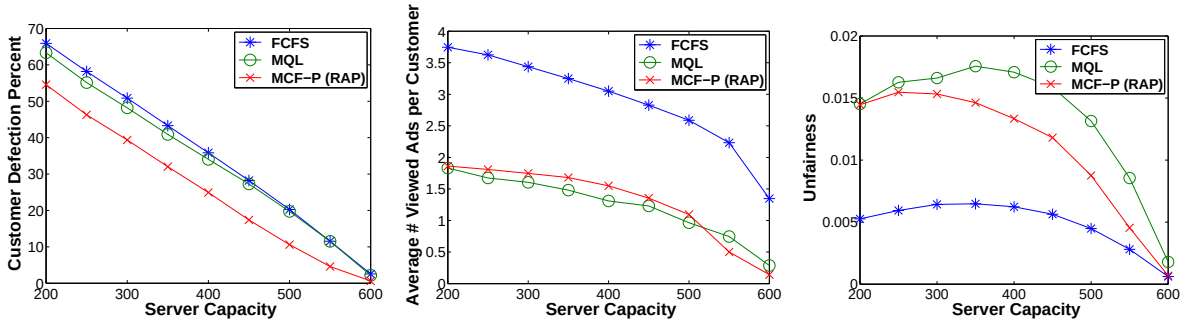


Figure 3.14: Arrival Rate Dependency Flowchart, λ and Price(P) are Initialized at the System Start



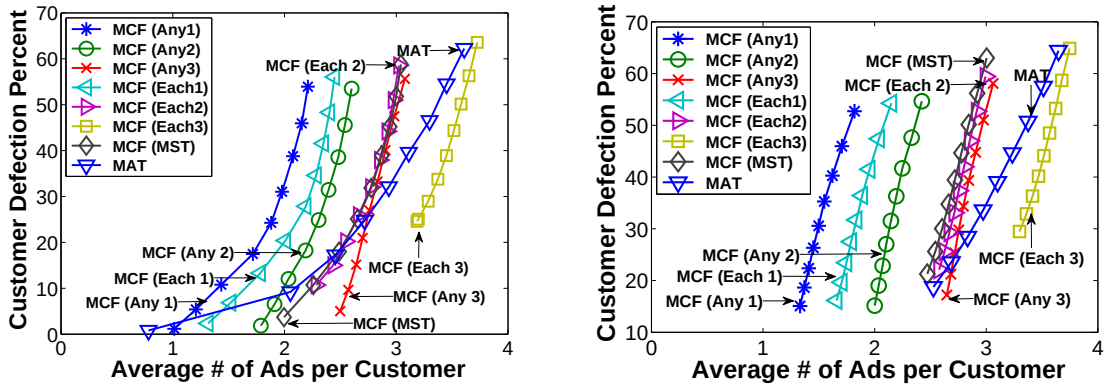
(a) Average Waiting Time

Figure 3.15: Effectiveness of the Heuristic Ad Allocation Algorithm



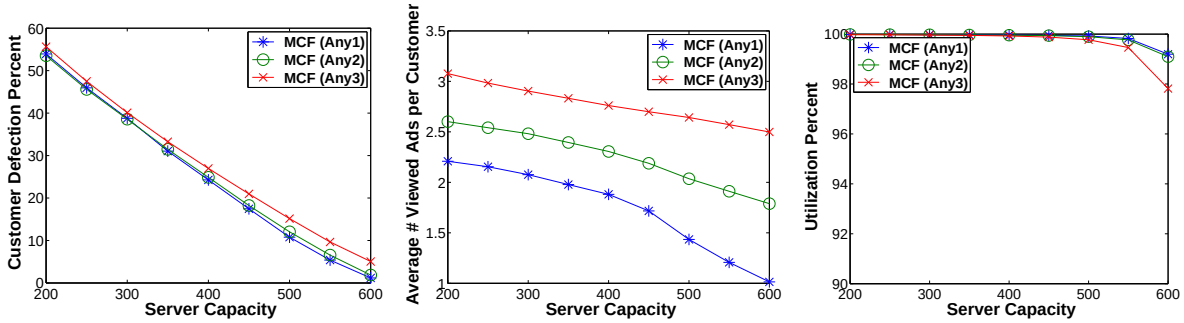
(a) Customer Defection Rate (b) Average Number of Viewed Ads (c) Unfairness

Figure 3.16: Comparing Effectiveness of Existing Scheduling Policies [ERMT, Equation-Based Model with Function 1]



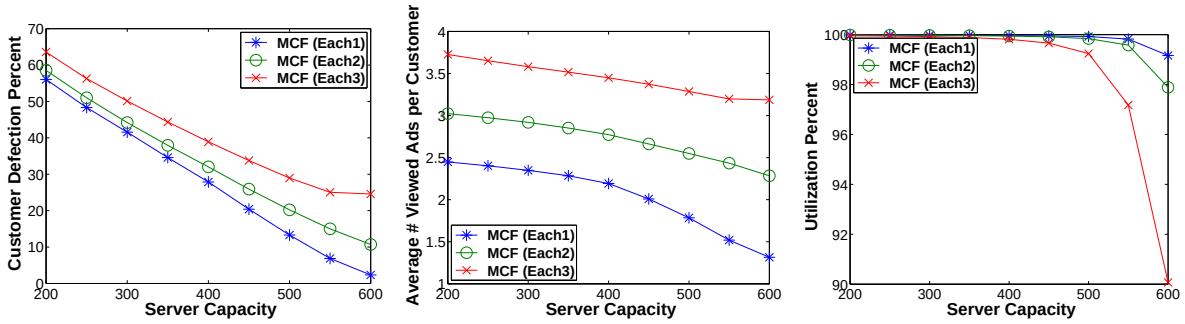
(a) ERMT (b) Patching

Figure 3.17: Comparing Effectiveness of New Scheduling Policies [Equation-Based with Function 1]



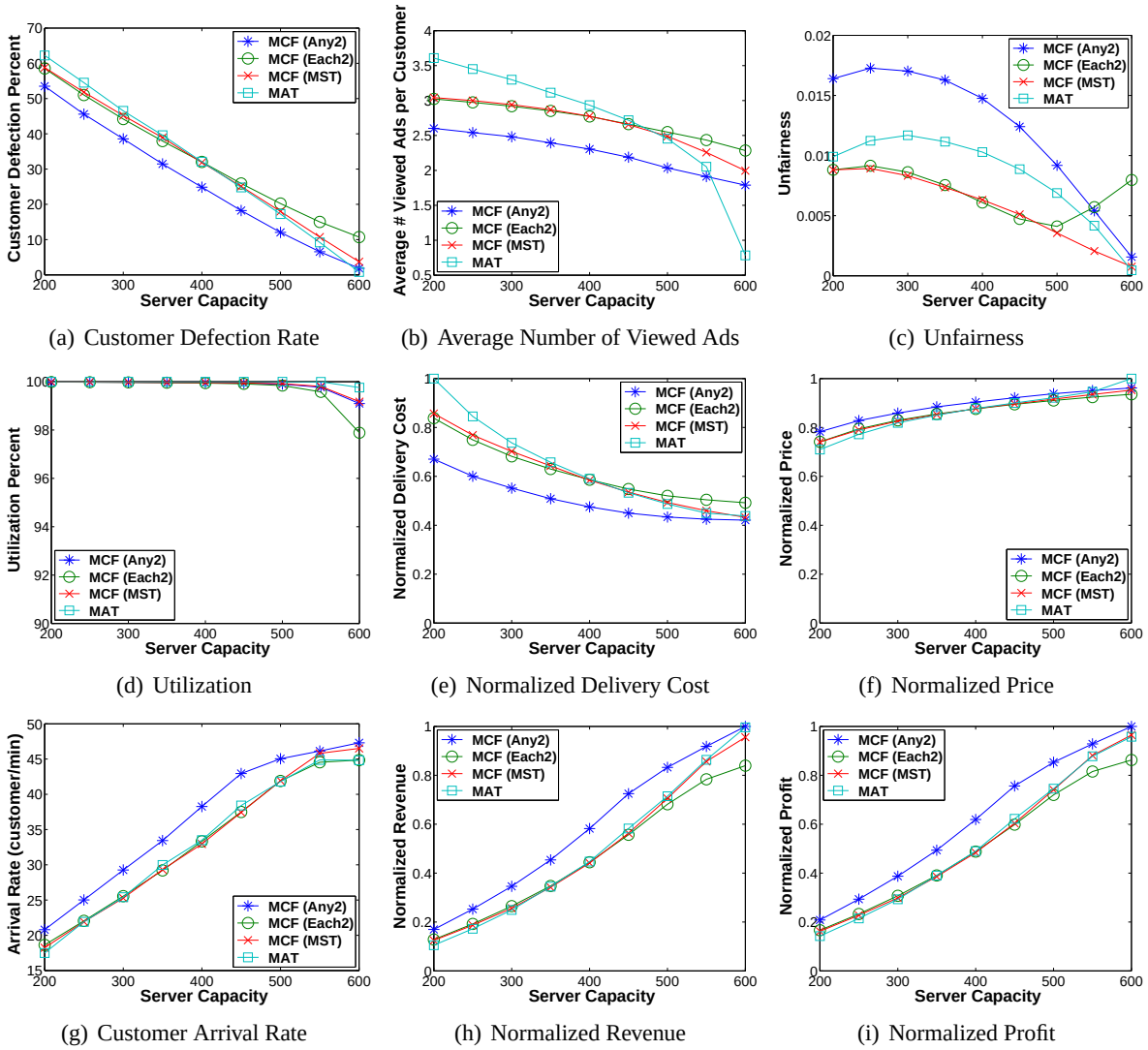
(a) Customer Defection Rate (b) Average Number of Viewed Ads (c) Utilization

Figure 3.18: Impact of Minimum Number of Ads on MCF (Any) [ERMT, Equation-Based with Function 1]



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

Figure 3.19: Impact of Minimum Number of Ads on MCF (Each) [ERMT, Equation-Based with Function 1]

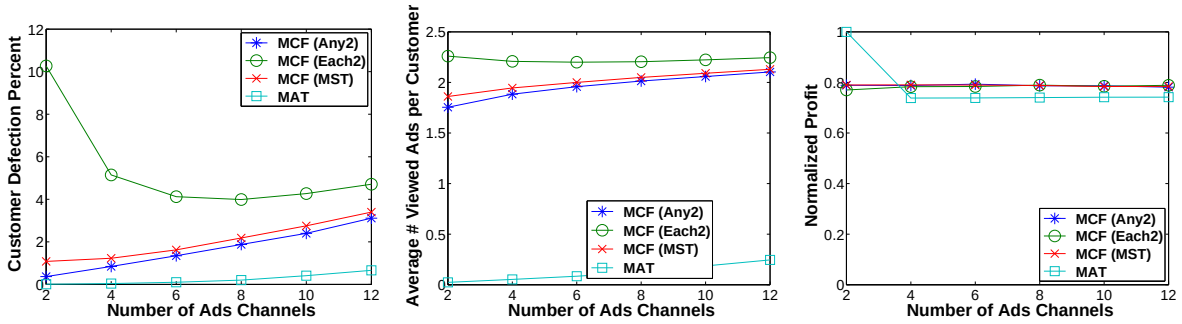


(a) Customer Defection Rate (b) Average Number of Viewed Ads (c) Unfairness

(d) Utilization (e) Normalized Delivery Cost (f) Normalized Price

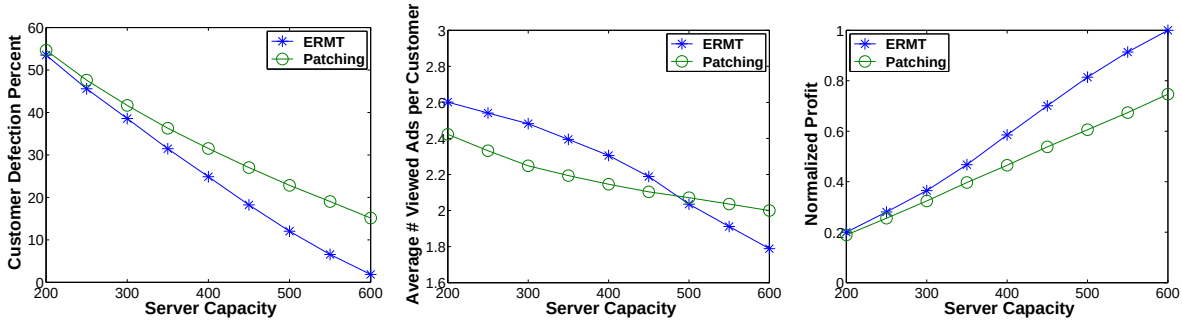
(g) Customer Arrival Rate (h) Normalized Revenue (i) Normalized Profit

Figure 3.20: Comparing Effectiveness of MCF-P (Any 2), MCF-P (Each 2), MCF-P (MST), and MAT [ERMT, Equation-Based with Function 1]



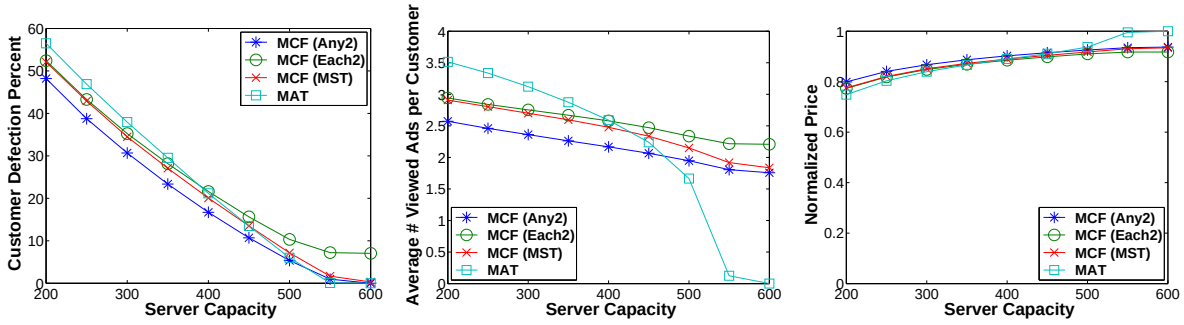
(a) Customer Defection Rate (b) Average Number of Viewed Ads (c) Normalized Profit

Figure 3.21: Impact of Number of Ad Channels [ERMT, Equation-Based with Function 1, Server Capacity = 500]

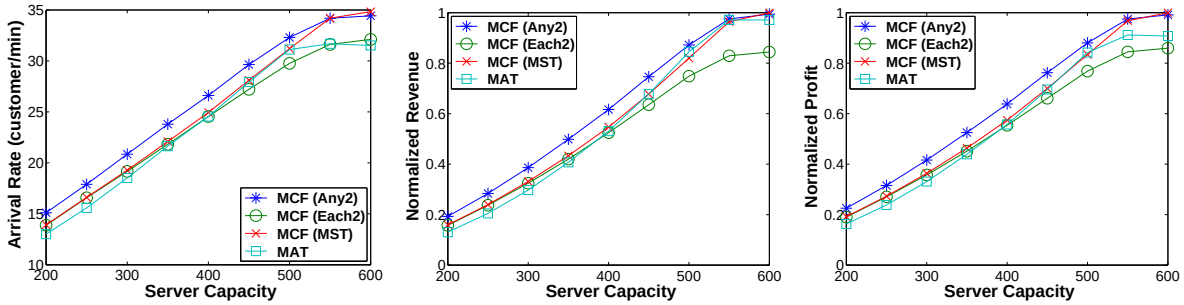


(a) Customer Defection Rate (b) Average Number of Viewed Ads (c) Normalized Profit

Figure 3.22: Comparing Effectiveness of Patching and ERMT [MCF-P (Any 2), Equation-Based with Function 1]

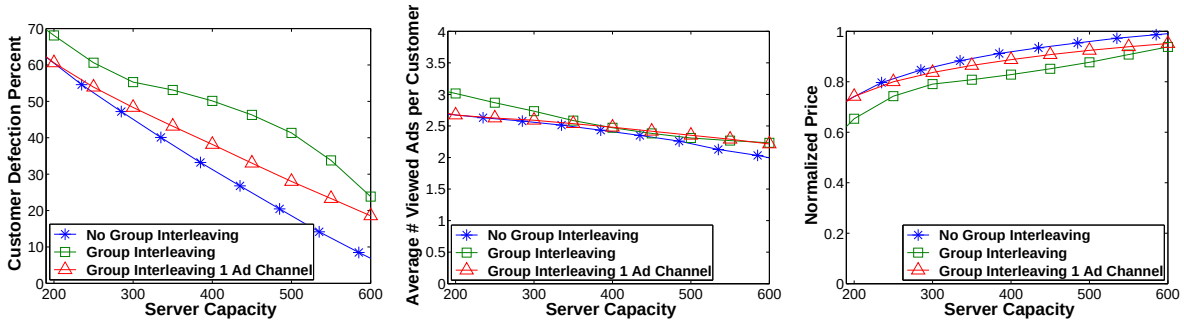


(a) Customer Defection Rate (b) Average Number of Viewed Ads (c) Normalized Price

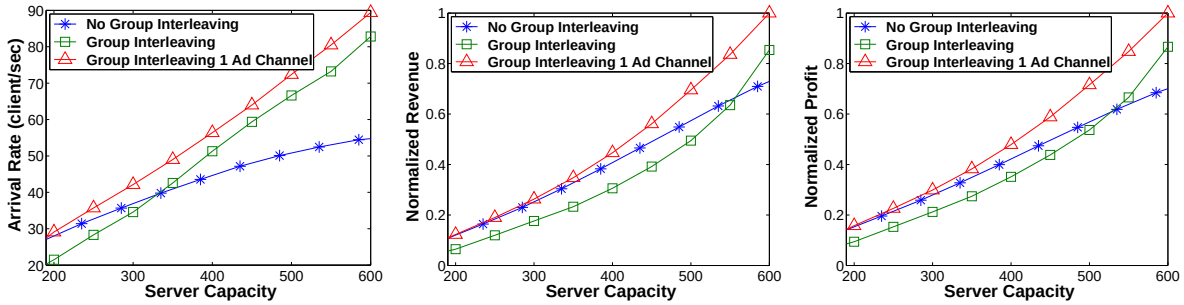


(d) Customer Arrival Rate (e) Normalized Revenue (f) Normalized Profit

Figure 3.23: Comparing Effectiveness of MCF-P (Any2), MCF-P (Each2), MAT and MST [ERMT, Equation-Based with Function 2]

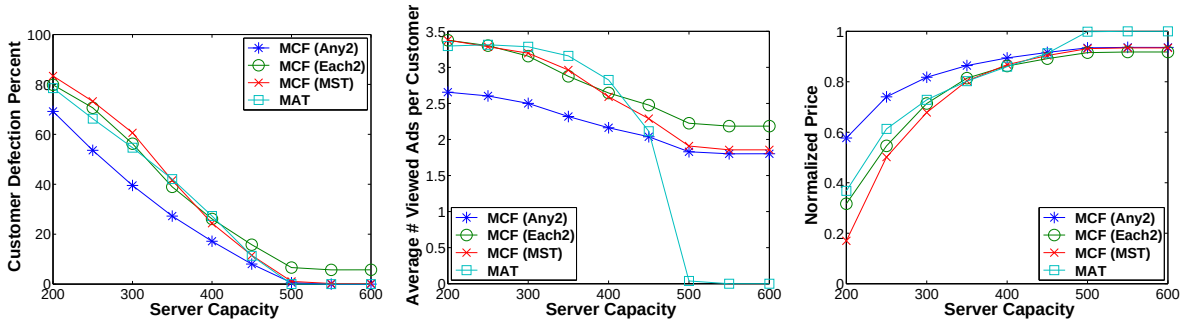


(a) Customer Defection Rate (b) Average Number of Viewed Ads (c) Normalized Price

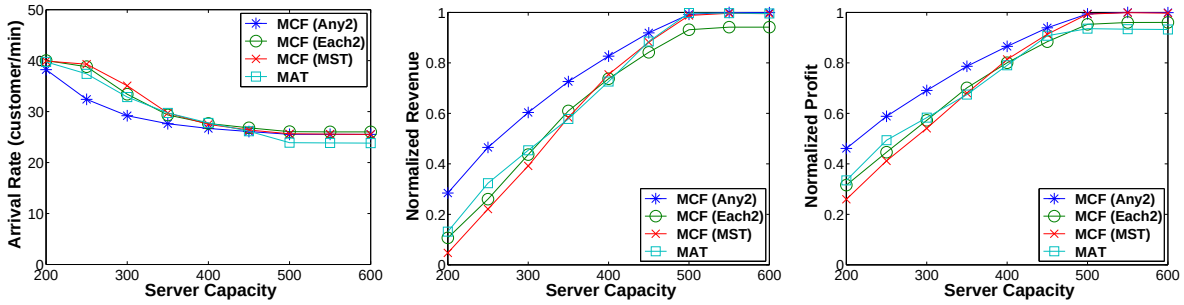


(d) Arrival Rate (e) Normalized Revenue (f) Normalized Profit

Figure 3.24: Comparing the Effectiveness of Targeted Ads Configurations [5x2, ERMT, MCF-P (Any2), Equation-Based with Function 1]

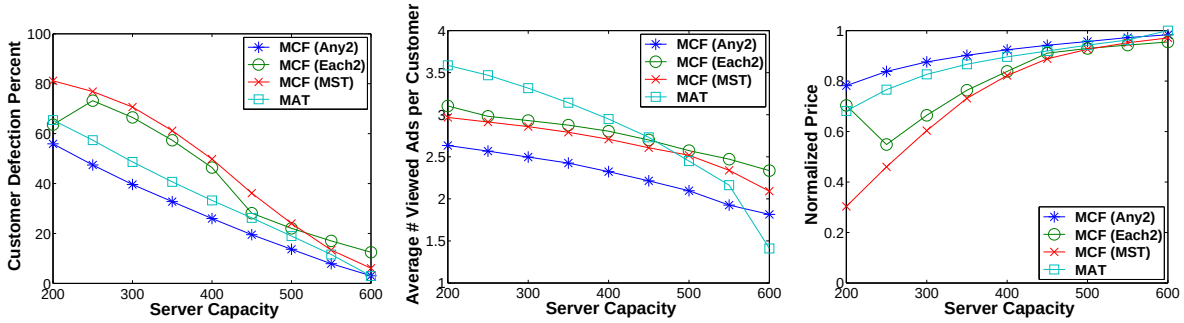


(a) Customer Defection Rate (b) Average Number of Viewed Ads (c) Normalized Price

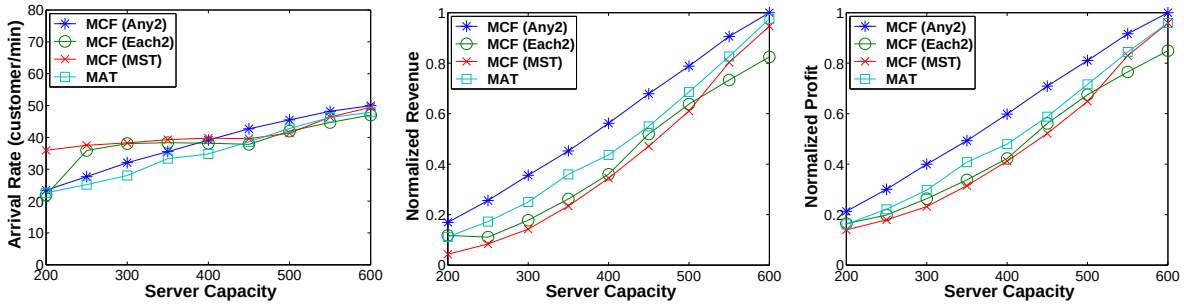


(d) Effective Arrival Rate (e) Normalized Revenue (f) Normalized Profit

Figure 3.25: Comparing Effectiveness of MCF-P (Any2), MCF-P (Each2), MAT and MST [ERMT, Willingness-Based Model]

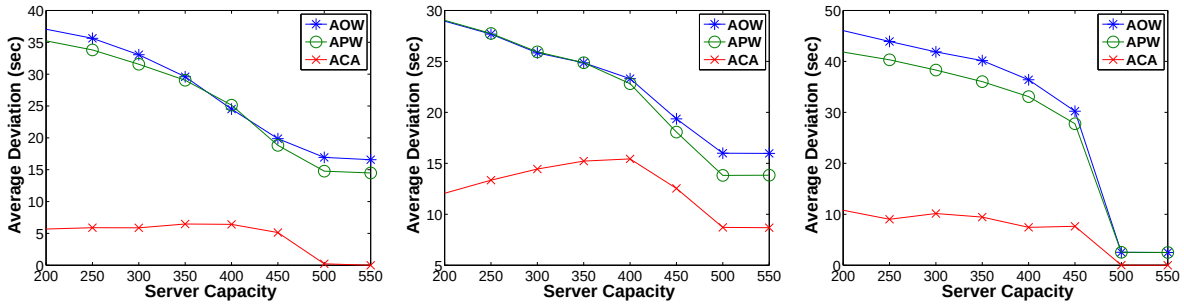


(a) Customer Defection Rate (b) Average Number of Viewed Ads (c) Normalized Price



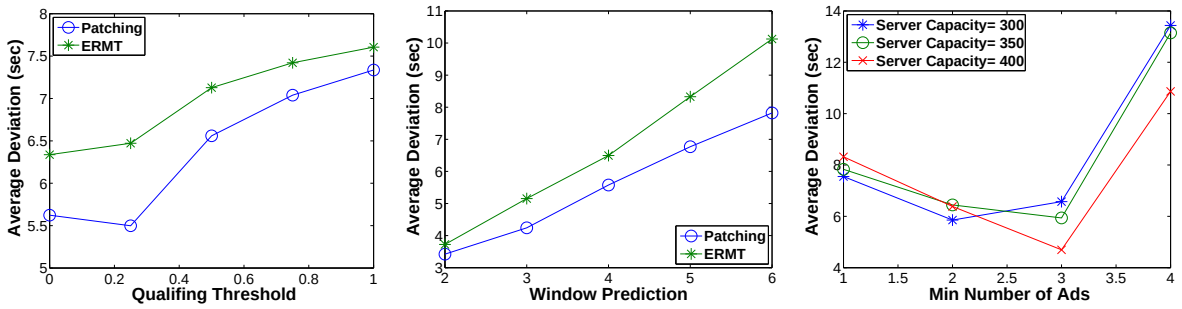
(d) Effective Arrival Rate (e) Normalized Revenue (f) Normalized Profit

Figure 3.26: Comparing Effectiveness of MCF-P (Any2), MCF-P (Each2), MAT and MST [ERMT, Hybrid Equation and Willingness Model]



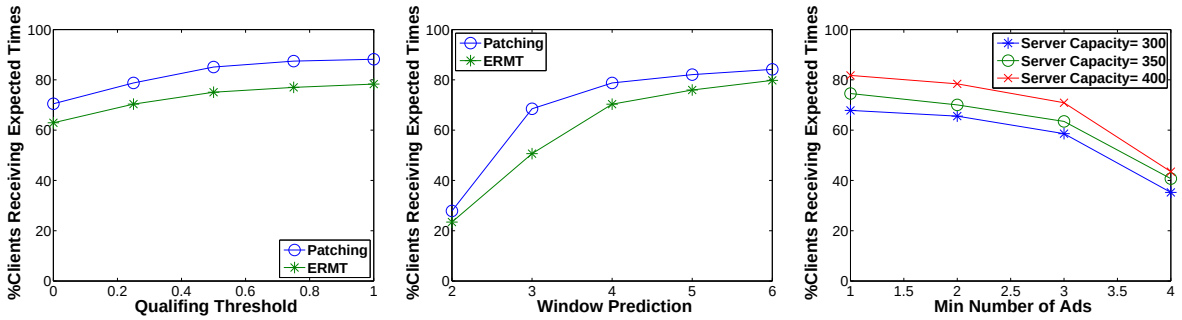
(a) Any 2 (b) Each 2 (c) MAT

Figure 3.27: Comparing the Effectiveness of Various Prediction Approaches [ERMT, MCF-P (Any2), Willingness-Based Model]



(a) Qualification Threshold (b) Prediction Window (c) Minimum Number of Ads

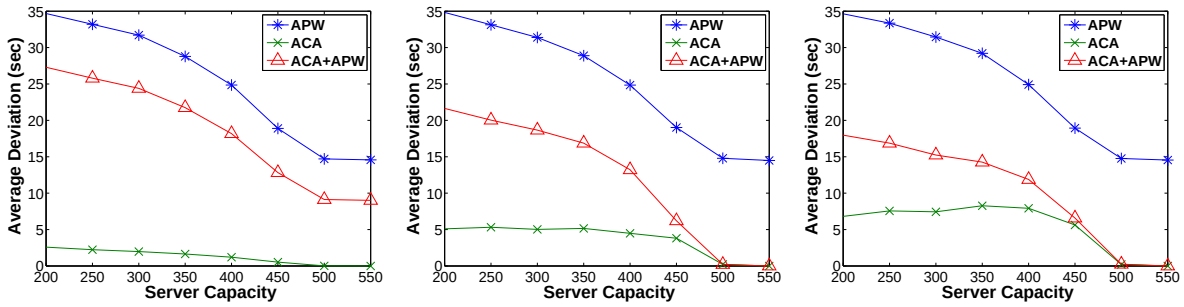
Figure 3.28: Impacts of Design Parameters on Prediction Accuracy [ACA, ERMT, MCF-P (Any2), Willingness-Based Model]



(a) Qualification Threshold (b) Prediction Window (c) Minimum Number of Ads

Figure 3.29: Impacts of Design Parameters on PCRE [ACA, ERMT, MCF-P (Any2), Willingness-Based Model]

Figure 4.14 compares the two alternative implementations of the prediction algorithm (ACA and ACA+APW) in terms of the average deviation. As mentioned earlier, ACA+APW gives expected times to all clients, whereas ACA cannot. The figure shows the average deviation results for three values of the prediction window (which does not affect APW). The figure demonstrates that the hybrid ACA+APW implementation has lower accuracy than ACA but higher than APW. The hybrid implementation is best used with a large value of the prediction window, so as to increase the percentage of clients receiving expected times by ACA (instead of APW).



(a) $W_p = 1$ (b) $W_p = 3$ (c) $W_p = 5$

Figure 3.30: Comparing the Two Alternative Implementations of the Prediction Algorithm [ERMT, MCF-P (Any2), Willingness-Based Model]

3.5.6 Analysis of Deviation Distributions

We so far compared various prediction schemes in only the average deviation. We discuss now the distributions of the deviation results, so that we can compare various schemes in the range, standard deviation (σ), and confidence interval (CI). Table 3.4 shows the means, standard deviations, and the 90% confidence intervals for various schemes. The results for ACA and ACA+APW are shown for three values of the prediction window. As expected, ACA provides the smallest standard deviation, and the shortest confidence interval, and these values increase with the prediction window. The hybrid scheme performs better than APW in terms of the standard deviation and the confidence interval but is worse than ACA. Note that the means of the distribution can be positive or negative. A negative value indicates a stronger negative deviation component, whereas a positive value indicates a stronger positive deviation component. A negative deviation means that a user waits shorter than expected, while a positive deviation means waiting longer than expected. It is possible to assign different weights for negative and positive deviations, but in this chapter we treat them equally.

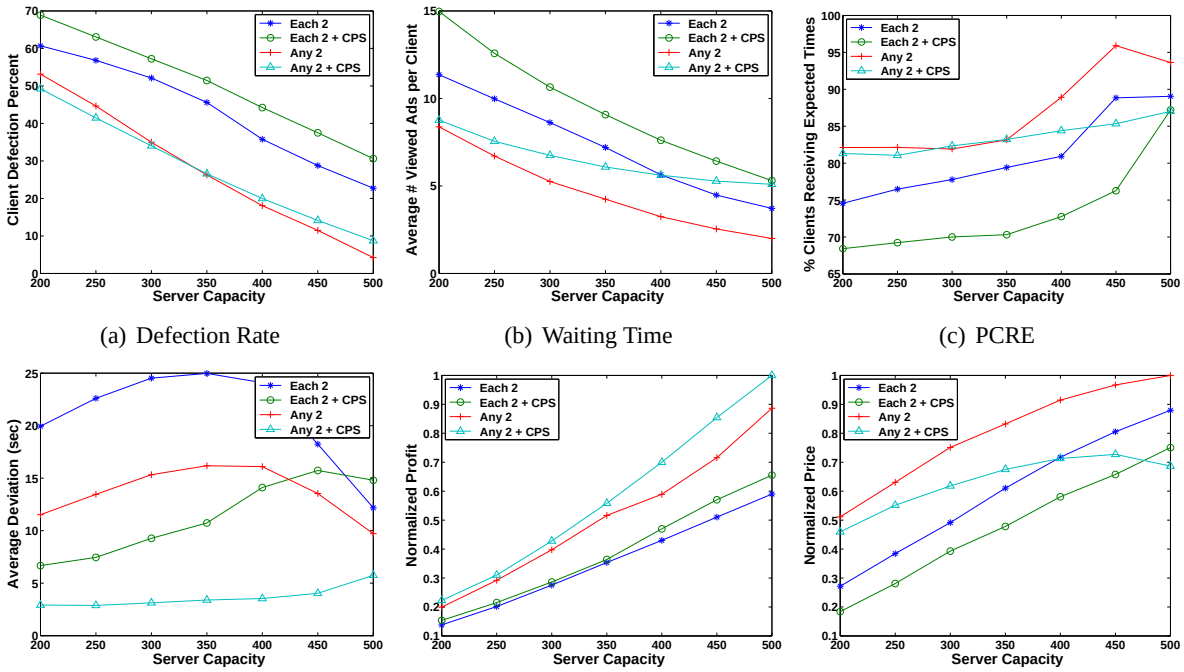
3.5.7 Effectiveness of the Proposed CPS Scheme

Figure 3.31 illustrates the effectiveness of CPS for “Any 2” and “Each 2” Scheduling with different server capacities. The results show that CPS may slightly increase the defection rate, but this is due

Table 3.4: Summary of Deviation Distributions [ERMT, Any 2, Model A, 350 Channels]

Scheme	**Mean** (Ads)	**Standard Dev.** (Ads)	90% **Confidence Interval** (Ads)
APW	0.0234	1.2909	[-3.1887,3.8113]
ACA, W_p=1	0.0852	0.4914	[0,1]
ACA, W_p=3	0.1193	0.6259	[0,1]
ACA, W_p=5	-0.0151	0.7601	[-1,1]
ACA+APW, W_p=1	0.3359	1.1843	[-2.0567,2.9433]
ACA+APW, W_p=3	0.5028	1.0609	[-2.243,2.757]
ACA+APW, W_p=5	0.2761	1.1038	[-2.0,3.0]

to the higher arrival rate achieved by accepting more client requests. 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 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%.



(a) Defection Rate (b) Waiting Time (c) PCRE

(d) Average Deviation (e) Profit (f) Price

Figure 3.31: Effectiveness of the Proposed CPS Scheme [ERMT, Hybrid Model]

3.6 Conclusions

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. Furthermore, we have proposed an accurate prediction algorithm for the scalable delivery framework.

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 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.

- The proposed waiting time prediction scheme, *Assign Closest Ad Completion Time* (ACA), is best to be combined with MCF-P (Any N) scheduling policy. The prediction accuracy in this case is within 25% of an ad length.

- When combined with ACA , MCF-P (Any N) also gives the best overall performance among all considered scheduling policies in terms of client defection probability, revenue, and profit.

- The prediction algorithm can give an expected waiting time to each client by using a hybrid implementation. In this case, the prediction window should be set to a large value to reduce the negative impact on accuracy. The accuracy remains within half an ad length for realistic server capacities.

CHAPTER 4

SCALABLE MULTI-CAMERA AUTOMATED SURVEILLANCE SYSTEM

In this chapter, we present a scalable automated watch-list-based surveillance system. We develop a solution that seeks to optimize the overall subject recognition probability by controlling the pan, tilt, and zoom of various deployed PTZ cameras, based on the characteristics of the subjects in the surveillance area. This chapter focuses primarily on face recognition.

The control of cameras is based on many factors, such as the direction of the subject's movement and its location, distance from the cameras, occlusion, overall recognition probability so far, and the expected time to leave the site, as well as the movements of cameras and their capabilities and limitations. Camera movement is broadly defined here as the time required to reach the assigned pan, tilt, zoom, and focus settings. None of the prior studies dealt with all of these influences. The developed solution works with realistic 3D environments and not just 2D scenes. The overall solution includes three new alternative schemes for scheduling the cameras: *Brute-Force Grouping* (BFG), *Grid-Based Grouping* (GBG), and *Elevator-Based Planning* (EBP). We analyze the effectiveness of the proposed solution and the alternative schemes through extensive simulation. We also enhance the system by applying a clustering mechanism. We analyze the impacts of the number of PTZ cameras, the subject arrival rate, the pre-recording interval length, the recording interval length, the PTZ movement step size, and the area of surveillance site.

The main contributions in this chapter can be summarized as follows.

- We propose an overall solution that optimizes the overall subject recognition probability, considering the impacts of a large number of influences on this probability, and incorporating a Watch List of subjects deemed dangerous.

- We address the camera scheduling problem in realistic 3D environments and develop and use a detailed and realistic simulator.

- We develop a detailed model of the overall recognition probability.
- We incorporate the practical limits and capabilities of the PTZ cameras.
- When considering the contribution of of successive camera frames on the overall recognition probability, the solution considers the *dependency* between these frames.
- We present three schemes for camera scheduling, including the highly efficient and novel EBP scheme.
- We study the effect of subject clustering on our system.

The rest of this chapter is organized as follows. Section 4.1 presents the proposed solution. Section 4.2 discusses the performance evaluation methodology. Section 4.3 presents the main results. And finally Section 4.4 draws the conclusion.

4.1 Proposed Solution

4.1.1 System Overview

The proposed AVS system employs wide-angle cameras, PTZ cameras, and a processing architecture. An oversimplified view of the system is shown in Figure 4.1. Wide-angle cameras are used to observe the site and send information to a processing system, which analyzes the scenes and then controls the PTZ cameras. The PTZ cameras capture higher quality frames of the subjects.

The processing architecture performs the following tasks.

- It analyzes the images of the wide-angle cameras to determine the locations and types of subjects and their speeds and directions of movement. This task utilizes the widely researched pedestrian detection algorithms [29] for determining the various attributes of subjects.
- It predicts the locations of the subjects when the cameras will start recording.

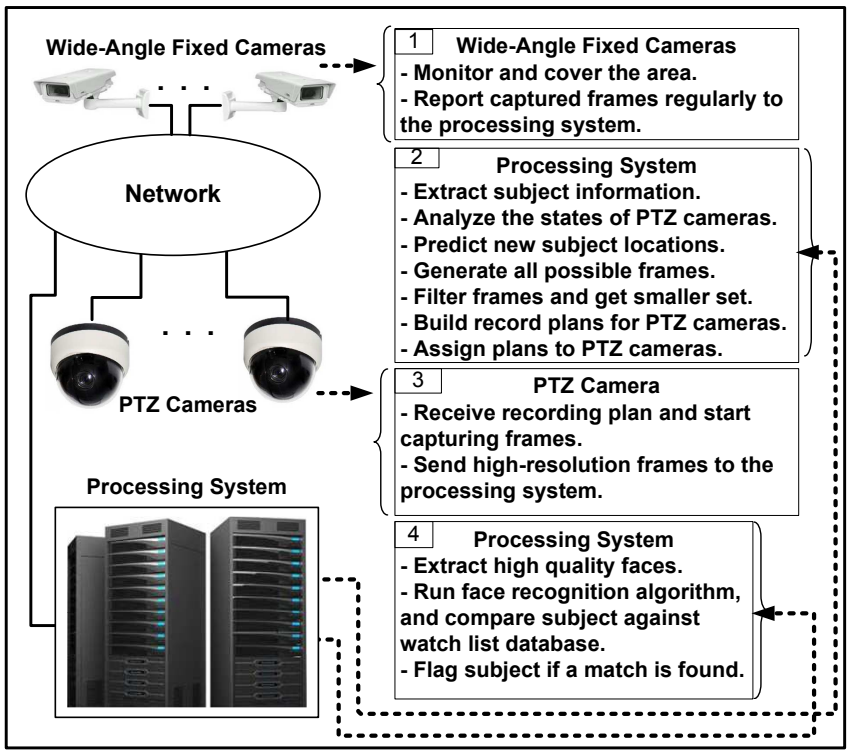


Figure 4.1: Automated Surveillance System Overview

- It runs a camera planning and assignment algorithm and controls the PTZ cameras. This task may include generating sets of possible frames according to the states of the PTZ camera and their capabilities and limitations.

- It runs the computer vision algorithms, such as face recognition.

The system operates into two alternating periods: a *pre-recording period* of T_p seconds and a *recording period* of T_R seconds, as in [31]. In the first period, the processing system reads the state and performs some prediction and algorithmic calculations. The PTZ camera frame assignment is also done in this period. In the second period, the processing system starts receiving high quality frames from the PTZ cameras and considers them for further processing. During the recording period, the cameras track the subjects.

The system employs a *Watch List*, which includes an image database of subjects that are deemed dangerous to the surveillance site.

4.1.2 Overall Solution Overview

The proposed solution targets the camera scheduling and assignment problem in 3D environments. Its main objective is to optimize the overall subject recognition probability by controlling the pan, tilt, and zoom of various deployed PTZ cameras, based on the characteristics of the subjects in the surveillance area. This control of cameras is based on the direction of the subject's movement and its location, distance from the cameras, occlusion, overall recognition probability so far, and the expected time to leave the site, as well as the movements of cameras and their capabilities and limitations.

As shown in Figure 4.2, the solution consists of four phases.

- *Frame Generation*– In this phase, the processing system utilizes the information provided by the fixed wide-angle cameras to detect all subjects captured by these cameras and determine their attributes, including, location, speed, and direction of movement. The attributes can be determined using one of the widely researched pedestrian detection algorithms [29]. Subsequently, the processing system predicts the locations of the subjects at the future time when the PTZ cameras enter the recording and tracking phase. The processing system then uses the prediction results to generate the set of all possible frames that can be captured by each PTZ camera through the examination of the different combinations of the camera's settings. This last step utilizes the capabilities and limitations of the PTZ cameras. Note that each PTZ camera can capture a different frame at each camera setting. The frame here is basically the *projection* of the view as seen by the camera at a specific FoV. By examining different settings, a set of frames encompassing the entire FoR will be generated. Combining all possible frames from all cameras produces the entire frame domain that will need to be analyzed in later phases. In this stage, a projection triangulation is used to estimate the 2D frame characteristics from the 3D site. Rotation and translation matrices are used to map the 3D coordinates to each camera coordinates [60, 61].

- *Frame Filtration*– After generating the sets of all possible frames, the processing system filters

these frames by eliminating the frames that do not capture any subject in the surveillance site and by selecting only the best camera frame among the sets of frames that capture exactly the same set of subjects. The selected frame is the one that achieves the maximum aggregate recognition probability of all subjects.

- *PTZ Camera Scheduling*– In this phase, the processing system carries out camera scheduling and frame assignment. The proposed schemes for this task are detailed in Subsection 4.1.5.

- *Recording and Tracking*– In this phase, the PTZ cameras apply the settings of camera planning and scheduling to capture the target subjects and start recording and tracking these subjects.

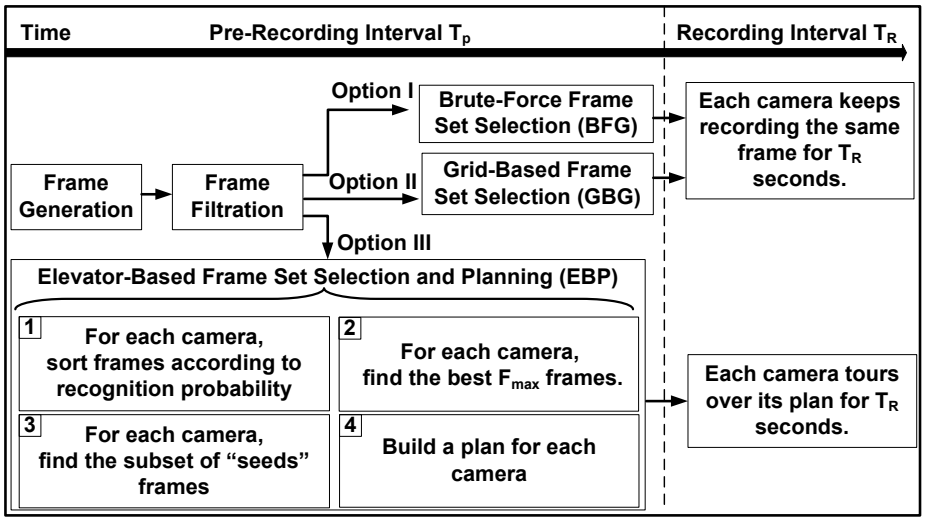


Figure 4.2: Illustration of the Proposed Solution

4.1.3 Formulation of the Objective Function

Since the main objective of the proposed solution is to optimize the overall recognition probability, we need to formulate this probability as an objective function. In this chapter, we focus primarily on face recognition.

Let us now discuss how the recognition problem can be formulated. Face recognition has been a challenging task. Face recognition algorithms demand images with higher quality compared with other

computer vision applications, such as detection and tracking. Many face recognition algorithms [53] and applications [54, 55] are designed to deal with one or more complications in the captured faces, such as the resolution, pose, and occlusion. The camera's zoom should be adjusted to achieve a minimum resolution, required by most face recognition algorithms. In particular, the inter-ocular (between eye pupils) distance should be 60 to 120 pixels at minimum [62, 63]. To keep this resolution in a dynamic scene, the camera's zoom should be adjusted according to the subject distance. Far subjects require higher zooms, leading to smaller FoV and thus a smaller number of covered subjects in the frame. The time spent on a specific frame should be selected carefully. If the camera spends a longer time on one frame, the covered subjects will have better chances for recognition, but this may prevent other subjects from being detected and recognized.

Let us now discuss various functions that can be used to model the impacts of various factors on the recognition probability. Zooming-in especially for far distances introduces a blurring effect noise, thereby reducing the recognition probability. Function $R_{zoom}(z_{ij})$ can be derived to capture the relationship between the recognition probability R and zoom z_{ij}, where z_{ij} is the zoom adopted by camera j to capture subject i, and produce a resolution of at least 60 inter-ocular pixels. Pose variation highly affects the recognition probability. Subjects with poses more than 25°are poorly recognized using normal recognition algorithms. To address this problem, many pose specific face recognition algorithms were proposed. These algorithms need more computational power and longer execution time to operate and they perform well only in controlled environments. We will limit poses to $\pm$25°pan or tilt boundaries. Function $R_{pose}(\theta_{ij})$ can be derived to capture the relationship between the recognition probability R and pose θ_{ij}, where θ_{ij} is the angle between subject i (face center) and the optical axis of camera j. Since a subject may be out of a camera's FoV or occluded by structures or other subjects with respect to a certain camera's settings (frame), functions $u_{cov}(i, j)$ and $u_{occ}(i, j)$ can be used to assess the coverage

and occlusion of the subject, respectively:

$$u_{cov}(i, j) = \begin{cases} 1 & \text{if subject } i \text{ is fully covered by frame } j \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

$$u_{occ}(i, j) = \begin{cases} 1 & \text{if subject } i \text{ is not occluded in frame } j \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

We form a utility function $\psi(j)$ to measure the total recognition probability of all subjects included in camera frame j:

$$\psi(j) = \sum_{i=1}^{N_s} \gamma(i, j), \tag{4.3}$$

where N_s is number of subjects in frame j, and $\gamma(i, j)$ is the recognition probability of subject i captured in camera frame j. This equation sums the recognition probability value contributed by each covered subject i in frame j. The recognition probability $\gamma(i, j)$ of subject i captured in camera frame j can be formulated as follows:

$$\gamma(i, j) = W(i) \times R_{pose}(\theta_{tij}) \times R_{pose}(\theta_{pij}) \times R_{zoom}(z_{ij}) \times u_{cov}(i, j) \times u_{occ}(i, j), \tag{4.4}$$

where $W(i)$ is a weighting factor. The weight depends on the expected time for the subject to leave the site, and the overall recognition probability $\Gamma(i)$ of the subject so far:

$$W(i) = \begin{cases} b_3^{(T_{leave} - T_{now})} \times (R_{thresh} - \Gamma(i)) & \Gamma(i) \leq R_{thresh} \\ 0 & \Gamma(i) > R_{thresh}, \end{cases} \tag{4.5}$$

where, b_3 is a constant, T_{leave} is the subject's expected time to leave the site, T_{now} is current time, $\Gamma(i)$ is the overall recognition probability of the subject so far, and R_{thresh} is the threshold after which

the subject is considered to be sufficiently recognized. The use of time-to-leave is adopted from the weighting strategies in [27, 30, 31]. Note that after a subject is considered sufficiently recognized the weight becomes zero and further action may be required whether the subject is recognized to be in the Watch List or Trusted List. If the subject is in the Trusted List, the system may not need to devote any resources on the subject any more. If it is in the Watch List, then the system should notify the administrators and provide the ability to track the subject. The subject may no longer be considered for recognition purposes but becomes a high priority for other operations, such as tracking.

The overall recognition probability of the subject so far can be given as follows.

$$\Gamma(i) = 1 - \prod_{j=1}^{S_{i_T}} (1 - \bar{\gamma}(i, j)), \tag{4.6}$$

where S_{i_T} is the number of times subject i is captured, and $\bar{\gamma}(i, j)$ is an unweighted version of the $\gamma(i, j)$ defined in Equation 4.4. We can see from Equation 4.5 that as $\Gamma(i)$ increases the weight decreases, and the sooner the subject leaves the area, the higher its weight becomes.

When computing the overall recognition probability, the system must consider the *dependency* between successive frames of the camera. Successive frames may not be fully independent if they are captured by the same camera with almost the same settings. If the frames are very close to one another in time, then the capturing of the second frame may not help increase the recognition probability of the subject, since almost the same picture of the subject is taken as the subject may not move much and his/her face expression and eye status (open, close, or partially open/close) may not significantly change. Up to our knowledge, this issue was not addressed at all in prior studies. We introduce a parameter, called *dependency period* (T_{dep}). All frames captured within this period by the same camera with the same settings are considered dependent, and thus only one of them can contribute to recognition probability.

4.1.4 Modeling Individual Factors on Objective Function

Let us now discuss the modeling of various factors on recognition probability. From the empirical data in [52], we can model the zoom as a linear function of the distance as follows:

$$Zoom(x) = a_0 \times x + b_0, \tag{4.7}$$

where a_0 and b_0 are constants, and x is the distance between a PTZ camera and the subject.

Based on the empirical data from [52], we derive the following model for the relationship between the recognition probability R and the zoom z:

$$R_{zoom}(z_{ij}) = a_1 \times e^{(b_1 \times z_{ij})}, \tag{4.8}$$

where z_{ij} is the zoom adopted by camera j to capture subject i with a resolution of 60 inter-ocular pixels, and a_1 and b_1 are constants.

Based on the empirical data in [50], which were verified by [28], and by limiting the poses to $\pm$25°pan or tilt boundaries, we derive the following model for the impact of pose on recognition probability:

$$R_{pose}(\theta_{ij}) = \begin{cases} e^{(-\frac{\theta_{ij}}{b_2})^2} & |\theta_{ij}| \leq 25 \\ 0 & \text{otherwise,} \end{cases} \tag{4.9}$$

where θ_{ij} is the angle between subject i (face center) and the optical axis of camera j, b_2 is constant. Same function is used for pan and tilt angles.

Figures 4.3, 4.4, and 4.5 demonstrate various models.

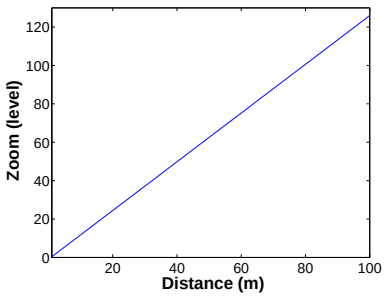


Figure 4.3: Zoom needed to achieve [60-80] pixel inter-ocular distance at each distance

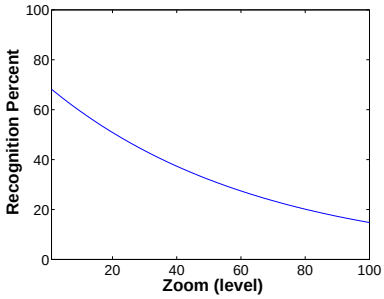


Figure 4.4: Zoom effect on recognition, [inter-ocular distance = 60-80 pixel]

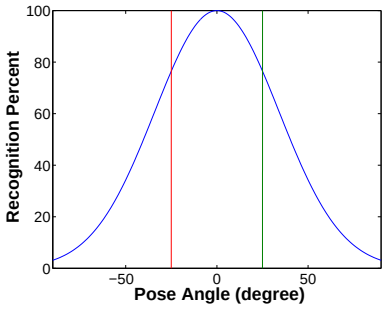


Figure 4.5: Face pose effect on recognition

4.1.5 PTZ Camera Scheduling

Camera scheduling is the core of our proposed solution. We present three schemes for camera scheduling: *Brute Force Grouping* (BFG), *Grid-Based Grouping* (GBG), and *Elevator-Based Planning* (EBP). BFG scans the filtered frame list from all C PTZ cameras and chooses a set of C frames, including one frame from each camera. This set must achieve the highest aggregate recognition probability of all subjects. This probability is measured by considering the recognition probability of each subject only once at its maximum occurrence in the set regardless of its number of occurrences. GBG is an adaptation of the algorithm in [31]. The work in [31] assumed that all cameras capture the same set of frames in the same fashion. In particular, a frame can be captured by any PTZ camera located at any position in the scene, and it will have the same "satisfaction value". It also assumed a frame domain derived from 2D scenes and not realistic 3D environments. We enhance the algorithm in the following aspects:

- dealing with cameras having different FoRs,
- generating frames according to cameras capabilities and not site area dimensions,
- grouping based on 3D scene by considering each camera's FoV and the locations of subjects,
- and assessing the overlap among frames in 3D scenes.

EBP refines the filtered frames and uses them to build detailed plans for each camera. The scheme captures the current settings of each camera, and its moving, zooming, and focusing speeds. In contrast with BFG and GBG, EBP allows each camera to view a different set of subjects at different times during one recording period rather than repeatedly tracking the same subjects.

Let us now discuss the proposed EBP scheme in more detail. As shown in Figure 4.2, this scheme proceeds into two main stages:

- generating plan seeds for each PTZ camera,
- and building the plan for each PTZ camera from these seeds.

The generation of the plans seeds encompasses Steps 1 to 4 in Figure 4.2. A simplified algorithm for seed generation is shown in Figure 4.6. The algorithm sorts frames in each camera list according to their overall recognition probabilities ψ and picks the best F_{max} frames for each camera to reduce the complexity of subsequent steps. F_{max} can be selected such that the maximum number of frames for each camera is considered within an implementation overhead of T_p seconds for camera scheduling. For each camera, out of the F_{max} frames, the algorithm selects a subset of $seeds$ frames (where $seeds < F_{max}$) such that this subset achieves the maximum aggregate recognition probability ($MARP$) of all subjects. This probability is measured as follows by considering the recognition probability of each subject only once at its maximum occurrence in the subset regardless of its number of occurrences:

$$MARP = \sum_{i}^{S_N} \sum_{j}^{Seeds} (\bar{\gamma}(i, j) \times u(i)_{max}), \tag{4.10}$$

where $\bar{\gamma}(i, j)$ is the unweighted version of the $\gamma(i, j)$ defined in Equation 4.4, $seeds$ is the number of frames in the subset, S_N is the number of subjects in the subset, and $u(i)_{max}$ for subject i is 1 only when it has the highest recognition probability ($\bar{\gamma}$) and 0 otherwise. The subset of frames is stored in $planSeeds_{arr}$.

00. *// Input: Filtered Frames Domain*
00. *// Output:* $planSeeds_{arr}$ *list of best frame sets*
01. **chooseFrames()**{
02. Sort each camera's frames according to their recognition probability ψ;
03. Get the best F_{max} frames for each camera;
04. Generate all $\begin{pmatrix} F_{max} \\ seeds \end{pmatrix}$ frame sets combinations;
05. Compute the aggregate recognition probability $MARP$ for each set combination per each PTZ camera;
06. Choose the best set of seeds for each PTZ camera and store them in $planSeeds_{arr}$;
07. } *// end chooseFrames()*

Figure 4.6: A Simplified Algorithm to Generate Seeds-Frames for Building Plans

The second stage of the EBP scheme generates the plan list for each camera based on the selected subset of frames ($planSeeds_{arr}$) in the previous stage. Figure 4.7 shows a simplified algorithm. The algorithm builds the plan for each PTZ camera individually. For each PTZ camera, it investigates its $planSeeds_{arr}$ and chooses the best reachable subset of frames, with possible duplications if necessary, during the recording period. It also considers the required time for the camera to move to a new frame. This time, broadly defined here as the time for the camera to achieve a specified particular pan, tilt, zoom, and focus setting, can be given by $T_{move} = PTZtime + FocusTime$. The algorithm also considers the recording time T_i for that candidate frame. These frames in $planSeeds_{arr}$ are ordered in terms of the required camera movement time. The algorithm then scans these frames in an elevator-like pattern, in the order of camera movement time. It starts moving from one frame to the next until reaching one end, and then moves backward, and repeats the cycle if necessary as long as the time of the recording period permits. The algorithm computes the best frame to start with by examining different plans generated from different starting points and selecting the plan with the maximum recording probability ($planRecProb$). Figure 4.8 illustrates by an example the process of building a plan using the elevator-based technique. The figure shows a set of four frames. Frame 1 is skipped because it is not the best candidate to cover by the current PTZ camera settings. The system continues building the plan

as long as the remaining time in the recording interval T_R permits. It traverses the frames from one end to the other until it stops at frame 3 because that frame is not reachable during the remaining time. This process is repeated for different starting points and all generated plans are compared. Finally, the best plan is chosen and assigned to the PTZ camera.

4.1.6 Frame Population Using Clustering

In order to populate frames in a more efficient way, we utilize the clustering as a pre-step to generate frames. The main advantage of the clustering process is that it enables us to focus on the areas that are populated with subjects. The problem of grouping subjects into clusters can be performed by adopting one of the widely investigated clustering algorithms [64, 65]. Subjects are grouped based on many attributes, which can be translated eventually into distance. In our work, a cluster is defined by three attributes: (1) cluster center, (2) cluster direction and (3) cluster size. To determine whether a subject belongs to a cluster, we examine three parameters and check whether their values are lower than the allowed maximum values. These maximum values are the following: (1) the maximum allowed distance between the cluster center and the subject $DIST\_TH$, (2) the maximum subject angle-offset from the cluster-center direction of motion $ANGLE\_TH$, and (3) the maximum perpendicular-distance between the subject and the cluster direction of motion $WIDTH\_TH$.

K-Means is one of the well known clustering methods. In K-Means, N entities need to be mapped to K centers according to certain conditions quantified as a distance. The standard Lloyad solution to the K-Means problem involves two major steps: (1) assignment, in which we find the nearest cluster center and assign the node to it, and (2) computing the new cluster center. In our clustering problem, the number and the locations of the K centers are unknown. In order to deal with that, we use a variation of the dendrogram hierarchical clustering method [66]. In this method, each entity is considered as a cluster center at the beginning. Then each pair of nodes with a minimum in-between distance are grouped together to form one cluster. We calculate the centers for the newly formed clusters and repeat

the first step. Figure 4.9 explains the dendrogram algorithm. Lines 11,13 and 14 calculate the Cartesian distance, the perpendicular distance, and the angle difference, respectively.

Figure 4.10 shows a simple illustration for cluster formation. Cluster $R1$ is initially formed, then another subject at location A is added to form cluster $R2$. XA and XB represent the Cartesian distance, where XC and XD are the perpendicular distance between cluster $R1$ center and the subjects at location A and location B, respectively. Subjects at location F and location G cannot be added to the formed cluster. The directions of movement for these subjects are more than the $ANGLE\_TH$. Subject at location B cannot be added too; because it has a perpendicular distance XD that is bigger than $WIDTH\_TH$. After we find the clusters, we start populating frames as in Figure 4.11. We pick the centers of the clusters and identify which camera can cover them with the best overall recognition probability. We also re-map the subjects in the provided clusters such that a more precise clusters are formed. By re-mapping, we do not change the centers of the clusters. We also test whether a subject belongs to a certain cluster center with respect to the *current camera settings (Pan-Tilt-Zoom)*. Moreover, sub-clusters are also generated from the same centers. These sub-clusters are smaller than the maximum cluster size but have a higher resolution.

After generating the frames using the clustering process, we apply the previously introduced algorithms GBG and EBP. To distinguish the new approach from the previously defined one, we rename the algorithms as *GBG with Clustering* (GBGC) and *EBP with Clustering* (EBPC). Furthermore, we have applied an enhancement to reduce the search domain of the new algorithms. For GBG and GBGC, we can enhance the filtration by excluding all the frames that are not reachable due to the long time of computation, camera movement and focus time. We call this feature the reachability enhancement. In the case of EBP and EBPC, we cannot determine the reachability before hand as the location of the camera with respect to the frame is not known until we compute the different plans. As a result, EBP and EBPC after enhancement need longer computation time than GBG and GBGC, respectively.

4.2 Performance and Evaluation

We develop in $C++$ a simulator, called *AutoSurvSim*, for an AVS system to support the proposed solution for controlling the PTZ cameras. Without loss of generality, we assume a surveillance site with a rectangular area. Subjects arrive randomly from any of the four directions and enter the site with a random direction and a random speed. Poisson distribution is used to model subject arrivals, while a truncated Gaussian distribution is used for the speed, and a truncated uniform distribution is used for the direction [31]. The PTZ cameras are located at the perimeter of the site with different (x,y,z) locations. Moreover, a detailed characterization for each camera is implemented, including (Pan-Tilt-Zoom) speeds with maximum and minimum limits and a step size, refocusing speed, and sensor dimensions. Furthermore, we consider the impact of frame dependence as discussed in Subsection 4.1.3. The results are collected by running the simulator on a system with 32-bit 2.4 GHz dual-core CPU and 2.5 GB RAM. The simulations are run for the time required to process 50000 subjects.

The performance metrics are the average subject recognition probability, the percentage of subjects covered/captured at least one, and the algorithms computation time. We analyze the impacts of the number of PTZ cameras, the arrival rate, the pre-recording interval length, the recording interval length, the PTZ movement step size, and the area of surveillance site. Moreover we adopt a frame independence policy to exclude the effect of watching the same frame in short period of time. Table 4.1 shows the main environment parameters and their values.

We conduct two set of experiments. The first one is to compare the three algorithms BFG, GBG and EBP on the same proposed framework. In this set, we show the results of the proposed algorithms without applying the clustering process and without the reachability enhancement. In the second set of experiments, we apply the clustering and the reachability enhancement then compare the two algorithms before and after the clustering, GBG and EBP with GBGC and EBPC, respectively. In the second set of results, we use arrival rates up to 8 subjects per second. We also reduce the default pre-recording period

time and recording period time to 2 seconds instead of 5 seconds. Table 4.2 shows the pre-clustering related parameters.

Table 4.1: Summary of Workload Characteristics [AVS, General]

Parameter	**Model/Value(s)**
Scene Area	Variable, Default = 80 × 60 m^2
Request Arrival	Poisson Process
Request Arrival Rate	Variable [0.1-1.0], Default = 0.5 Req/second
Request Speed	Truncated Normal Distribution [0.5-2.5]
Request Speed rate	0.5 m/sec
Request Direction	Truncated Uniform Distribution [-40°40°] degree with ⊥ to entrance side
PTZ Cameras	Variable [2-8], Default = 4
PTZ Pan	Variable [0-180] degrees
PTZ Tilt	Variable [0-90] degrees
PTZ Zoom	Variable [1-50] levels
Sensor Resolution	[640 × 480] pixels
Inter-Ocular Distance	60 pixels
Focus Time	0.5 second
T_p, T_r	Variable [1-10] second, Default = 5
$PTZStep$	Variable [3-6] Step, Default = 4
R_{thresh}	0.9
Eq 4.7 constants a_0,b_0	1.269 , -0.909
Eq 4.8 constants a_1,b_1	0.693 , -0.0154
Eq 4.9, Eq 4.5 constants b_2,b_2	48.31, 0.9
T_{dep}	0.5 second

Table 4.2: Summary of Workload Characteristics [AVS, Clustering]

Parameter	**Model/Value(s)**
Request Arrival Rate	Variable [1-8.0], Default = 8.0 Req/second
Sensor Resolution	[1024 × 768] pixels
Cluster Length	Variable, Default=20m
Cluster Width	Variable, Default=5m
Angle Threshold	25 degrees
T_p, T_r	Variable [1-5] second, Default = 2

4.3 Result Presentation and Analysis

4.3.1 Comparing Algorithms without Clustering

Effect of Number of PTZ Cameras

Let us now start by studying the effect of changing the number of PTZ cameras on the system performance and time complexity. Figure 4.12 depicts the effect of increasing the number of cameras in

a scene when using different scheduling schemes. The percentage of covered subjects and the average subject recognition probability increase when the number of PTZ cameras increases. This is mainly because more subjects are being captured before leaving the site, and the number of different captures for one subject increases too. However, the time complexity for the system also becomes higher. Increasing the number of cameras leads to a bigger search domain. EBP shows the best gain in performance with the least increase in time complexity. As a result, it is better to have more PTZ cameras given that the time complexity is acceptable and the cameras are allocated efficiently.

Effect of Subject Arrival Rate

The effectiveness of various scheduling schemes in dealing with scalable workloads is shown in Figure 4.13. The figure compares the three schemes under different subject arrival rates. As expected, the three metrics become worse as the arrival rate is increased because more subjects leave the surveillance site without being captured, less time is spent on the captured subjects, and the search space becomes larger, respectively. These results demonstrate that EBP handles higher arrival rates much better than BFG and GBG, and that the gaps in performance and time complexity between EBP and the other two schemes become wider as the arrival rates increases. Note that EBP captures more subjects during a recording period, whereas GBG and BFG need to search the surveillance site each time to capture new subjects, causing them to degrade much faster than EBP.

Effect of Pre-Recording Time and Recording Time Interval Lengths

The impact of the design parameters pre-recording interval T_p and the recording interval T_R are shown in Figures 4.14 and 4.15. Figures 4.14(a) and 4.14(b) explain the effect of T_p on the percentage of covered subjects and average subject recognition probability. It shows a degradation in both metrics when T_p gets longer due to (1) more subjects leaving the site during the pre-recording time, (2) subject state prediction over long intervals becoming obsolete, and (3) less time being spent on recording in a

bounded overall processing time. The same behavior is noticed for T_R intervals. The more time is spent recording the same frame, the fewer independent frames are captured, and the least covered subject percentage and average subject recognition probability are achieved, as shown in Figures 4.15(a) and 4.15(b).

Effect of Pant Tilt Zoom Stepping

The change in PTZ camera settings (pan angle, tilt angle, and zoom level) can be performed in a fine step size, or a coarse step size. As the step size gets smaller, we can generate more frames and choose better frame sets from a larger domain. Figures 4.16(a) and 4.16(b) show some improvement when the step is smaller. Figure 4.16(c) shows that the time complexity increases as the step size decreases since the frame generation, filtration and search take longer time.

Impact of Scene Area

We also investigate the scalability of scheduling schemes as the area of the surveillance system increases. Figures 4.17(a) and 4.17(b) show the effect of the area. The covered subjects percentage and the average recognition subject probability decrease with the area. The limited FoR of the PTZ cameras will not be able to cover the area as it grows. In addition, the algorithm time complexity increases since the life time of the subjects in the scene becomes longer, as shown in Figure 4.17(c).

4.3.2 Studying the Effect of Clustering

Effect of Number of PTZ Cameras

Figure 4.18(a) shows that when we increase the number of cameras in a scene, the covered subjects percentage increases. It also shows that EBPC performs the best, followed by EBP. Figure 4.18(b) shows that the average subject recognition probability increases when number of cameras increases and it shows that EBPC produces the best quality. The algorithm time complexities are compared in Figures

4.18(c) and 4.18(d). These results indicate that EBPC and GBGC have the least overall execution time. The new implemented reachability enhancement makes GBGC and GBG take less time than EBPC and EBP, respectively. However, the planning algorithm still delivers better overall recognition probability and percentage of covered subjects.

Effect of Subject Arrival Rate

Figure 4.19(a) shows the effect of arrival rate on the percentage of covered subjects. As the arrival rate increases, GBG and EBP degrade much faster than GBGC and EBPC, respectively. The clustering makes the system more scalable with respect to the arrival rate because the clustering process allows the search of only the areas that are populated with subjects. Figure 4.19(b) shows the same trend for the average subject recognition probability. EBPC behaves better at higher arrival rates. Figures 4.19(c) and 4.19(d) show how the time complexity increases when the arrival rate increases. EBPC and GBGC still have less overall execution time than EBP and GBG, respectively.

Recording Time Interval Lengths

Figures 4.20(a) and 4.20(b) demonstrate the effect of T_R on the percentage of covered subjects and the subject recognition probability. They show a small degradation in both metrics as the recording period increases. The main reason for the degradation is that when more time is spent on recording, more subjects leave the scene without being captured. This behavior is more obvious with GBG and GBGC since they capture one frame in each recording interval.

Effect of Pre-Recording Time Interval Lengths

Figures 4.21(a) and 4.21(b) demonstrate the effect of T_p on the percentage of covered subjects and subject recognition probability. They show an increase in performance and then start to degrade. A peak is formed in each curve at the best pre-recording interval. The shorter this period is the better because we

can spare more time for recording. We observe that EBPC has the best performance with the least period. As this period increases, the performance degrades for all algorithms due to (1) more subjects leaving the area with longer pre-recording periods, (2) subject state predictions over long intervals becoming obsolete, and (3) the less time being spent on recording when longer time is spent on calculations, in a bounded overall simulation time.

Impact of Scene Area

Figures 4.22(a) and 4.22(b) show how increasing the area affects the percentage of covered subjects and subject recognition probability. It is expected that these metrics will decrease when the area increases. The FoVs of cameras will not be able to cover the area as it grows. Additionally, the algorithm execution times will increase because the lifetimes of subjects in the scene will become longer, as shown in Figures 4.22(c) and 4.22(d). With larger areas, the scene become more sparse, and the camera moving-time overhead between frames in the plan in EBP and EBPC becomes longer. GBG and GBGC tend to perform relatively better as the area increases.

Impact of Cluster Size

Figures 4.23(a) and 4.23(b) show how increasing the cluster size affects the system. As we increase the cluster size, more subjects are being covered and better performance is achieved. However, the increase in performance with the cluster size saturates after a certain value. With larger cluster, subjects are captured at lower resolutions. Hence, the algorithms start preferring smaller cluster sizes with better resolutions.

4.4 Conclusions

We have addressed the camera control problem in automated video surveillance by developing a solution that seeks to maximize the overall subject recognition probability. This control of cameras is

based on the direction of the subject's movement and its location, distance from the cameras, occlusion, overall recognition probability so far, and the expected time to leave the site, as well as the movements of cameras and their capabilities and limitations. The developed solution works with realistic 3D environments and not just 2D scenes.

We have analyzed the effectiveness of the proposed solution through extensive simulation. The main results can be summarized as follows.

- The proposed EBP scheduling scheme achieves the best recognition probability.
- The number of PTZ cameras used in the network highly affects the scheduling schemes, with different schemes benefiting from increasing this number at different degrees.
- The subject recognition probability improves with the number of PTZ cameras, but the plan building time becomes longer. Therefore a compromise must be made.
- EBP achieves the best scalability when increasing the subject arrival rate or the area of the site being monitored.
- Applying clustering enhances the scalability of the surveillance system.

```
00. //Input: planSeeds_arr, a Seeds frame list to each camera
00. //Output: finalPlan, detailed plan for each camera
00. BuildPlans(){
01.  for (c = 0; i < C; c++){ // for each camera
02.   finalPlan[c] = {}; maxRecProb = 0;
03.   // sort frames in each list according to the zoom
04.   sort(planSeeds_arr[c][ ], zoom);
05.   for (j = 0; j < seeds; j++){ //for each frame
06.    plan = {}; planRecProb = 0; // initialize plan variables
07.    k = j; // assign start node for a camera plan
08.    if (T_move,c→k > T_0) then
09.     continue; // Time to move to frame k exceeds T_0
10.    else {
11.     plan.push(planSeeds_arr[c][k]);
12.     planRecProb+ = ψ(k);
13.     forwardFlag = 1;
14.     T_rec = T_R;
15.     T_rec = T_rec − T_i;
16.     while (T_rec > 0){
17.      if (k == 0) then  forwardFlag = 1;
18.      if (k == seeds) then  forwardFlag = 0;
19.      if ( forwardFlag == 1) then k++;
20.      else  k − −;
21.      if ((T_move,c→k + T_i) < T_rec) then{
22.       plan.push(planSeeds_arr[c][k]);
23.       planRecProb+ = ψ(k);
24.       T_rec = T_rec − T_i;
25.      } End of if
26.     } End of while (T_rec > 0)
27.    } End of else T_move is valid
28.    if (planRecProb > maxRecProb){
29.     maxRecProb = planRecProb;
30.     finalPlan[c] = plan;
31.    } End of if
32.   } End of frame loop j
33.  } End of camera loop c
34. } // End of BuildPlans()
```

Figure 4.7: A Simplified Algorithm for Building Cameras' Plans, [T_R: recording interval, T_0: remaining time in the pre-recording interval, T_i: recording time per frame, T_{rec}: remaining time in the recording interval, $T_{move,c\rightarrow k}$: time to move camera c to frame k.]

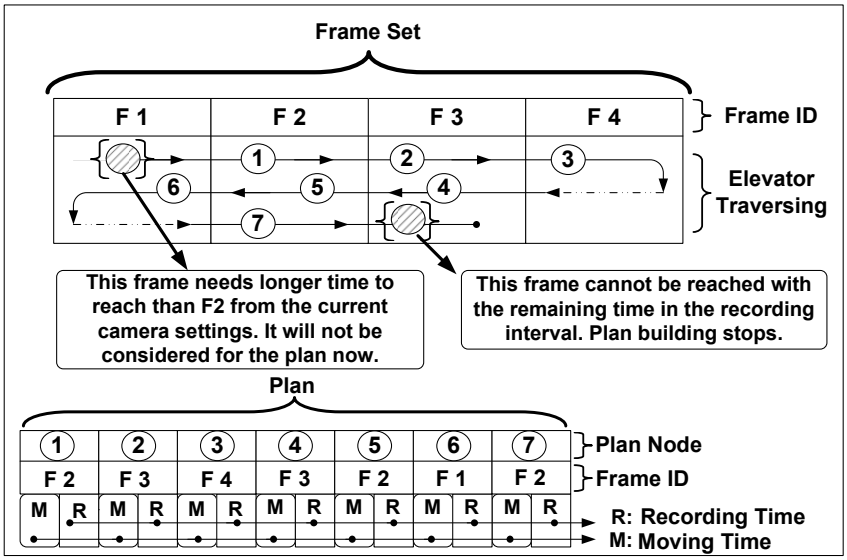


Figure 4.8: An Example of Plan Generation

```
01. //Input: subjectList[], List of the active subjects
02. //Output: clusterList[],Cluster center list of the grouped subjects
03. Find-N-Clusters(){
04.  for (i = 0; i < subjectList.size(); i++){
05.   clusterList[i].Center = subjectList[i].Loc;
06.   clusterList[i].Dir = subjectList[i].Dir; // Direction angle
07.  }
08.  for (i = 0; i < clusterList.size(); i++){
09.   mergIndex = −1;
10.   for (j = 0; j < clusterList.size(); j++){
11.    Distance = DIST(clusterList[i].Center, clusterList[j].Center);
12.    // Calculate the perpendicular distance
13.    DistancePer = DISTPER(clusterList[i].Center, clusterList[j].Center);
14.    Direction = DIR(clusterList[i].Dir, clusterList[j].Dir);
15.    if (Direction > ANGLE_TH||Distance > DIST_TH||DistancePer > WIDTH_TH)
16.     continue;
17.    totalDist = (DIST_TH−Distance)/DIST_TH+(WIDTH_TH−DistancePer)/WIDTH_TH+
18.    (ANGLE_TH − Direction)/ANGLE_TH;
19.    if (j == 0)
20.     minDist = totalDist;
21.    if (totalDist > minDist)
22.     continue;
23.    minDist = totalDist;
24.    mergIndex = j;
25.   } End of loop j
26.   if (mergIndex == −1)
27.    continue;
28.   clusterList[i] = Merge(clusterList[i], clusterList[mergIndex])
29.   clusterList[i].Center = Average(clusterList[i].Center, clusterList[mergIndex].Center)
30.   clusterList[i].Dir = Average(clusterList[i].Dir, clusterList[mergIndex].Dir)
31.   Remove(clusterList[j])
32.  } End of loop i
33. } // End of Find-N-Clusters()
```

Figure 4.9: Simplified Algorithm to Find Clusters Centers and Subjects

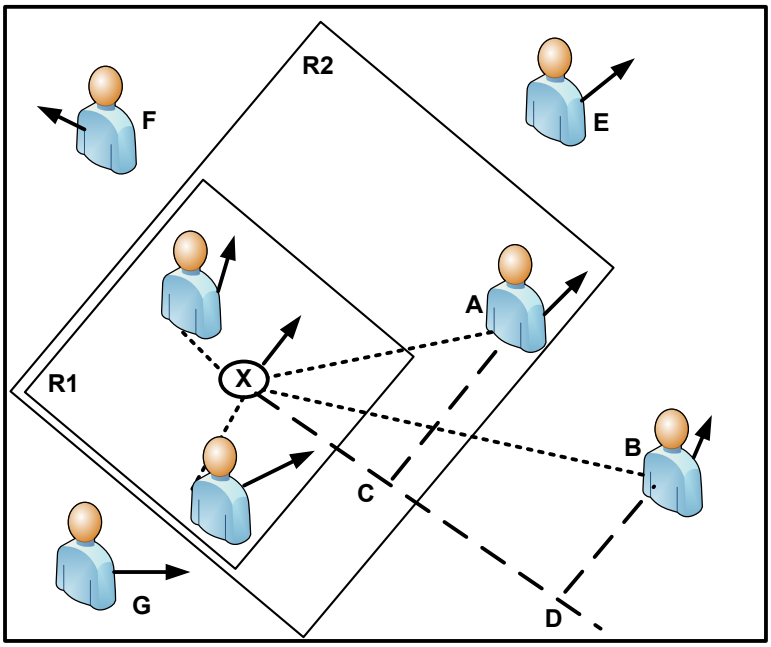
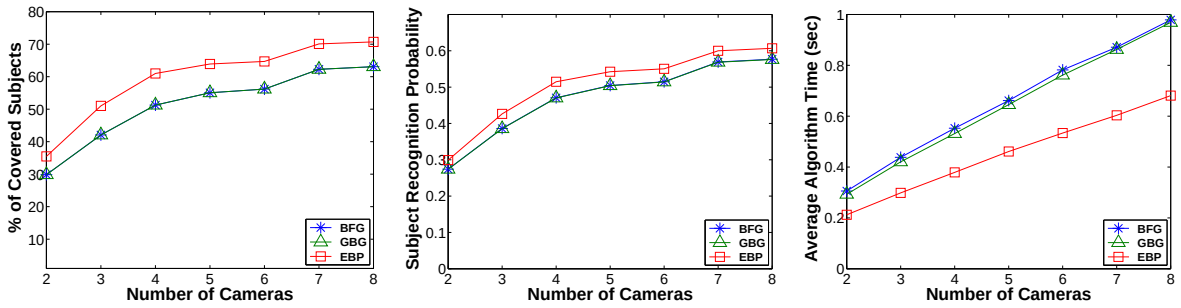


Figure 4.10: Clusters Illustration

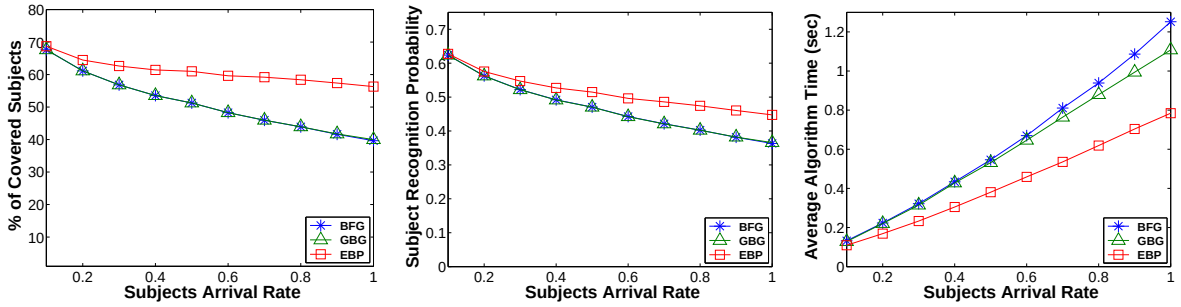
```
01. //Input: subjectList[], clusterList[], cameraList[]
02. //Output: frameList[], a frame list
03. populateFrameList(){
04.   for (c = 0; c < cameraList.size(); c++){ // for each camera
05.     FindOcclusion(subjectList); // Location
06.     for (i = 0; i < clusterList.size(); i++){ // for each cluster
07.       tempFrame.PTZ = FindPTZ(cameraList[c],clusterList[i]); // find the Pan-Tilt-Zoom settings
08.       tempFrame.RecProb = 0;
09.       for (j = 0; j < subjectList.size(); j++){ // for each subject
10.         if (subjectList[j].Invalid()) continue;
11.         //add recognition prob. of the subject to the frame
12.         tempFrame.RecProb += subjectRecProb(subjectList[j]);
13.       } End of loop j
14.         frameList.Add(tempFrame)// add current frame to the frame list
15.     } End of loop i
16.   } End of loop c
17. } // End of populateFrameList()
```

Figure 4.11: Simplified Algorithm to Populate Frames



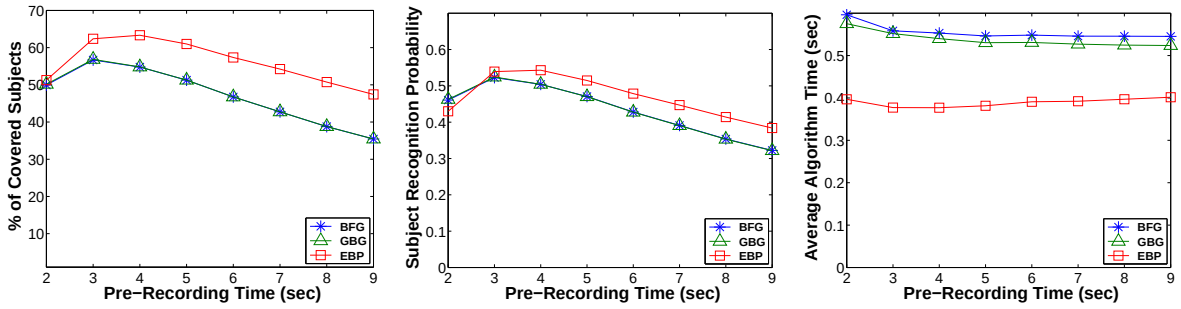
(a) Covered Subjects Percent (b) Average Subject Recognition Probability (c) Average Time

Figure 4.12: Comparing Effectiveness of Various Scheduling Approaches



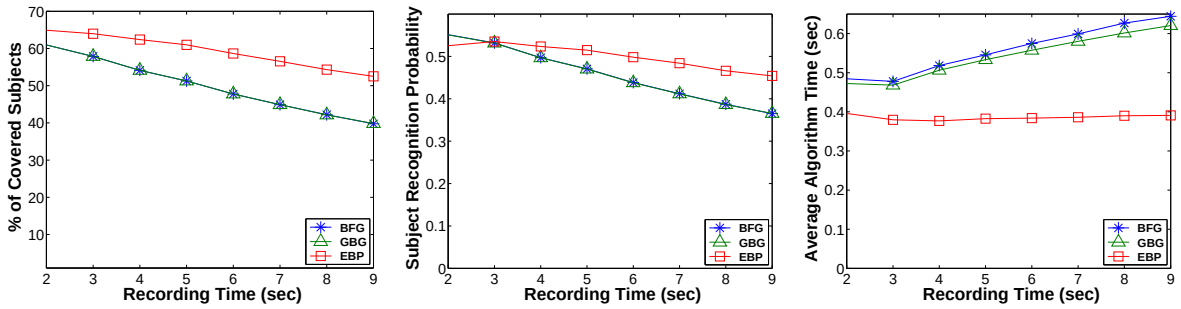
(a) Covered Subjects Percent (b) Average Subject Recognition Probability (c) Average Time

Figure 4.13: Impact of Arrival Rate



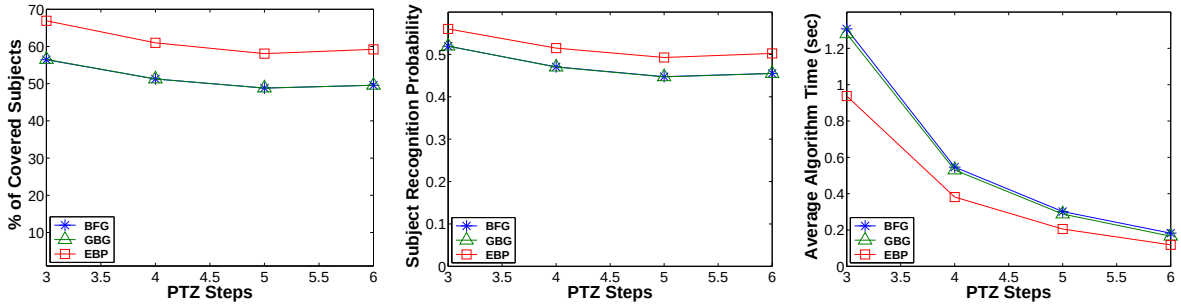
(a) Covered Subjects Percent (b) Average Subject Recognition Probability (c) Average Time, T_R

Figure 4.14: Impact of Pre-recording Period Length



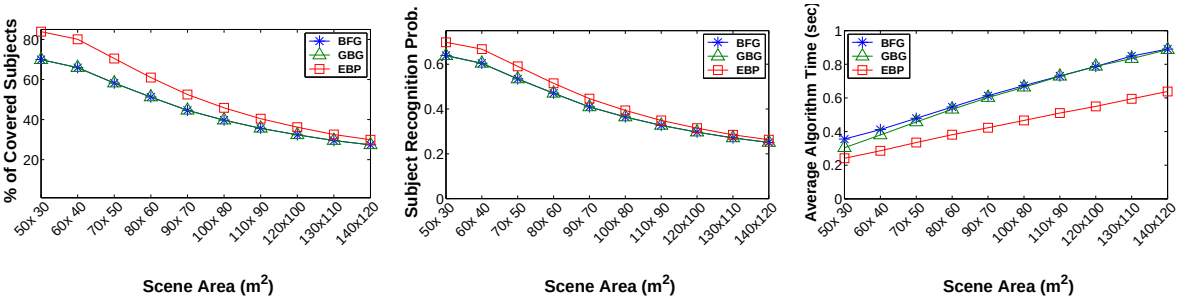
(a) Covered Subjects Percent (b) Average Subject Recognition Probability (c) Average Time

Figure 4.15: Impact of Recording Period Length



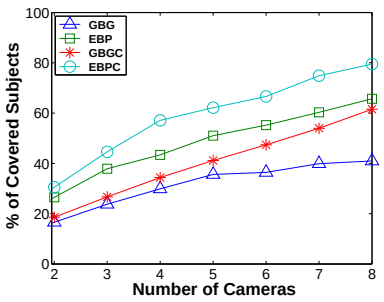
(a) Covered Subjects Percent (b) Average Subject Recognition Probability (c) Average Time

Figure 4.16: Impact of PTZ Camera Step Size

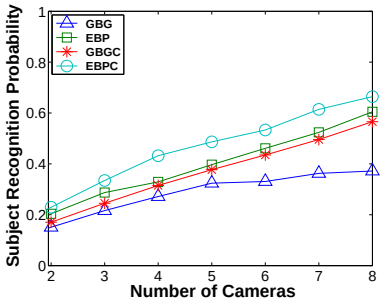


(a) Covered Subjects Percent (b) Average Subject Recognition Probability (c) Average Time

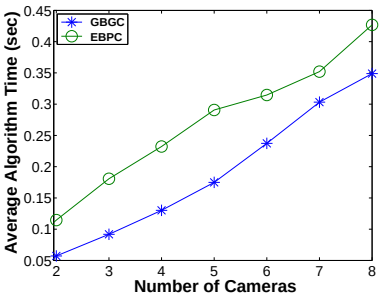
Figure 4.17: Impact of Scene Area



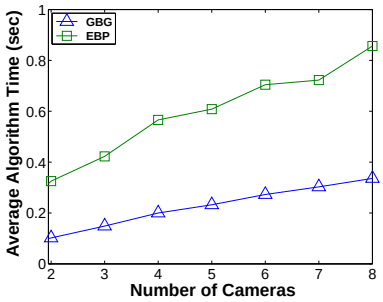
(a) Covered Subjects Percent



(b) Average Subject Recognition Probability

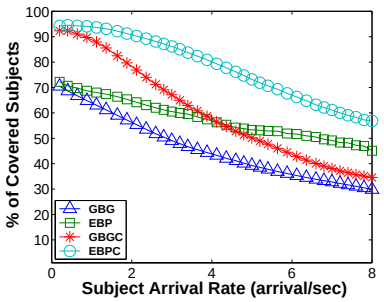


(c) Average Time

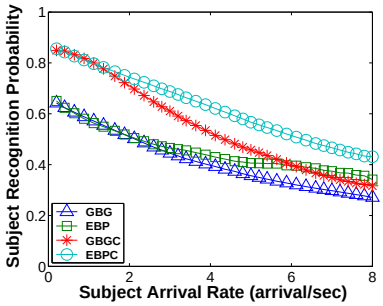


(d) Average Time

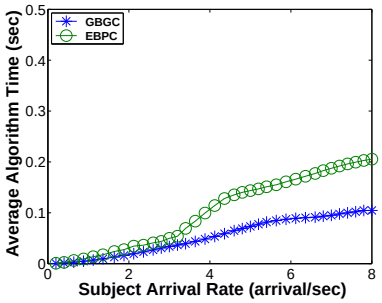
Figure 4.18: Comparing Effectiveness of Clustering with Number of Cameras



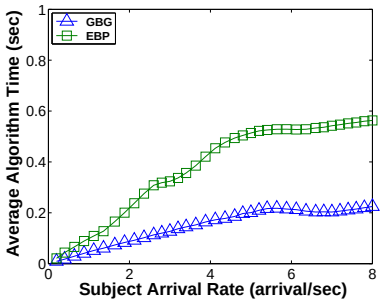
(a) Covered Subjects Percent



(b) Average Subject Recognition Probability

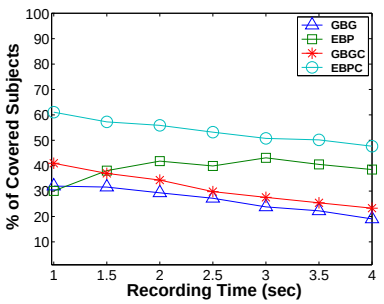


(c) Average Time

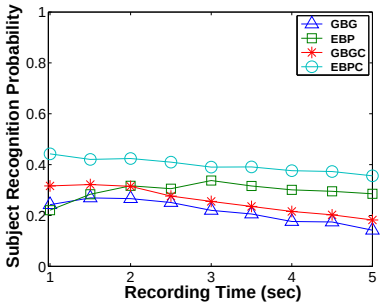


(d) Average Time

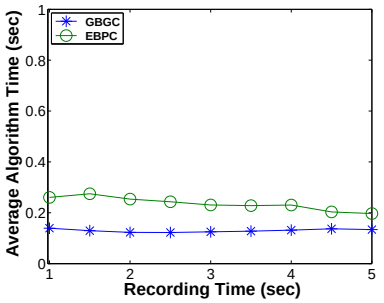
Figure 4.19: Comparing Effectiveness of Clustering with Arrival Rate



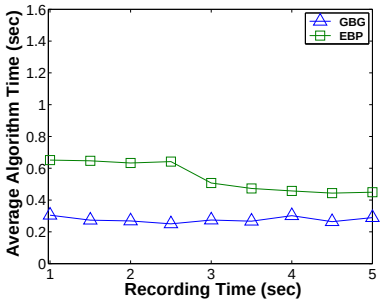
(a) Covered Subjects Percent



(b) Average Subject Recognition Probability

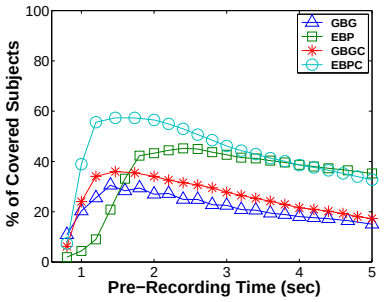


(c) Average Time

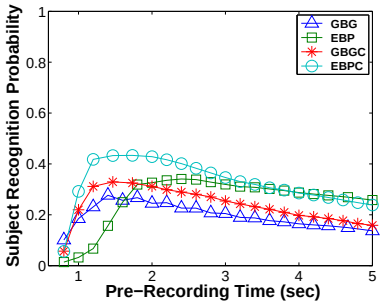


(d) Average Time

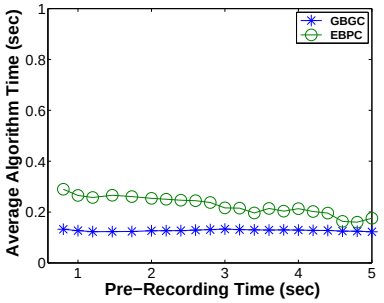
Figure 4.20: Comparing Effectiveness of Clustering with Recording Period



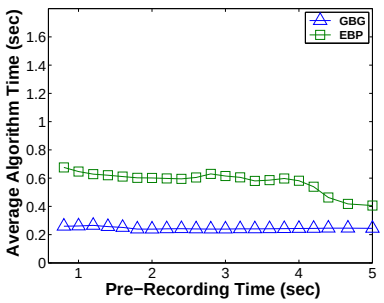
(a) Covered Subjects Percent



(b) Average Subject Recognition Probability

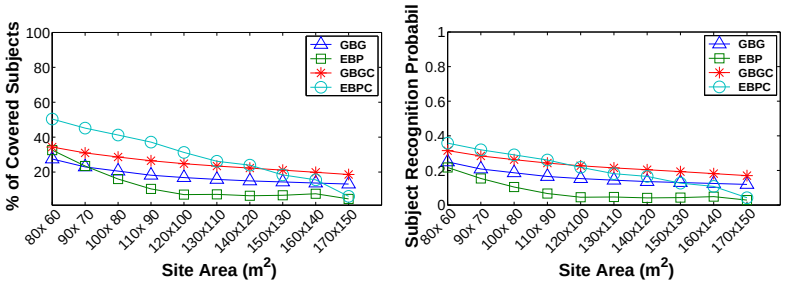


(c) Average Time

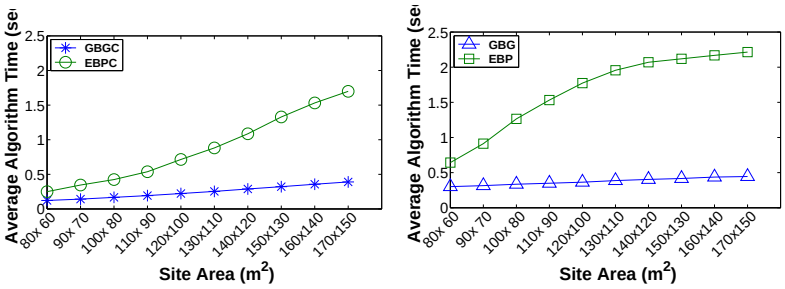


(d) Average Time

Figure 4.21: Comparing Effectiveness of Clustering with Pre-recording Period

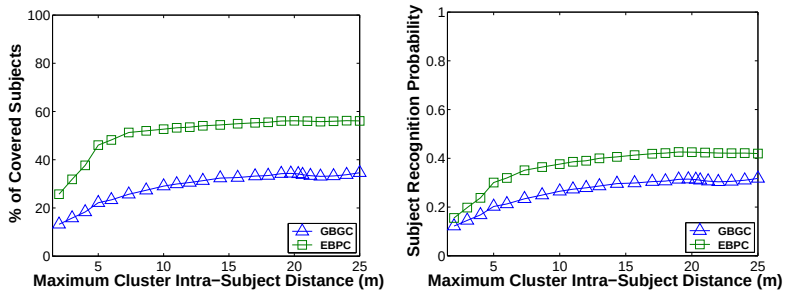


(a) Covered Subjects Percent (b) Average Subject Recognition Probability



(c) Average Time (d) Average Time

Figure 4.22: Comparing Effectiveness of Clustering with Scene Area



(a) Covered Subjects Percent (b) Average Subject Recognition Probability

Figure 4.23: Comparing Effectiveness of Clustering with Cluster Size

CHAPTER 5

SUMMARY AND FUTURE WORK

In this chapter, we summarize the work that has been presented in the dissertation and list the related publications.

5.1 Summary

- We have developed a delivery framework for streaming media with advertisements and an associated pricing model. The delivery model combines the benefits of periodic broadcasting and stream merging. The advertisements' revenues are used to subsidize the price of the media streaming, and the pricing is determined based on the total ads' viewing time. For the proposed framework, we have presented an efficient heuristic ad allocation scheme and four constraint-based scheduling policies and have analyzed two ad delivery options. In addition, we have studied the support of targeted advertisements. We have studied the effectiveness of the proposed framework and strategies through extensive simulation, considering several performance metrics and three models of the arrival rate.
- We have proposed a highly accurate waiting-time prediction algorithm that estimates the expected number of viewed ads by utilizing detailed information about the system state and the applied scheduling policy. We also have proposed a pricing scheme based on the expected waiting times, which include ads' viewing times. The revenue generated by ads is used to subsidize the price and are allocated to clients proportionally to their expected waiting times.
- We have analyzed 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.
- We have developed a solution for scalable automated watch-list-based surveillance. The main objec-

tive of this work is to maximize subjects' recognition probability. The system drives PTZ cameras by utilizing information captured by wider angle view cameras. we have used empirical data to investigate cameras with different FoVs, capabilities and limitations. We also implemented a prediction algorithm to investigate frames quality from the 3D scene before capturing. We investigated three different scheduling algorithms: Brute Force Grouping (BFG), Grid Based Grouping (GBG), and Elevator Based planning (EBP). Practical factors are considered while developing and applying algorithms.

- We have integrated a clustering solution for the subject grouping problem in the scalable automated watch-list-based surveillance system. The system with the clustering technique showed better results especially in terms of scalability.

5.2 Future Work

Many directions can be pursued from this research. In the ad streaming topic, the ad targeting is getting more popular. Delivering ads that match viewers' interests is being of extreme necessity for business owners. Finding the best ad-match for a viewer is one of the important topics that can be investigated and integrated in our system.

In the automated video surveillance system, we found that managing the time is very important in increasing the system performance. However, we only focused on enhancing the frames population and selection algorithms. Another significant time overhead exists in the camera focus time. Developing an algorithm that enhances the camera focus time will further improve the system performance.

5.3 List of Publications

- **Musab Al-Hadrusi** and Nabil J. Sarhan. A Scalable Delivery Framework and a Pricing Model for Commercial VOD Systems with Video Ads. Multimedia Tools and Applications, 2013. (under second round of revision and review)

- **Musab Al-Hadrusi** and Nabil J. Sarhan. Efficient Control of PTZ Cameras in Automated Video Surveillance Systems. In Proceedings of the IEEE International Symposium on Multimedia, (ISM 2012), December 2012.
- **Musab Al-Hadrusi** and Nabil J. Sarhan. Client-Driven Price Selection for Scalable Video Streaming with Advertisements. In Proceedings of the International MultiMedia Modeling Conference (MMM 2012), Klagenfurt, Austria, January 2012.
- Nabil J. Sarhan and **Musab Al-Hadrusi**. Waiting-Time Prediction and QoS-Based Pricing for Video Streaming with Advertisements. In Proceedings of the IEEE International Symposium on Multimedia, (ISM 2010), December 2010.
- Nabil J. Sarhan, Mohammad A. Alsmirat, and **Musab Al-Hadrusi**. Waiting-Time Prediction in Scalable On-Demand Video Streaming. ACM Transactions on Multimedia Computing, Communications, and Applications (ACM TOMCCAP), Volume 6, Issue 2, March 2010.
- **Musab Al-Hadrusi** and Nabil J. Sarhan. A Scalable Delivery Framework and a Pricing Model for Streaming Media with Advertisements. In Proceedings of SPIE Multimedia Computing and Networking (MMCN), January/February 2008.
- Mohammad Alsmirat, **Musab Al-Hadrusi**, and Nabil J. Sarhan. Analysis of Waiting-Time Predictability in Scalable Media Streaming. In Proceedings of ACM Multimedia 2007, pages 727 - 736, September 2007.
- **Musab Al-Hadrusi** and Nabil J. Sarhan. Scalable Delivery and Pricing of Streaming Media with Advertisements. Short Paper. In Proceedings of ACM Multimedia 2007, pages 791 - 794, September 2007.

BIBLIOGRAPHY

[1] Alexa, "`http://www.alexa.com/topsites`" .

[2] YouTube, "`http://www.youtube.com/t/press_statistics`" .

[3] The Associated Press, "http://www.usatoday.com/sports/football/nfl/story/2011-12-20/super-bowl-online-streaming/52101970/1," .

[4] Kirstie Ball, David Lyon, David M. Wood, Clive Norris, Charles Raab, "A report on the surveillance society: For the information commissioner by the surveillance studies network. Online Report at http://www.ico.gov.uk/upload/documents/library/data_protection/practical_application/surveillance_society_full_report_2006.pdf," 2006.

[5] Satyajit Banerjee, Atish Datta Chowdhury, Subhas Kumar Ghosh, and Subhas Kumar Ghosh, "Video surveillance with PTZ cameras: The problem of maximizing effective monitoring time.," in *Proc. of International Conference on Distributed Computing and Networking (ICDCN)*, 2010, pp. 341–352.

[6] Ser-Nam Lim, Larry S. Davis, Anurag Mittal, and Anurag Mittal, "Task scheduling in large camera networks.," in *Proc. of Asian Conference on Computer Vision (ACCV)*, 2007, pp. 397–407.

[7] Chung-Hao Chen, Yi Yao, David Page, Besma R. Abidi, Andreas Koschan, Mongi A. Abidi, and Mongi A. Abidi, "Camera handoff and placement for automated tracking systems with multiple omnidirectional cameras," *Computer Vision and Image Understanding*, pp. 179–197, 2010.

[8] Yiming Li and B. Bhanu, "A comparison of techniques for camera selection and handoff in a video network," in *Proc. of ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC).*, 30 2009-sept. 2 2009, pp. 1 –8.

[9] Benjamin Deutsch, Stefan Wenhardt, Heinrich Niemann, and Heinrich Niemann, "Multi-step multi-camera view planning for real-time visual object tracking.," in *Proc. of German Association for Pattern Recognition (DAGM) Symposium*, 2006, pp. 536–545.

[10] C. Piciarelli, C. Micheloni, and G.L. Foresti, "PTZ camera network reconfiguration," in *Proc. of ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC).*, 30 2009-sept. 2 2009, pp. 1 –7.

[11] George Pallis and Athena Vakali, "Insight and perspectives for content delivery networks," *Communications of the ACM*, vol. 49, pp. 101–106, January 2006.

[12] B. Li C. Wu and S. Zhao, "Diagnosing network-wide p2p live streaming inefficiencies," in *Proc. of IEEE INFOCOM*, April 2009.

[13] V. Aggarwal, R. Caldebank, V. Gopalakrishnan, R. Jana, K. Ramakrishnan, and F. Yu, "The effectiveness of intelligent scheduling for multicast video-on-demand," in *Proc. of ACM Multimedia*, 2009, pp. 421–430.

[14] S. Ratnasamy and A. Ermolinskiy, "Revisiting IP-multicast," in *Proc. of ACM SIGCOMM*, September 2006.

[15] K. Almeroth, "Multicast help wanted: From where and how much?," *Keynote Speech, Workshop on Peer-to-Peer Multicasting, Consumer Communications and Networking Conference*, January 2007.

[16] Kien A. Hua, Ying Cai, and Simon Sheu, "Patching: A multicast technique for true Video-on-Demand services," in *Proc. of ACM Multimedia*, 1998, pp. 191–200.

[17] Ying Cai and Kien A. Hua, "An efficient bandwidth-sharing technique for true video on demand systems," in *Proc. of ACM Multimedia*, Oct. 1999, pp. 211–214.

[18] D. L. Eager, M. K. Vernon, and J. Zahorjan, "Optimal and efficient merging schedules for Video-on-Demand servers," in *Proc. of ACM Multimedia*, Oct. 1999, pp. 199–202.

[19] Marcus Rocha, Marcelo Maia, Italo Cunha, Jussara Almeida, and Sergio Campos, "Scalable media streaming to interactive users," in *Proc. of ACM Multimedia*, Nov. 2005, pp. 966–975.

[20] Huadong Ma, G. Kang Shin, and Weibiao Wu, "Best-effort patching for multicast true VoD service," *Multimedia Tools and Applications*, vol. 26, no. 1, pp. 101–122, 2005.

[21] Nabil J. Sarhan, Mohammad A. Alsmirat, and Musab Al-Hadrusi, "Waiting-time prediction in scalable on-demand video streaming," *ACM Transactions on Multimedia Computing, Communications, and Applications (ACM TOMCCAP)*, vol. 6, no. 2, pp. 1–24, March 2010.

[22] Dan Rayburn, *Streaming and Digital Media: Understanding the Business and Technology*, Focal Press, 2007.

[23] E. Harwood, *DIGITAL CCTV: A Security Professionals Guide*, Elsevier, 2008.

[24] H. Kruegle, *CCTV Surveillance: Analog and Digital Video Practices and Technology*, Elsevier, second edition, 2007.

[25] F. Nilsson, *Intelligent Network Video: Understanding Modern Video Surveillance Systems*, CRC Press, 2009.

[26] Rita Cucchiara, "Multimedia surveillance systems," in *Proc. of ACM International Workshop on Video Surveillance and Sensor Networks (VSSN)*, New York, NY, USA, 2005, pp. 3–10, ACM.

[27] Cash J. Costello, Christopher P. Diehl, Amit Banerjee, and Hesky Fisher, "Scheduling an active camera to observe people," in *Proc. of ACM International Workshop on Video Surveillance and Sensor Networks (VSSN)*, New York, NY, USA, 2004, pp. 39–45, ACM.

[28] S. Lim N. Krahnstoever, T. Yu and K. Patwardhan, "Collaborative control of active cameras in large-scale surveillance," in *Networks Principles and Applications in Multi-Camera*, H. Aghajan and A. Cavallaro, Eds., pp. 165–188. Elsevier, 2009.

[29] B. Schiele P. Dollar, C. Wojek and P. Perona, "Pedestrian detection: A benchmark," in *Proc. of Computer Vision and Pattern Recognition Conf. (CVPR)*, June 2009.

[30] Faisal Qureshi and Demetri Terzopoulos, "Surveillance camera scheduling: a virtual vision approach," *Multimedia Systems*, vol. 12, no. 3, pp. 269–283, 2006.

[31] Yiliang Xu and Dezhen Song, "Systems and algorithms for autonomous and scalable crowd surveillance using robotic PTZ cameras assisted by a wide-angle camera," *Auton. Robots*, vol. 29, no. 1, pp. 53–66, 2010.

[32] Hazem El-Alfy, David Jacobs, Larry Davis, and Larry Davis, "Assigning cameras to subjects in video surveillance systems.," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2009, pp. 837–843.

[33] K. A. Hua and S. Sheu, "Skyscraper broadcasting: A new broadcasting scheme for metropolitan Video-on-Demand system," in *Proc. of ACM SIGCOMM*, Sept. 1997, pp. 89–100.

[34] L. Juhn and L. Tseng, "Harmonic broadcasting for Video-on-Demand service," *IEEE Trans. on Broadcasting*, vol. 43, no. 3, pp. 268–271, Sept. 1997.

[35] J.-F. Pâris, S. W. Carter, and D. D. E. Long, "Efficient broadcasting protocols for video on demand," in *Proc. of the Int'l Symp. on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, July 1998, pp. 127–132.

[36] Cheng Huang, Ramaprabhu Janakiraman, and Lihao Xu, "Loss-resilient on-demand media streaming using priority encoding," in *Proc. of ACM Multimedia*, Oct. 2004, pp. 152–159.

[37] P. Gill, L. Shi, A. Mahanti, Z. Li, and D. Eager, "Scalable on-demand media streaming for heterogeneous clients," *ACM Transactions on Multimedia Computing, Communications, and Applications (ACM TOMCCAP)*, vol. 5, no. 1, pp. 1–24, Oct. 2008.

[38] S. W. Carter and D. D. E. Long, "Improving Video-on-Demand server efficiency through stream tapping," in *Proc. of International Conference on Computer Communication and Networks (ICCCN)*, Sept. 1997, pp. 200–207.

[39] D. L. Eager, M. K. Vernon, and J. Zahorjan, "Minimizing bandwidth requirements for on-demand data delivery," *IEEE Trans. on Knowledge and Data Engineering*, vol. 13, no. 5, pp. 742–757, Sept. 2001.

[40] L. Gao, J. Kurose, and D. Towsley, "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.

[41] A. Dan, D. Sitaram, and P. Shahabuddin, "Scheduling policies for an on-demand video server with batching," in *Proc. of ACM Multimedia*, Oct. 1994, pp. 391–398.

[42] C. C. Aggarwal, J. L. Wolf, and P. S. Yu, "The maximum factor queue length batching scheme for Video-on-Demand systems," *IEEE Trans. on Computers*, vol. 50, no. 2, pp. 97–110, Feb. 2001.

[43] Nabil J. Sarhan and Bashar Qudah, "Efficient cost-based scheduling for scalable media streaming," in *Proc. of Multimedia Computing and Networking Conf. (MMCN)*, January 2007, pp. 327–334.

[44] P. Basu, A. Narayanan, W. Ke, T. D. C. Little, and A. Bestavros, "Optimal scheduling of secondary content for aggregation in video-on-demand systems," in *Proc. of International Conference on Computer Communications and Networks*, Oct. 1999, pp. 104–109.

[45] Prithwish Basu and Thomas D. C. Little, "Pricing considerations in video-on-demand systems," in *Proc. of ACM Multimedia*, Nov. 2000, pp. 359–361.

[46] P. Jonathon Phillips, Harry Wechsler, Jeffrey Huang, Patrick J. Rauss, and Patrick J. Rauss, "The feret database and evaluation procedure for face-recognition algorithms.," *Image and Vision Computing*, pp. 295–306, 1998.

[47] P. Jonathon Phillips, Patrick Grother, Ross J. Micheals, Duane M. Blackburn, Elham Tabassi, Mike Bone, and Mike Bone, "Face recognition vendor test 2002.," in *Proc. of Workshop on Analysis and Modeling of Faces and Gestures AMFG*, 2003, p. 44.

[48] P. Jonathon Phillips, W. Todd Scruggs, Alice J. O'Toole, Patrick J. Flynn, Kevin W. Bowyer, Cathy L. Schott, Matthew Sharpe, and Matthew Sharpe, "Frvt 2006 and ice 2006 large-scale experimental results.," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 831–846, 2010.

[49] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 643–660, 2001.

[50] Ralph Gross, Simon Baker, Iain Matthews, and Takeo Kanade, "Face recognition across pose and illumination," *Handbook of Face Recognition*, 2004.

[51] Mislav Grgic, Kresimir Delac, Sonja Grgic, and Sonja Grgic, "Scface - surveillance cameras face database.," *Multimedia Tools and Applications*, pp. 863–879, 2011.

[52] Yi Yao, Besma R. Abidi, Nathan D. Kalka, Natalia A. Schmid, and Mongi A. Abidi, "Improving long range and high magnification face recognition: Database acquisition, evaluation, and enhancement," *Computer Vision and Image Understanding*, vol. 111, pp. 111–125, August 2008.

[53] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," *ACM Computing Surveys*, vol. 35, pp. 399–458, December 2003.

[54] http://www.bayometric.com/products/Face Recognition-SDK.htm, "Faceit sdk," .

[55] http://www.neurotechnology.com/face biometrics.html, "Verilook sdk," .

[56] Xuhui Zhou, Robert T. Collins, Takeo Kanade, and Peter Metes, "A master-slave system to acquire biometric imagery of humans at distance," in *Proc. of ACM SIGMM International Workshop on Video surveillance (IWVS)*, New York, NY, USA, 2003, pp. 113–120, ACM.

[57] Arun Hampapur, Sharath Pankanti, Andrew W. Senior, Ying li Tian, Lisa M. G. Brown, Ruud M. Bolle, and Ruud M. Bolle, "Face cataloger: Multi-scale imaging for relating identity to location.," in *Proc. of IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*, 2003, pp. 13–20.

[58] Srinivasan Jagannathan and Kevin C. Almeroth, "The dynamics of price, revenue, and system utilization," in *Proc. of the IFIP/IEEE International Conference on Management of Multimedia Networks and Services*, 2001, pp. 329–344.

[59] Rick E. Bruner and Jai Singh, "Video ad benchmarks: Average campaign performance metrics. online white paper at `http://www.doubleclick.com/insight/pdfs/dc_videobenchmarks_0702.pdf`," February 2007.

[60] J. Ponce D. A. Forsyth, *Computer Vision: A Modern Approach*, Prentice Hall, first edition, 2002.

[61] M. Shah, "Fundamentals of computer vision," .

[62] B.J. Boom, G.M. Beumer, L.J. Spreeuwers, and R.N.J. Veldhuis, "The effect of image resolution on the performance of a face recognition system," in *Proc. of International Conference on Control, Automation, Robotics and Vision(ICARCV),*, dec. 2006, pp. 1 –6.

[63] Jingdong Wang, Changshui Zhang, and Heung yeung Shum, "Face image resolution versus face recognition performance based on two global methods," in *Proc. of Asia Conference on Computer Vision (ACCV)*, 2004.

[64] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 881–892, July 2002.

[65] Rui Xu and Donald C. Wunsch II, "Survey of clustering algorithms," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645–678, 2005.

[66] Joe H. Ward, Jr., "Hierarchical grouping to optimize an objective function," vol. 58, no. 301, pp. 236–244, Mar. 1963.

ABSTRACT

DESIGN AND ANALYSIS OF SCALABLE VIDEO STREAMING SYSTEMS

by

MUSAB SALEM AL-HADRUSI

May 2013

Adviser: Dr. Nabil Sarhan

Major: Computer Engineering

Degree: Doctor of Philosophy

Despite the advancement in multimedia streaming technology, many multimedia applications are still face major challenges, including provision of Quality-of-Service (QoS), system scalability, limited resources, and cost. In this dissertation, we develop and analyze a new set of metrics based on two particular video streaming systems, namely: (1) Video-on-Demand (VOD) with video advertisements system and (2) Automated Video Surveillance System (AVS).

We address the main issues in the design of commercial VOD systems: scalability and support of video advertisements. We develop a scalable delivery framework for streaming media content with video advertisements. The delivery framework combines the benefits of stream merging and periodic broadcasting. In addition, we propose new scheduling policies that are well-suited for the proposed delivery framework. We also propose a new prediction scheme of the ad viewing times, called *Assign Closest Ad Completion Time* (ACA). Moreover, we propose an enhanced business model, in which the revenue generated from advertisements is used to subsidize the price. Additionally, we investigate the support of targeted advertisements, whereby clients receive ads that are well-suited for their interests and needs. Furthermore, we provide the clients with the ability to select from multiple price options, each with an associate expected number of viewed ads. We provide detailed analysis of the proposed VOD system, considering realistic workload and a wide range of design parameters.

In the second system, Automated Video Surveillance (AVS), we consider the system design for optimizing the subjects recognition probabilities. We focus on the management and the control of various Pan, Tilt, Zoom (PTZ) video cameras. In particular, we develop a camera management solution that provides the best tradeoff between the subject recognition probability and time complexity. We consider both subject grouping and clustering mechanisms. In subject-grouping, we propose the *Grid Based Grouping* (GBG) and the *Elevator Based Planning* (EBP) algorithms. In the clustering approach, we propose the *(GBG) with Clustering* (GBGC) and the *EBP with Clustering* (EBPC) algorithms. We characterize the impact of various factors on recognition probability. These factors include resolution, pose and zoom-distance noise. We provide detailed analysis of the camera management solution, considering realistic workload and system design parameters.

AUTOBIOGRAPHICAL STATEMENT

Musab Al-Hadrusi is a Ph.D candidate in the Electrical and Computer Engineering Department at Wayne State University. He received his M.Sc. degree in Computer Engineering and his B.S. degree in Electrical and Computer Engineering from Jordan University of Science and Technology. Musab Al-Hadrusi main research is in multimedia streaming and scheduling, automated video surveillance, systems simulation and modeling. Musab Al-Hadrusi has served as reviewer and TPC member in different multimedia conferences and journals such as IEEE transaction on multimedia and MMedia (2012,2013). He is also an IEEE, Tau-Beta-Pi and Golden Key member.