VIDEO PERSONALIZATION FOR RESOURCE-CONSTRAINED ENVIRONMENTS

by

YONG WEI

(Under the Direction of Suchendra M. Bhandarkar)

ABSTRACT

This dissertation has focused on the design and implementation of a client-centered multimedia content adaptation system suitable for mobile environments comprising of resource-constrained handheld devices or clients. The current proliferation of mobile computing devices and network technologies has created enormous opportunities for mobile device users to communicate with multimedia servers, using multimedia streams. One of the natural limitations of these handheld devices is that they are constrained by their battery power capacity, rendering and display capability, viewing time limit and in many situations, by the available network bandwidth connecting these devices to video data servers. Thus, it is a necessity to develop a mobile client-centered multimedia personalization system to fulfill clients' request while satisfying client-side resource constraints.

The primary contributions of this work are (1) the overall architecture of the client-centered content adaptation system, (2) a data-driven multi-level hidden Markov model (HMM)-based approach to perform both video segmentation and video indexing in a single pass, (3) the formulation and implementation of a Multiple-choice Multi-dimensional Knapsack Problem (MMKP)-based video personalization strategy, (4) the multiple-stage client request aggregation strategy that reduces the mean client-experienced latency without significant reduction in the

average relevance of the delivered video content to the client's request, and (5) a client-side energy-aware multimedia streaming strategy to efficiently utilize client's battery power. The overall framework of the system is modular and extensible. New techniques can be incorporated into the individual subsystems without changing other parts and the overall architecture of the system.

INDEX WORDS: Multimedia personalization, Video personalization, Mobile computing, Video Segmentation, Video Indexing, Hidden Markov Model, Knapsack problem, Multidimensional multiple-choice knapsack problem, Request aggregation, Clustering, K-means, Hierarchical clustering, Linear prediction, Multimedia streaming

VIDEO PERSONALIZATION FOR RESOURCE-CONSTRAINED ENVIRONMENTS

by

YONG WEI

M.S., University of Georgia, 2000

M.S., Nanjing University, China, 1994

B.S., Nanjing University, China, 1991

A Dissertation Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

ATHENS, GEORGIA

2007

VIDEO PERSONALIZATION FOR RESOURCE-CONSTRAINED ENVIRONMENTS

by

YONG WEI

Major Professor:	Suchendra M. Bhandarkar
Committee:	Hamid R. Arabnia Kang Li Xiangrong Yin

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
August 2007

DEDICATION

This dissertation is dedicated to my parents, Dr. Baoming Wei, Dr. Yuzhu Qiu, my wife, Dr. Xiuping Tao and my son, Franklin. Their deepest love has made this work possible.

ACKNOWLEDGEMENTS

I would like to thank my major professor, Dr. Suchendra Bhandarkar for his continuous guidance and support along the way.

I am also grateful for Dr. Hamid R. Arabnia, Dr. Kang Li and Dr. Xiangrong Yin for their serving on my advisory committee and their help.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER 1

INTRODUCTION

This dissertation is a report of an investigation into the problem of mobile client-centered multimedia personalization in resource-constrained environments, specifically on the sub-domain of video personalization. Each step taken in the development of a software system to support dynamic creation of personalized multimedia content is described.

1.1 **Background**

Recent developments in mobile computer networks and the advancement of mobile devices now allow for the usage of multimedia in mobile applications. The proliferation of the variety of devices such as cell phones, pocket PCs and PDAs has created the need to provide these devices with access to multimedia contents traditionally achieved on more powerful desktop machines. At the same time, applications aim more and more to provide personalized and adaptive content and services to better meet the client's need [Brusilovsky, 1998], [Lemlouma, 2003]. It is a clear trend to mark off the transition from the one-size-fits-all parading to a one-to-one addressing of the client's needs [Kopf, 2006], [Mohan, 1999]. The necessity to develop a client-centered application becomes even more important for a mobile user. Not only the still limited multimedia capacities of the mobile devices and network conditions but also the heterogeneity of these devices and the user's mobility itself requires to be considered by the mobile multimedia application.

The heart of a multimedia personalization system is to include the mobile user/client in the center when designing the technological basis for bringing multimedia content to a mobile user. The multimedia personalization application remains in the background and requires minimum interactions from mobile users while the mobile user carries out a task such as watching the weather forecast of the destination the user is driving to. The capabilities of these mobile devices vary widely and are limited in terms of network connectivity, processor speed, display constraints, and decoding capabilities. It is a real challenge to implement a universally compliant system that fits various usage environments.

1.2 **Research Challenges of Mobile Multimedia Personalization**

In the design of the mobile client-centered multimedia personalization system, the different aspects of user influences need to be considered. Depending on the natures of an actual application, various characteristics of an individual client such as content preferences, the knowledge of a certain domain and special needs have to be obtained and exploited by the system. This information is termed as a *user profile*. The user profile influences the multimedia content and format the personalization system selects and delivers.

A mobile client is constrained by his/her usage environment. A usage environment includes conditions of viewing time, location, bandwidth of connection, battery power capacity, screen resolution and CPU computing power. This information is termed as *usage profile*. The client environment influences as well the selection and presentation of multimedia information.

Figure 1.1 illustrates the general process of dynamically generating personalized multimedia content. Inputs to the process are multimedia data, the associated metadata, as well as user profile and usage profile. The personalization engine consists of the two shadowed modules, i.e. context dependent multimedia selection and composition modules. On a request, the

personalization engine exploits the user and usage profiles and metadata to choose the most suitable multimedia data. The selected multimedia data best suits the client's preferences and satisfies the constraints imposed by the client usage profile.

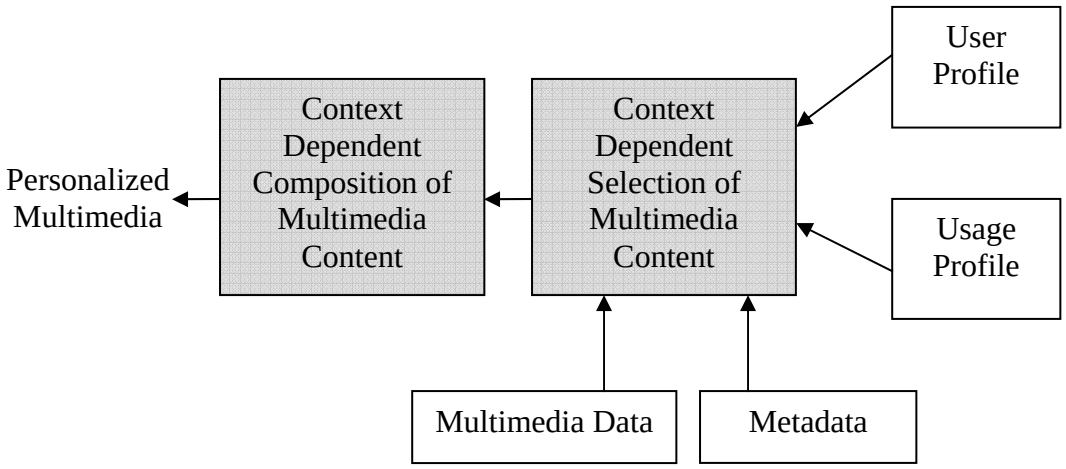


Figure 1.1 Process of Client-centered Multimedia Personalization

The necessity to reflect the user profile and usage profile in the process of multimedia personalization affects the entire process from multimedia data acquisition to the delivery of multimedia content to the mobile clients. The chain of stages of the multimedia personalization for mobile clients is shown in Figure 1.2. A similar diagram is presented by Boll [2005].

- Acquisition of Multimedia Data: In addition to capturing multimedia data, it is desirable that the environmental context metadata is associated with multimedia content at the time of capturing. Geographical location information of capturing devices, time, and user descriptions metadata can be facilitated later in video annotation [David, 2004] and content-based retrieval. The challenge at this stage is to associate context metadata early

in the process. In some situations, it is difficult to obtain or select an appropriate set of metadata information to be used at the stage of data acquisition.

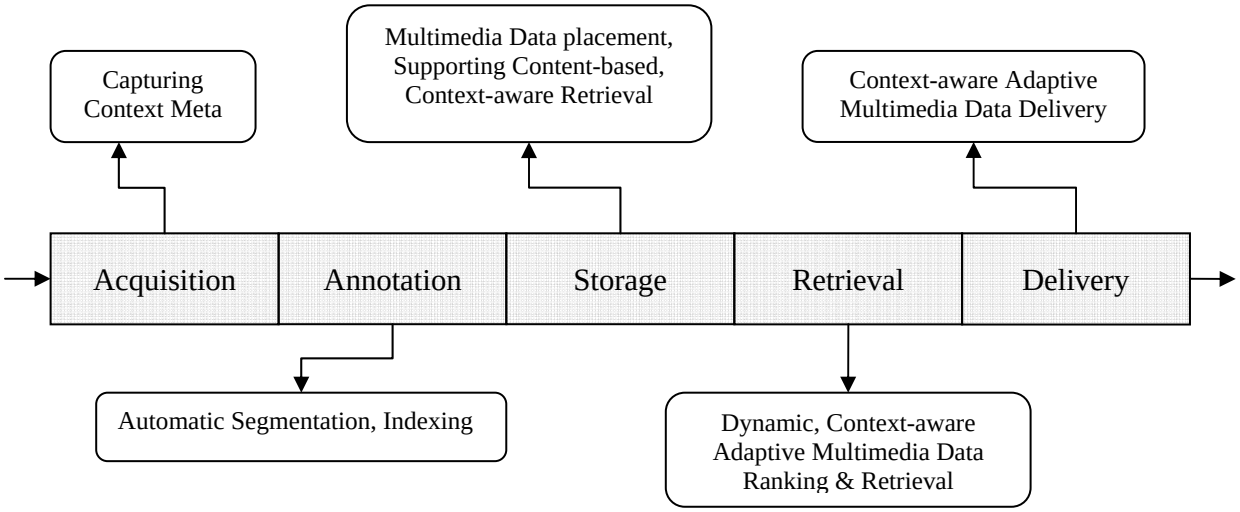


Figure 1.2 Stages of the Multimedia Personalization for Mobile Clients: Challenges and Opportunities

- Multimedia Data Annotation: Based on the acquired data, additional annotation is needed to associate additional metadata for later retrieval and usage. The challenge here is to derive as much user and usage-profile related metadata as possible to facilitate sophisticated client-centered multimedia content ranking and retrieval. Furthermore, humans tend to use high-level features (concepts), such as keywords, text descriptors, to interpret images and measure their similarity, whereas the features automatically extracted using computer vision techniques are mostly low-level features (color, texture, shape, spatial layout, etc.). In general, there is no direct link between the high-level concepts and the low-level features [Naphade, 2002], [Sethi, 2001].

- Data Storage: The annotated multimedia data needs to be organized and placed in a way that it can reflect client interest/preference and the mobile device's system parameters. Only then can a client request determined by user preference and usage environment parameters be efficiently and optimally answered [Bunningen, 2004]. The storage architecture of the annotated media data should be able to facilitate similarity comparison to rank candidate contents based on client preferences.
- Retrieval: To fulfill a request from a client, both the user profile and usage profile have to be considered in multimedia retrieval. The retrieval needs to integrate the client's preferences, device system and usage environment parameters in the process of choosing and compositing the multimedia response to the client's request. This is a constrained optimization problem. The challenge of solving this problem is to find an optimal solution to the constrained optimization problem, and, in the meantime, the solution should be suitable in the context of multimedia personalization [Wei, ACM TOMCAAP]. Another challenge is that in order to rank candidate multimedia data, similarity between client preferences and candidate media data needs to be measured [Vasconcelos, 2005], [Yu, 98].
- Delivery: The client-centered multimedia content delivery needs to address the issues of data placement, caching and distribution of the multimedia data. When the clients are resource-constrained, multimedia streaming should utilize client resources efficiently.

1.3 **Existing Approaches**

Multimedia document personalization tools such as SMIL [Ayars, 2001] allow the specification of adaptive multimedia presentation by the so-called "presentation alternatives". SMIL is a declarative XML-based language. The manual authoring of such context adaptive multimedia

documents are complex. Boll et al. [Boll, 1999] propose an approach that a multimedia document is enriched by the different presentation alternatives needed for the user contexts in which the document is to be viewed. However, this approach is limited by the presentation complexity.

In the area of dynamic generation of personalized multimedia, early systems are mainly text-focused (such as Amazon.com). Lemlouma et al. [Lemlouma, 2003] propose to use a multimedia processing architecture to adaptively transform multimedia content to meet the user context. The adaptation process chooses the version with the smallest data size as a response to the client request. No constrained optimization is utilized in the adaptation process to provide the user with a best solution under user preference and constraints.

Kopf et al. [Kopf, 2006] propose a color adaptation algorithm for videos in order to make them suitable for various mobile devices. The color depth of a video is adapted to facilitate the playback of videos on mobile devices which support only a limited number of different colors.

MobiCon [Lahti, 2005] integrates video clip capturing with context-aware, personalized clip annotation to support keyword-based video sharing for mobile phone clients. The video clip annotation process is semi-automatic, i.e., automatic annotation suggestions based on context data and manual annotation with user-specific keywords. The limitation of MobiCon is that expensive human interaction is needed to annotate video clips, and the generation of video content is only determined by client's preference. No user usage profile parameter is considered in the process of personalization.

Tseng et al. [Tseng, 2004] propose a video personalization system which integrates a VideoAnnEx video annotation sub-system [Lin, 2003], and a VideoSue video personalization [Tseng 2002] sub-system to dynamically generate and deliver personalized video summaries to

mobile clients. VideoAnnEx can only detect shot boundaries based on image histogram difference, and indexing is done manually by selecting semantic terms from a video content description lexicon. It is very expensive to annotate a large amount of video data manually. Video candidate ranking is done by a simple voting algorithm, no semantic similarity measure is utilized to compare the client video content preference and the semantic labels of video segments. Video personalization in this approach is 0/1 Knapsack Problem (0/1 KP)-based. Client-side resource capacity is not optimally utilized. This approach cannot support multiple client-side constraints simultaneously.

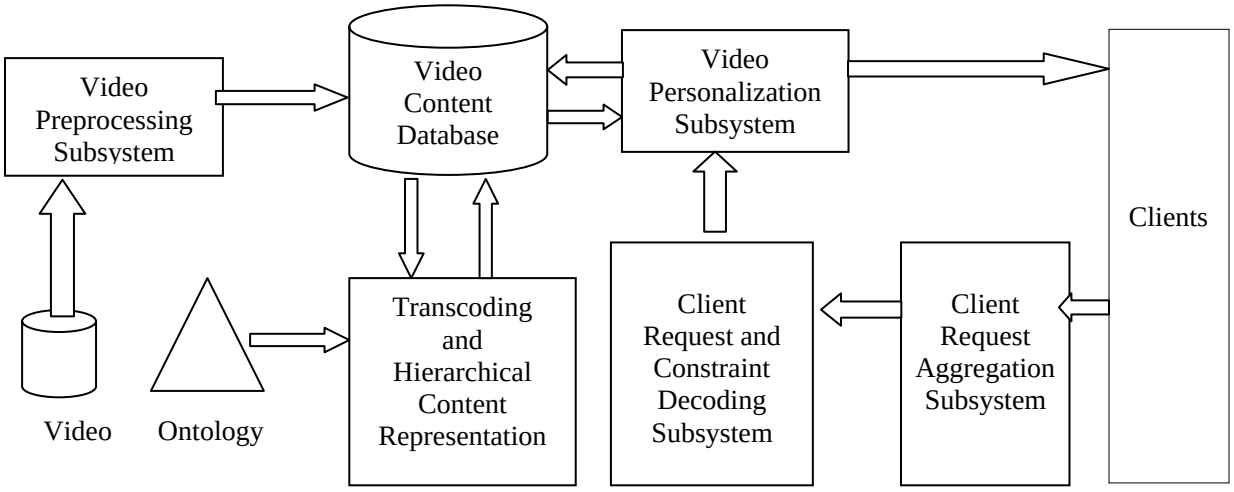


Figure 1.3 Overall System Architecture

1.4 System Architecture

The client-centered multimedia adaptation system consists of the following five subsystems, as shown in Figure 1.3: (1) the video preprocessing subsystem, (2) the hierarchical video content representation and multi-level video transcoding subsystem, (3) the client request and client

constraint decoding subsystem, (4) the video personalization subsystem, and, (5) the client request aggregation subsystem. The relationships amongst the five aforementioned subsystems are described in the following subsections.

1.4.1 **Video Preprocessing Subsystem**

The video preprocessing subsystem performs temporal video segmentation and video indexing. In order to provide mobile clients with personalized video content, the original video streams are first segmented and indexed in the temporal domain. A data-driven stochastic algorithm based on multi-level Hidden Markov Models (HMMs) is proposed to perform both video segmentation and video indexing automatically in a single pass.

1.4.2 **Multi-level Video Transcoding and Hierarchical Video Content Representation Subsystem**

Each indexed video segment is transcoded at multiple levels of abstraction. In the proposed scheme, semantic-level transcoding based on key frame selection and motion panorama computation and low-level transcoding based on bit rate reduction, and temporal and spatial resolution reduction are closely integrated. The original video segment and its transcoded versions are deemed to constitute a multi-level *content group*. To facilitate efficient content-based retrieval, a hierarchical ontology-based description of the video content is employed. A multi-level *content group* is associated with a set of appropriate semantic terms derived from the aforementioned ontology.

1.4.3 **Client Query and Constraint Decoding Subsystem**

The client query and client constraint decoding subsystem acts as an intermediary between the video personalization subsystem and the mobile client. A client query (request) consists of the client's preference(s) with regard to video content and a list of client-side resource constraints. A

client query protocol is established to facilitate the communication of the query between the client and the subsystem. A client query under the currently implemented protocol is a structure with two fields: *PREFERENCES* and *CONSTRAINTS*. The *PERFERENCES* field is a list of strings representing semantic terms that encapsulate the client's request for information whereas the *CONSTRAINTS* field is a list of numerical parameters representing the client-side resource constraints such as the viewing time limit, bandwidth limit and the limit on the amount of data the client can receive. Client queries transmitted in the format specified by the above protocol are received and subsequently decoded by the subsystem. The decoded query is then forwarded to the video personalization subsystem for further processing.

1.4.4 **Video Personalization Subsystem**

The goal of video personalization is to display a video summary that preserves as much of the semantic content desired by the client as possible while simultaneously satisfying the resource constraints imposed by the (potentially) mobile client. In the video personalization subsystem, the client's video content preference(s) is (are) matched with the video segments (and their various versions) stored in the video database. In order to generate a personalized video summary, the client usage environment and the client-side resource constraints are evaluated. The personalization engine compiles an optimal video summary that is most relevant to the client's content preference(s) subject to the resource constraints imposed by the client.

1.4.5 **Client Request Aggregation Subsystem**

When there are a large number of clients sending their requests to the server, the average client-experienced latency is long if every client request is processed individually. The proposed client request aggregation strategy clusters similar client requests together so that the number of requests sent to the server is reduced, reducing the average client latency. The client requests are

heterogeneous in multiple dimensions, i.e., they are different in their video content preferences, and in client-side constraints. A multi-stage clustering strategy is proposed to group similar request together one dimension at a time.

1.5 **Structure of the Dissertation**

After the introduction, the work in this dissertation is divided into five chapters. An outline of the remaining chapters is given below.

Chapter 2: The problem of semantic-based video segmentation and indexing is reviewed in depth in this chapter, with particular attention on hidden Markov model (HMM)-based approaches. After reviewing the state of the art in the field, concepts of HMM and multiple-level HMM are discussed. Implementation details of the multiple-level HMM-based video indexing are given. Video indexing experiment results are presented and discussed in this chapter.

Chapter 3: The state of the art in the field of video personalization is reviewed in depth, with an emphasis on the Knapsack Problem (KP)-based approaches. The multiple-choice Multi-dimensional Knapsack Problem (MMKP) is investigated with attention to its application in video personalization. The MMKP-based Video personalization experiment results are compared with the personalization strategies.

Chapter 4: A multi-stage clustering-based client request aggregation strategy is developed. The client-experienced system performances with and without the aggregation are compared.

Chapter 5: A client-side energy-aware multimedia streaming algorithm is investigated. Experimental results are compared with the existing history-based prediction algorithm.

Chapter 6: The primary contributions that this work has made are presented. Possible future research directions are discussed as a result of the investigation in this dissertation.

CHAPTER 2

MULTI-LEVEL HIDDEN MARKOV MODEL AND VIDEO SEGMENTATION AND INDEXING

2.1 **Introduction**

Semantic video indexing is regarded as the first step towards automatic retrieval and personalization of video data since it enables users to access videos based on their interests and preferences regarding video content. It is the process of attaching concept terms from a video descriptive ontology to segments of a video. Until now, video indexing is mostly carried out manually by assigning a limited number of keywords to the video document. The manual nature of the work makes indexing of video documents an expensive and time consuming task. Therefore automatic classification of video segments is necessary.

The content of a video is multimodal, i.e. the content of a video segment can be represented by low-level visual features, and semantic-level feature. Furthermore, auditory and textual features are also used to represented content of video segments. Therefore, in order to take the full advantage of the multimodal information, it is necessary to integrate these multiple features together to improve performance of video segmentation and indexing. Hidden Markov Models [Manning, 1999] [Rabiner, 1989] are frequently used as a statistical classification method for multimodal integration and classification. This feature makes HMM a suitable tool for multimodal video segmentation and indexing.

Video segmentation and indexing typically includes two sub-processes, temporal segmentation of the video stream and semantic labeling of the resulting video segments. These

two sub-processes are usually performed as two separate steps. For example, in IBM's VideoAnnEx video annotation system [Tseng, 2004], the video stream is first segmented into shots. An annotator then manually associates shots with terms selected from a predefined lexicon. For large amounts of video data, the manual annotation process involves intense human interaction and is extremely time consuming.

In this work, we propose a data-driven multi-level HMM-based approach to perform both video segmentation and video indexing in a single pass. The proposed approach uses only the visual features in a video stream in order to avoid potential audio-visual mismatches. The proposed approach is purely data-driven, i.e. no-domain specific knowledge about the structure of the video program is needed to syntactically or semantically model the video content.

The remainder of the chapter is organized as follows. Section 2.2 provides a general overview of video segmentation and indexing, reviewing the current state of the art. Concepts of HMM are discussed in Section 2.3. Section 2.4 describes the image features (especially the Tamura features) used in the construction of the video semantic unit level HMMs. In Section 2.5, we describe the construction of HMMs for the video semantic units, and the organization (via concatenation) of the individual HMMs based on a video program model. The data-driven video program model learning approach is detailed. Section 2.6 explains the performance measures used and the experimental results obtained. Section 2.7 concludes our work with an outline for future research.

2.2 **Video Segmentation and Indexing: A Review**

2.2.1 **Video Segmentation**

Early video segmentation methods compare pixels of successive image frames. Pixels of consequent frames can be compared pair wise. Dissimilarity between two successive frames is

measured to detect shot boundaries. Luminance pixel-wise difference is calculated and various shot change detection methods are used to find the location of video shot boundaries. When the $difference \geq Threshold$, a boundary is detected [Nagasaka, 1991]. The evolution of temporal derivative of the pixel intensities [Taniguchi 1997] can be used as a criterion for shot change detection. Temporal derivative evolution is a measure of difference over many frames. The image frame is filtered by a Gaussian mask to calculate temporal derivative of pixels. Pixel level comparison is a costly and is sensitive to minor camera operation like zooming. A more robust method is histogram comparison.

Histogram comparison-based video segmentation compares two successive image frames based on global histograms. A shot change is detected if the histograms of two consequent frames differ significantly [Wactlar, 1996]. An advantage of the histogram level information-based segmentation is that it uses frame-wide information. Thus it is more robust to camera movement and luminance changes. The straight forward histogram-based difference is to calculate the gray-level histograms of two successive image frames [Tonomura, 1990]. Histograms of color spaces, such as RGB, YIQ, etc are used to calculate the difference too [Gargi, 1996], [Pye, 1998].

In color images, some color components may have more influence than others [Dailianas, 1995]. In the proposed approach, a weighted histogram difference of two successive frame *f* and *f'* which is defined as follows:

$$d(f, f') = \frac{r}{s} d(f, f')_{red} + \frac{g}{s} d(f, f')_{green} + \frac{b}{s} d(f, f')_{blue} \qquad (2.1)$$

where r, g, b are luminance for the red, green and blue component of the picture respectively. $s = (r + g + b)/3$. Dissimilarity of images is measured by color histogram intersection. Given

two histograms, I_i and I_j, each containing n bins, the normalized match index of the intersection of histograms is defined as follows [Lee, 2005].

$$dis(i, j) = 1-\sum_{k=1\sim n} \min(I_{i,k,} I_{j,k}) / \sum_{k=1\sim n} I_{j,k} \qquad (2.2)$$

Block comparison-based segmentation is robust to noise and luminance changes. Similarity measure is performed on block-sampled images [Kasturi, 1991]. Block mean and variance of pairs of block with the same spatial coordinates in image frames are compared. RGB images are converted to HSV space [Lee, 2001]. Then the mean and values of Hue and Saturation are calculated for each block. Difference of two successive blocks is the block mean difference. Block histograms can be used to calculate the difference too [Swanberg, 1993].

More sophisticated features such as edges, contour of objects, planar points, moment, Tamura features are used to catch the time-varying characteristic of image frames in a video stream.

- Edges

Zabih et al [Zabih, 1999] use extracted edges of an image frame to segment videos. Two consecutive frames are converted into binary edge images. An entering edge pixel is an edge pixel that appears far from an existing edge pixel. An existing edge pixel is an edge pixel that disappears far from an existing edge pixel. Shot boundaries are detected and classified as cuts, fades and dissolves by counting the numbers of entering and exiting edge pixels.

- Focus of Expansion Points

Ardebilian et al [Ardebilian, 2000] detect changes of focus of expansion (FOE) of two successive images to detect shot boundaries. Contour detection is done with Deriche filtering. The FOE is extracted using double Hough transformations (DHT) applied to the contour images. The positions of these FOE points are used as indices of shot segmentation in a video sequence.

- Tamura features

Tamura features [Tamura, 1978] are used to capture the texture characteristics of image frames at human perception level. Tamura contrast, Tamura coarseness and Tamura directionality are successfully used in content-based image retrieval.

- Statistical Features

Principle component analysis (PCA) is applied to video segmentation [Yilmaz, 2000]. Rows of a RGB image frame are concatenated into a row vector. A 3×3 covariance matrix of the RGB color space is calculated. For two successive frames, angles between principle axes are defined as follows.

$$S(X,Z) = (X^T \bullet Z)/(\|X\| \bullet \|Z\|) \qquad (2.3)$$

where X and Z are principal axes of succeeding frames RGB color spaces and $S(X,Z)$ denotes the angle of rotation between the two successive frames. For frames within a shot, $S(X,Z) < Threshold$.

Singular value decomposition (SVD) is use to perform shot detection [Gong, 2000]. An image frame is divided into 3×3 blocks. A 125-bin histogram on the RGB space is calculated. SVD is performed on the feature vector. The K largest singular values are used to calculate similarity among frames.

- Motion Features

Motion can be used as an important feature to detect video shot. Various techniques using motion feature of image sequence are reviewed as follows.

Image frames are divided into 8×8 blocks [Akutsu, 1992]. Motion vectors among matching blocks of successive frames are calculated. For each block, average inter frame correlation coefficients are computed. This value represents similarity between frames. Motion smoothness

is defined as ratio of velocity to motion in each frame. A shot boundary is detected at local maximum of motion smoothness. Porter et al [Porter, 2000] propose a motion-based method in frequency domain to perform video shot detection. Frames are divided into 32×32 blocks. For a give block in frame *n*, the matching block in frame *n+1* is found using the normalized correlation. For computation simplicity, the correlations are calculated in the frequency domain.

High-level semantic features of video segments are detected and utilized in the process of video segmentation and indexing.

- People detection

People in video documents are detected by means of their faces and other body parts. Face detection techniques aim to identify all image regions which contain a face. If a face is detected, image location of the face is returned [Yang, 2002]. Based on the evaluation of Pham [Pham, 2000], the neural network-based system [Rowley, 1998] is the best.

Not only the head, but the whole human body is detected [Mohan, 2001]. The algorithm first locates the constituent components of the human body, such as head, legs and arms. Each individual detector is based on the Haar wavelet transform. After ensuring that these components are present in the proper geometric configuration, a second example-based classifier combines the results of the component detectors to classify a pattern as either a person or non-person.

- Object Detection

Object detection is a generalization of the people detection problem. Visual appearance of specific objects is used to detect the presence of passenger cars in image frames by using a product of histograms [Schneiderman, 2000]. Each histogram represents the joint statistics of a subset of wavelet coefficients and their positions on the object. Passenger cars are detected by using statistical modeling to account for variation.

Since the appearance of objects might vary widely among image frames, object motion detection is the most valuable feature in this case. A typical method to detect moving objects of interest starts with a segmentation of the image frame. Regions in the image frame sharing similar motion are merged in the second stage. An image frame is segmented into the motion-based regions [Nguyen, 2000].

2.2.2 **Semantic Video Indexing**

Video stream data can be viewed as a hierarchy. At the lowest level, video stream data is made of frames. A collection of image frames taken by a single camera operation, focusing on one object or one event is termed a *shot*. However, just as a phoneme can appear in many different words, visually similar video shots can appear in different video segments with different semantic meanings. Thus video shot segmentation by itself cannot support content-based video retrieval at a semantic level.

Definition 2.1: A *semantic unit* within a video stream is a video segment that can be associated with a clear semantic meaning or concept, and consists of a concatenation of semantically and temporally related video shots or video scenes.

Instead of detecting video shots or scenes, it is often much more useful to recognize semantic units within a video stream to be able to support video retrieval based on high-level semantic content. Note that visually similar video shots or video scenes may be contained within unrelated semantic units. Thus, video retrieval based purely on detection of video shots or video scenes will not necessarily reflect the semantic content of the video stream.

In recent years, various applications of HMMs to video segmentation and indexing have been studied. A clear advantage of the HMM-based video segmentation is that it is capable to integrate multimodal features easily [Huang, 1999]. Nam et al [Nam, 1998] propose to integrate

both the visual and auditory features of video stream to detect and index violent scenes in TA drama and movies. Huang et al. [Huang, 2000] use both audio and visual features in an HMM-based scheme to perform the video scene recognition. Li et al. [Li, 2001] propose a HMM framework to detect *play* events in sports videos. [Eickeler, 1999] and [Chaisorn, 2003] use an HMM-based predefined program model to index news programs. In the aforementioned works, however, the system performance could be compromised due to audio-visual mismatch [Huang, 2000] and inaccurate domain-dependent knowledge about the video scenes and the video program structure [Li, 2001],[Eickeler, 1999].

2.3 **Hidden Markov Model**

The HMM is defined following notations used by Rabiner [Rabiner, 1989], [Rabiner, 1993]. Assume the stochastic process to be modeled is governed by a finite number of states, $S = \{s_1,...,s_N\}$. The actual state at time t is denoted as q_t. The emission generated by a PDF dependent on the current state q_t can be observed. The PDF is denoted as $b_j(o_t)$, where o_t is the observable emission, under state j. $O = \{o_t : t = 1,...,T\}$ is a complete observation sequence generated by a HMM, denoted as λ. It is assumed that the current state q_t in a HMM is only dependent on the previous state q_{t-1}. This is called the Markovian assumption.

A HMM λ is denoted as follows

$$\lambda = (A, B, \pi) \qquad (2.4)$$

where A is the state transition matrix, B is the set of emission probabilities and π is the initial state distribution probabilities. A defines the transition behavior between states, as well as the topology of the HMM. The transition probability for moving from state *i* to state *j* is defined as

$$a_{ij} = P(q_t = j \mid q_{t-1} = i), 1 \le i, j \le N \qquad (2.5)$$

Each state of the HMM has a PDF, b_j, defining the probability of generating an observation emission o_t at time *t*. For this dissertation, the PDF for each state is assumed to be a mixture of Gaussian components, defined as

$$b_j(O) = \sum_{k=1}^{M} c_{jk} \mathrm{N}(O; \mu_{jk}, \sum_{jk}), 1 \le j \le N \qquad (2.6)$$

where M is the number of Gaussian components, μ_{jk} is the mean of mixture of Gaussian component k for the PDF of state j, and $\sum_{jk}$ is the covariance matrix. c_{jk} is the weighting term for each Gaussian component where

$$\sum_{k=1}^{M} c_{jk} = 1, 0 \le c_{jk} \le 1, 1 \le j \le N, 1 \le k \le M \qquad (2.7)$$

Finally, the initial state distribution probability vector $\pi = \{\pi_j : j = 1, ..., N\}$ defines the probabilities of the HMM commencing at any state, given the observation sequence *O*.

Restriction in the transition matrix A defines the topology and behavior of the HMM. Bakis HMM restricts movement from the left states to right states only. Figure 2.1 displays a Bakis HMM with 3 states.

One problem with this approach is that the video features are continuous. To convert the video feature sequence into discrete form, methods such as vector quantisation are required. In this form, the continuous feature sequences become a sequence of discrete symbols generated from a known “code-book”. However, converting the continuous data into discrete form can result in serious degradation of the signal, and information loss [Rabiner, 1993]. The approach we take is to employ continuous density HMM models. For each state, $b_j(O)$ is represented by a

continuous PDF, with the most widely applied distribution being Gaussian. Mixtures enhance the power of a model by representing a complex distribution as a combination of simple Gaussian components. It is common practice to use a mixture of Gaussians for modeling each state, where the same number of mixture components is normally fixed across all states.

An objective of this dissertation is to identify video semantic units such as *Anchor*, *News*, *Weather* and *Commercial*. Little is known about the underlying physical processes which generate the observable visual features in the video stream. Therefore, restricting movement between these states, without prior investigation into the underlying generation process, is justified [Rabiner, 1993].

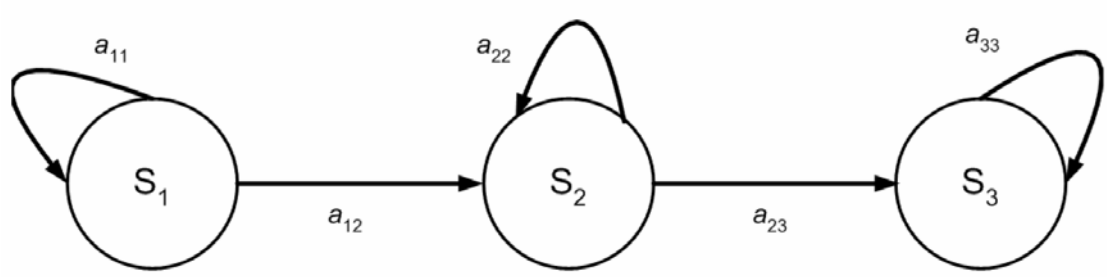


Figure 2.1 An Example of a 3 State Bakis HMM

2.4 **Hidden Markov Model and Video Segmentation and Indexing**

In continuous speech recognition systems, the continuous speech resulting from a spoken sentence is modeled at both, the acoustic-phonetic (sub-word) level and the language level. In most modern speech recognition systems, these sub-word units are modeled by Hidden Markov Models (HMMs) [Ney, 1999] which have been shown to be powerful stochastic models capable of approximating many time varying random processes [Rabiner, 1989]. Inspired by the success

of modern HMM-based continuous speech recognition systems and HMM-based video segmentation approaches [Eickeler, 1999], we propose a data-driven multi-level HMM-based approach to perform both video segmentation and video indexing in a single pass.

The multi-level HMM-based segmentation and indexing algorithm is essentially a stochastic model-based segmentation algorithm wherein the input video stream is classified frame by frame into *semantic units*. A semantic unit within a video stream is a video segment that can be associated with a clear semantic meaning or concept, and consists of a concatenation of semantically and temporally related video shots or video scenes. Temporal boundaries in the video stream are then marked at frame locations that represent a transition from one semantic unit to another. One of the advantages of the proposed multi-level HMM-based segmentation algorithm is that once the set of HMMs for a video stream are defined, future image sequences can be segmented, classified and indexed in a single pass. Furthermore, semantic units can be added without having to retrain the HMMs corresponding to the other semantic units. Thus, the proposed multi-level HMM makes it possible to process different types of videos in a modular and extensible manner so as to enable video retrieval based on semantic content.

Instead of detecting video shots or scenes, it is often much more useful to recognize semantic units within a video stream to be able to support video retrieval based on high-level semantic content. Note that visually similar video shots or video scenes may be contained within unrelated semantic units. Thus, video retrieval based purely on detection of video shots or video scenes will not necessarily reflect the semantic content of the video stream. The semantic units within a video stream can be spliced together to form a logical video sequence that the viewer can understand. In well organized videos, such as TV broadcast news and sports programs, the video can be viewed as a sequence of semantic units that are concatenated based on a predefined video

program syntax. Parsing a video file into semantic units enables video retrieval based on high-level semantic content and playback of logically coherent blocks within a video stream. Automatic indexing of semantic components within a video stream can enable a viewer to jump straight to points of interest within the indexed video stream, or even skip advertisement breaks during video playback.

In the proposed scheme, a video stream is modeled at both, the semantic unit level and the program model level. For each video semantic unit, an HMM is generated to model the stochastic behavior of the sequence of feature emissions from the image frames. Each image frame in a video stream is characterized by a multi-dimensional feature vector. A video stream is considered to generate a sequence of these feature vectors based on an underlying stochastic process that is modeled by a multi-level HMM. The advantages of the proposed approach are summarized as follows.

- **Video segmentation and video indexing are performed in a single pass**. This is extremely valuable when dealing with large amounts of video data to populate a video database. Although, video segmentation and video indexing are performed off line, they are computationally intensive and often result in a serious bottleneck during the creation of a video database. The ability to perform video segmentation and video indexing in a single pass alleviates this bottleneck to some extent.
- **No domain-dependent knowledge about the structure of video programs is used.** The probabilistic grammar used to define the video program is learned entirely from the training data. This allows the proposed approach to handle various kinds of videos in a modular and extensible manner without having to manually redefine the program model.

- **Semantic unit level HMMs are used to model video units with clear semantic meanings.**

The proposed data-driven approach does not need to use HMMs to model video edit effects. This not only simplifies the collection and processing of training data, but also ensures that all video segments in the video database are labeled with concepts with clear semantic meanings in order to facilitate video retrieval based on semantic content. Furthermore, although a semantic unit might include some video edit effects, these effects are considered part of the semantic unit and, as such, are not labeled separately. The HMM representation of a semantic unit can accommodate these edit effects implicitly.

2.5 Image Features for Semantic Units

The success of an HMM-based algorithm for video segmentation and video indexing depends greatly on the image features extracted from each frame in the video stream. These features should contain enough information about each image frame, yet should capture the differences amongst the frames in distinct semantic units in order to be able to distinguish them. In this work, we use two categories of image features. The first category includes a set of simple features. The dynamic characteristics of the image frames comprising the video stream are captured by the differences of successive image frames at both, the pixel level and the histogram level. Various motion-based measures describing the movement of the objects in the image frames are used, including the motion centroid of the image, and intensity of motion. Measures of illumination change at both, the pixel level and the histogram level are also included in the multi-dimensional feature vector. Definitions of these features are given in [Eickeler, 1999].

In the second category, Tamura features [Tamura, 1978] are used to capture the textural characteristics of the image frames at the level of human perception. Tamura contrast, Tamura coarseness and Tamura directionality have been used successfully in content-based image

retrieval [Flickner, 1995]. In our work, inclusion of these features is observed to improve the accuracy of temporal video segmentation and video indexing. Definitions of the Tamura features are given as follows:

Tamura contrast

Consider

$$k = \frac{1}{N}\sum_{i \in O}(c[i] - \mu)^4 \qquad (2.8)$$

where μ is the average of the color values in the neighborhood of pixel (x, y) denoted by $O(x, y)$, $c[i]$ is the color or intensity of the ith pixel in the neighborhood $O(x, y)$, and N is the number of pixels in the neighborhood $O(x, y)$. The Tamura contrast at pixel (x, y), denoted by *TCon(x, y)*, is given by

$$TCon(x, y) = \begin{cases} 0 & k < \varepsilon \\ \sigma^2 / \sqrt[4]{k} & otherwise \end{cases} \qquad (2.9)$$

where σ^2 is the color covariance computed in the neighborhood of pixel (x, y) in the image frame and ε is a predefined threshold.

Tamura coarseness

The Tamura coarseness measures the spatial scale at which the difference in color values between pixels in a local neighborhood of a given pixel (x, y) is a maximum. Given a pixel (x, y), 5 spatial scales are used to measure the horizontal and vertical differences of the mean color value. The horizontal difference and the vertical difference of the mean color values at location (x, y) at scale k are given by

$$E_H(x, y, k) = \left| A(x - 2^k, y, k) - A(x + 2^k, y, k) \right| \qquad (2.10)$$

$$E_V(x, y, k) = \left| A(x, y - 2^k, k) - A(x, y + 2^k, k) \right| \qquad (2.11)$$

where $k \in [0,4]$, and $A(x, y, k)$ is the mean color value at pixel (x, y) for window size $(2^k + 1) \times (2^k + 1)$ when $k > 0$. The window size is 1×1 when $k = 0$.

Let us define $E(x, y, k) = \max(E_H(x, y, k), E_V(x, y, k))$

The Tamura coarseness, denoted by *TCoar(x, y)*, at pixel (x, y) is given by

$$TCoar(x, y) = \arg\{\max_k(E(x, y, k))\} \qquad (2.12)$$

The definition of Tamura coarseness give above, calls for the computation of the mean color value $A(x, y, k)$ in 20 distinct windows if 5 spatial scales are used. The computation cost is high if we perform the summation directly by enumerating the color values of each pixel in each window. Hence, we need an efficient way to compute $A(x, y, k)$. The *integral image* [Viola, 2004] provides an efficient way to compute the summation of in a rectangular window. Given an original input image $I(x, y)$, the integral image is given by

$$J(x, y) = \int_0^y \int_0^x I(u, v) du dv \qquad (2.13)$$

For a discrete image $I(x, y)$, the integrals in equation (2.13) are replaced by their corresponding summations. The integral image $J(x, y)$ can be computed efficiently using the following recurrence relation

$$J(x, y) = I(x, y) + J(x-1, y) + J(x, y-1) - J(x-1, y-1) \qquad (2.14)$$

The mean color value within a given rectangular window (x_1, y_1, x_2, y_2), with corner points (x_1, y_1) and (x_2, y_2), can be computed as:

$$A(x_1, y_1, x_2, y_2) = \frac{J(x_2, y_2) - J(x_2, y_1) - J(x_1, y_2) + J(x_1, y_1)}{R} \qquad (2.15)$$

where R is the area size of the rectangular window (x_1, y_1, x_2, y_2). Thus the computation of the mean color value within a given rectangular window can achieved efficiently with three summation operations as shown in equation (2.15).

Tamura directionality

Tamura directionality is simply the intensity gradient orientation $\theta(x, y)$ at a pixel (x, y) and is given by

$$\theta(x, y) = \tan^{-1} \frac{I_y(x, y)}{I_x(x, y)} \qquad (2.16)$$

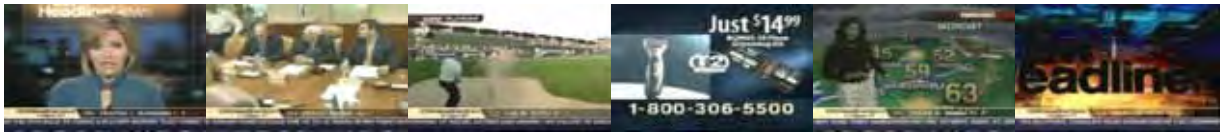
where the intensity gradient components $I_x(x, y)$ and $I_y(x, y)$ are computed using the *Sobel* edge operator.

2.6 HMMs for Characterization of Semantic Units in a Video Stream

In the proposed video segmentation and video indexing scheme based on semantic video content, we define six semantic concepts for TV broadcast news video, i.e. *News Anchor*, *News*, *Sports News*, *Commercial*, *Weather Forecast* and *Program Header*, and three semantic concepts for Major League Soccer (MLS) video, i.e. *Zoom Out*, *Close Up* and *Replay*. Representative images for each of these semantic concepts are shown in Figure 2.2(a)-(b). An HMM is formulated for each individual semantic concept. The optimal HMM parameters for each semantic unit are learned from the feature vector sequences obtained from the training video data. The standard HMM training procedure based on the Baum-Welch algorithm [Baum, 1970] is used. In our scheme, the HMMs for individual semantic units are trained separately using the training feature vector sequences. This allows for modularity in the learning procedure and flexibility in terms of being able to accommodate various types of video data. When new video data for a semantic unit

are presented, we only need to retrain the corresponding HMM for the relevant semantic unit without having to retrain any of the HMMs corresponding to the other semantic units.

Since the states in an HMM are hidden, researchers typically use heuristics to guess the correct HMM topology [Boreczky et al. 1998]. In our work, we adopt a universal left-to-right HMM topology (i.e., an HMM topology where no backward state transitions are allowed) with continuous observations of the feature vector emissions. The distribution of the feature vector emissions in the HMM is approximated by a mixture of Gaussian distributions. The number of Gaussian mixture components is fixed at three in all of our HMM implementations. The reason for the above HMM design choices is that in the case of actual video data, little is known about the underlying physical processes which generate the observable visual features in the video stream. Using a universal left-to-right HMM topology with a three Gaussian component mixture as a default choice makes it easy construct semantic unit HMMs for unknown data without prior detailed investigation into the underlying feature generation process [Eickeler et al. 1999], [Shinoda et al. 2005]. Furthermore, the above approach to HMM design can be used, without any modification, to recognize new semantic units in a video stream.



(a) TV News Broadcast: From Left to Right: *News Anchor*, *News*, *Sports News*, *Commercial*, *Weather Forecast* and *Program Header*

(b) MLS Video: From Left to Right: *Zoomed Out, Close Up* and *Replay*

Fig. 2.2 Representative Image Frames of Video Semantic Units

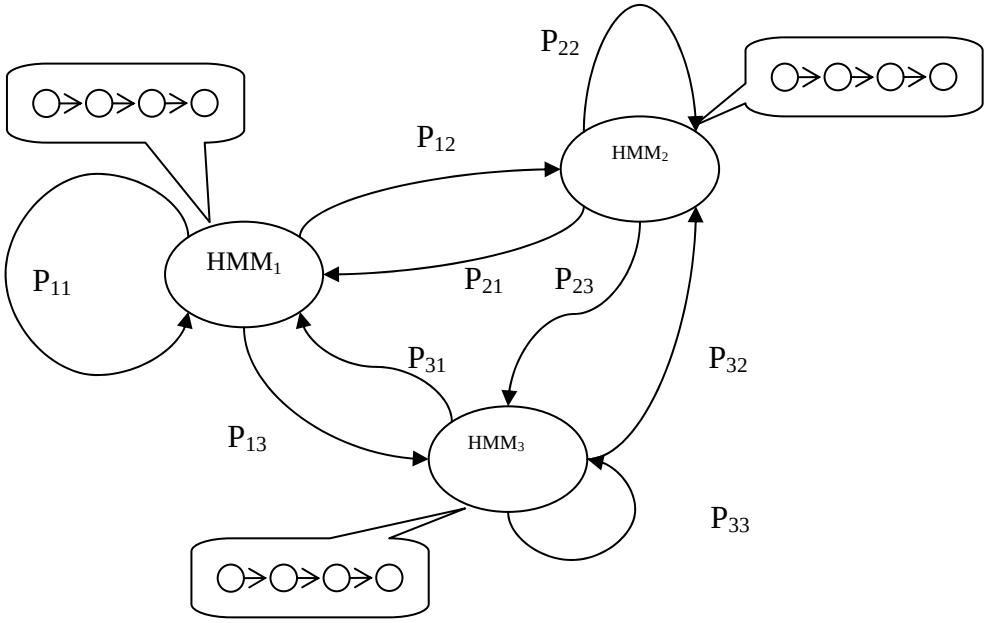


Fig. 2.3 Concatenation of the Individual HMMs

2.7 Multi-level HMMs for Single-Pass Video Segmentation and Indexing

The search space for the proposed single-pass video segmentation and video indexing procedure is characterized by the concatenation of the HMMs corresponding to the individual semantic units. The HMM corresponding to an individual semantic unit essentially models the stochastic behavior of the sequence of image features within the scope of that semantic unit. Transitions

amongst these semantic unit HMMs are regulated by a pre-specified video program model. Figure 2.3 depicts the concatenation of the individual HMMs corresponding to the three semantic units comprising the video program model. The topologies of the individual HMMs are described in the callouts in Figure 2.3. The parameter $p_{ij}, 1 \le i, j \le 3$ is the transition probability from semantic unit *i* to semantic unit *j*. The transition probability matrix $P_{3\times 3}$, where $P_{ij} = p_{ij}, 1 \le i, j \le 3$, essentially defines the video program model.

In this work, a data-driven approach is proposed to estimate the video program model directly from the training data using sequential maximum likelihood estimation, i.e., no domain-dependent knowledge about the structure of the video program is used. Most researchers typically use domain-specific knowledge about the video program in order to determine the video program model [Eickeler, 1999], [Huang, 2005], [Li, 2001]. This knowledge-driven approach becomes untenable as the size of the semantic unit vocabulary and the complexity of video program increase. The inaccuracy in the estimation of the video program model directly affects the segmentation and indexing results. In this work, the video program is represented by a 2-gram model determined by the conditional probability of the semantic unit sequence given a sequence of image feature vectors as shown in equation (2.17). Statistical language models (SLMs) can be typically represented by *n*-gram models [Brown et al. 1992]. When *n* is large, a correspondingly large amount of training data is required to estimate the *n*-gram model parameters, and the training process is computationally expensive. Hence in the proposed approach, a 2-gram model is chosen to represent the video program. The training data for estimation of the parameters of the video program model are assumed to be manually pre-labeled.

The single-pass video segmentation and video indexing procedure is formulated in terms of the following Bayesian decision rule: Given a sequence of image feature vectors $f_1 \ldots f_T$, determine a semantic unit sequence $U_1 \ldots U_N$ such that the conditional probability of the semantic unit sequence given the sequence of image feature vectors is maximized, i.e.,

$$\max(\Pr(U_1 \ldots U_N \mid f_1 \ldots f_T)) \sim \max(\Pr(U_1 \ldots U_N) \bullet \Pr(f_1 \ldots f_T \mid U_1 \ldots U_N)) \qquad (2.17)$$

In equation (2.17), $f_1 \ldots f_T$ are the feature vectors extracted from the image frames in the video stream to be segmented and indexed and $\Pr(U_1 \ldots U_N)$ is the video program model. The video program model regulates the transition probability from a predecessor semantic unit to a successor semantic unit. The Viterbi algorithm [Forney, 1973] [Viterbi, 1967] is used to determine the optimal path in the concatenation of the HMMs. Figure 2.4 depicts the single-pass segmentation and indexing is of a video stream containing the semantic units *A*, *B* and *C*. The y-axis represents the hidden states within the HMM for an individual semantic unit. The bold curves in Figure 4 indicate the change of states within the semantic units *A*, *B* and *C*. The video stream in this example is segmented into a semantic unit sequence *BACBA*. Bold curves within a semantic unit are monotonically non-decreasing because the HMMs for the individual semantic units have a strict left-to-right topology (i.e., backward-going state transitions are not permitted). Note that although a two-level HMM is used in the current implementation, the underlying technique for single-pass segmentation and indexing of video can be generalized to a multi-level HMM with any number of levels.

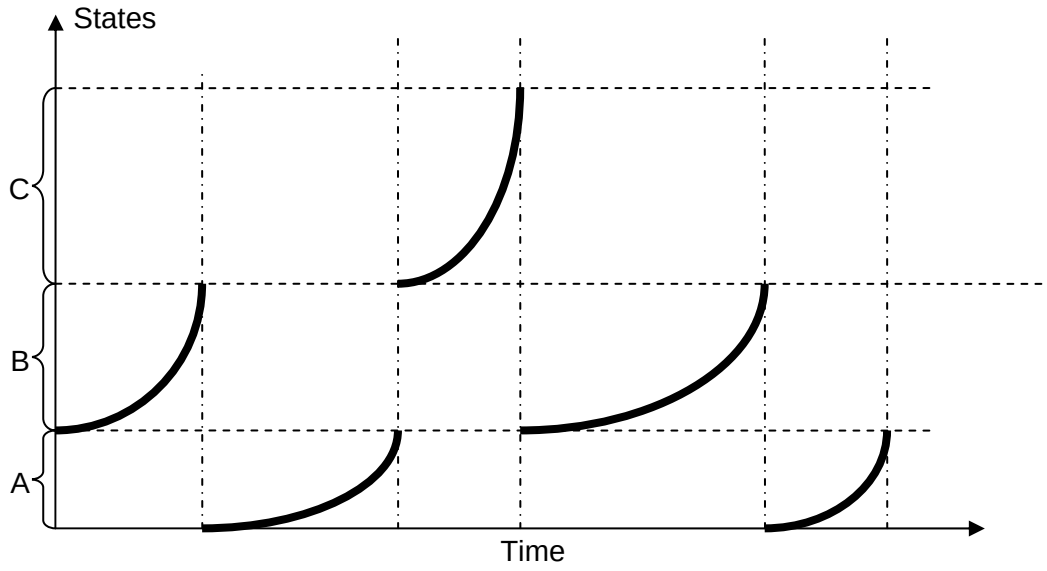


Figure 2.4 Sincle-pass Segmentation and Indexing of a Video Stream Containing Semantic Units *A*, *B* and *C*

2.8 Experiment Results of the Multi-level HMMs-based Video Segmentation and Indexing

We recorded 2 hours of the CNN Headline News program and 1.5 hours of the Major League Soccer (MLS) program respectively. The video streams were digitized to a frame resolution of 180 × 120 pixels with a frame rate of 30 frames per second. Sixteen minutes of the CNN video and one hour of the soccer video was reserved for testing. For generation of training data, the remainder of the CNN news video data was manually segmented into six semantic categories, *News Anchor*, *News*, *Commercial*, *Program Header*, *Weather Forecast* and *Sports News* and denoted by semantic concepts 1 through 6 respectively, and the MLS video data was manually segmented into three semantic categories, *Zoom Out*, *Close Up* and *Replay* and denoted by semantic concepts 1 through 3 respectively. A multi-dimensional feature vector was extracted for each image frame in the training video. For each of the semantic units, a left-to-right HMM

with continuous emission of observations was trained using feature vector sequences derived from the training video. To estimate the 2-gram video program model, the training video was manually labeled with labels selected from the aforementioned semantic concepts. The maximum likelihood estimation of the video program model was performed using the labeled training sequence.

The performance measurements for the above single-pass video segmentation and video indexing scheme comprise of two aspects, performance evaluation of video segment boundary detection and performance evaluation of video segment classification. To measure the performance of the video segment boundary detection algorithm, parameters such as insertion rate, deletion rate and boundary detection accuracy [Eickeler, 2000] were used. These parameters are defined as follows. The insertion rate $R_{insertion}$ denotes the fraction of unassigned boundaries in the detected boundaries. The deletion rate $R_{deletion}$ denotes the fraction of missed boundaries in the ground truth sequence boundaries. The boundary detection accuracy $Accuracy_B$ measures the average shift (in terms of number of frames) between the detected boundary and actual boundary locations. Thus

$$R_{insertion} = \frac{boundaries_{inserted}}{boundaries_{\det ected}} \tag{2.18}$$

$$R_{deletion} = \frac{boundaries_{missed}}{boundaries_{actual}} \tag{2.19}$$

$$Accuracy_B = \frac{\sum \Delta frames}{boundaries_{actual}} \tag{2.20}$$

where $\sum \Delta frames$ is the amount of shift (measured in terms of number of frames) between the detected boundary location and the actual boundary location.

To measure the video segment classification accuracy, we define the following metric:

$$Accuracy_C = \frac{S_{correct}}{S_{correct} + S_{false}} \qquad (2.21)$$

where $s_{correct}$ is the number of correctly indexed video segments and s_{false} is the number of incorrectly indexed segments. In our experiments, some incorrectly indexed segments were observed to be very short. These short segments were observed to contain a very small fraction of the total number of image frames in the video stream. Thus, the measure $Accuracy_c$ by itself does not reflect the classification performance because it treats very short and incorrectly classified segments on par with the relatively long and correctly classified segments. Hence, in order to measure the number of correctly classified image frames, we use a frame-based measure to determine the fraction of correctly classified image frames in the total number of frames as follows:

$$Accuracy_F = \frac{frames_{correct}}{frames_{correct} + frames_{false}} \qquad (2.22)$$

where $frames_{correct}$ and $frames_{false}$ are the numbers of correctly classified and incorrectly classified image frames respectively. $Accuracy_F$ thus represents the percentage of correctly classified frames. It measures the temporal classification accuracy, i.e. the relative duration of correctly recognized segments of a video stream. The algorithm provided by Eickeler et al. [2000] was used to compute performance measures in equations (2.18) - (2.22).

In Figure 2.5, the recognized semantic label sequence and the ground truth semantic label sequence of the CNN Headline News video stream are plotted against the frame number for the entire test video segment. For the news video, the single-pass video segmentation and video indexing algorithm was observed to detect most of the segment boundaries and label them correctly, except for some portion of the *Commercial* segment which was incorrectly classified.

In Tables 2.1 and 2.2, the numerical measures of performance for the single-pass video segmentation and indexing algorithm are tabulated. Figure 2.5 shows that most of the inserted boundary detection and false segment classification occurs during the *Commercial* segment (semantic concept ID=3). This is because of the complex nature of the content of TV commercials. In TV commercial programs, there could be large video segments similar to those found in the other semantic units such as *News Anchor* and *Sports*. These scenes in TV commercials are visually similar to those in *News Anchor* and *Sports*, and hence are classified incorrectly by the algorithm.

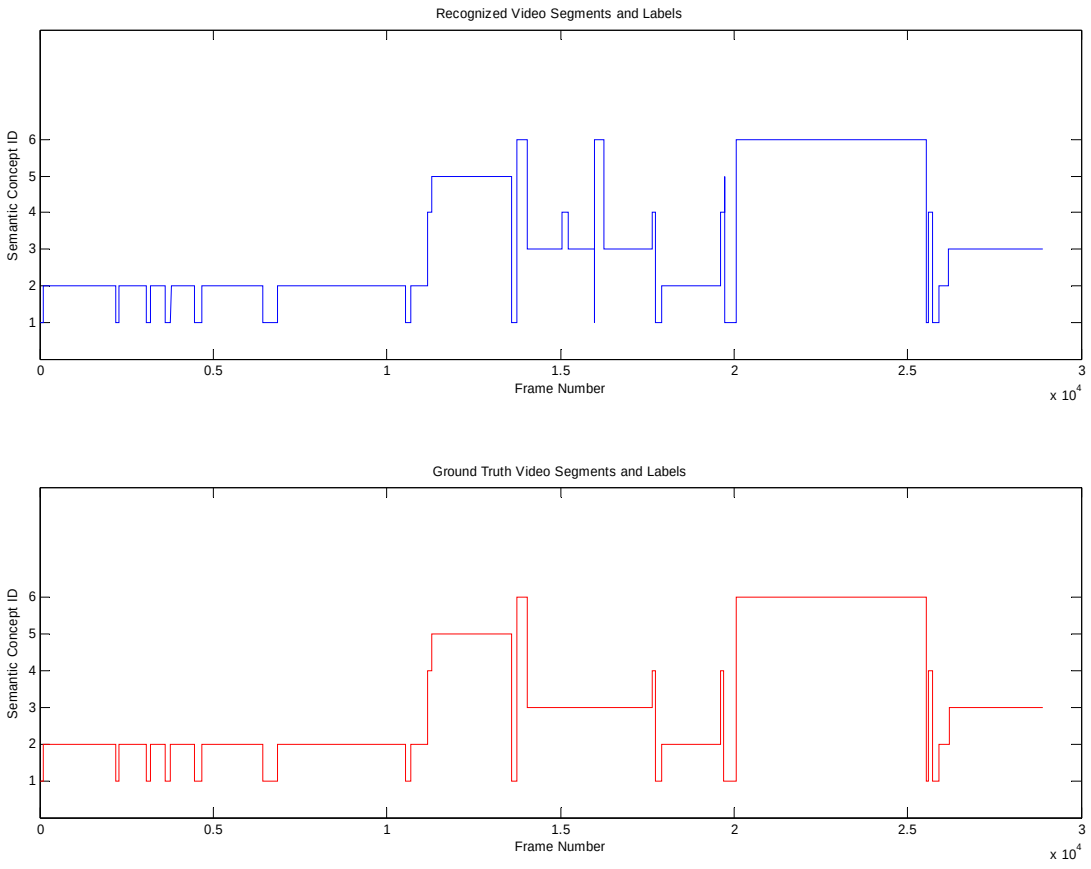


Figure 2.5 Frame Number vs. Recognized/Ground Truth Video Segments and Labels: CNN Headline News Video

Table 2.1 Performance Measures for Video Segment Boundary Detection: CNN Headline News Video

Actual Boundaries	Detected Boundaries	Inserted Boundaries	Deleted Boundaries	Insertion Rate (%)	Deletion Rate (%)	$Accuracy_B$ (Frame)
29	36	7	0	7/36=19.4	0/29=0	55/29=1.9

Table 2.2 Performance Measures for Video Segment Classification: CNN Headline News Video

Correctly Classified Segments	Incorrectly Classified Segments	$Accuracy_C$ (%)	Total Number of Frames	Correctly Classified Frames	$Accuracy_F$ (%)
32	5	86.5	28898	514	28384/28898=98.2

In Figure 2.6, the recognized semantic label sequence and the ground truth semantic label sequence are plotted against the frame number for the MLS test video segment. In Tables 2.3 and 2.4, the corresponding numerical measures of performance for the single-pass video segmentation and video indexing algorithm are tabulated in the case of the MLS test video segment. Experiment results show that the proposed multi-level HMM-based video segmentation and video indexing algorithm can segment and index MLS videos quite accurately.

Table 2.3 Performance Measures for Video Segmentation Boundary Detection: MLS Video

Actual Boundaries	Detected Boundaries	Inserted Boundaries	Deleted Boundaries	Insertion Rate (%)	Deletion Rate (%)	$Accuracy_B$ (Frame)
57	60	3	0	3/60 = 5	0/60 = 0	671/57=11.8

Table 2.4 Performance Measures for Video Segment Classification: MLS Video

Correctly Classified Segments	Incorrectly Classified Segments	$Accuracy_C$ (%)	Total Number of Frames	Correctly Classified Frames	$Accuracy_F$ (%)
59	2	96.7	31150	783	30367/31150=97.5

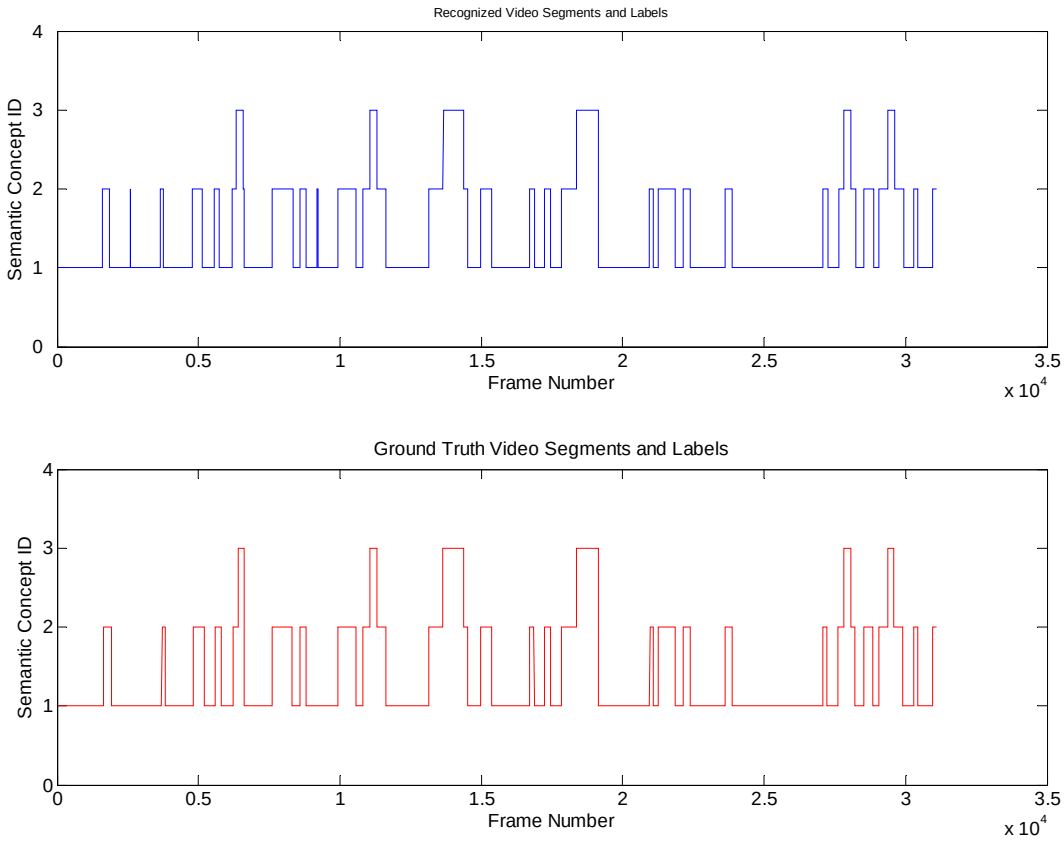


Figure 2.6 Frame Number vs. Recognized/Ground Truth Video Segments and Labels: MLS Video

The proposed multi-level HMM-based video segmentation and video indexing algorithm is implemented on a Dell Precision workstation with dual 3.19GHz CPUs and 2.0 GB of RAM. In the case of the CNN Headline News video, it takes 3076 seconds to extract feature vectors from the training data, 1394 seconds to train the HMMs for the individual video semantic units, and

1400 seconds to segment and index the CNN Headline News test video of duration of 962 seconds as presented in Figure 2.5. For the MLS video, it takes 1078 seconds to extract feature vectors from the training data, 873 seconds to train the HMMs for video semantic units, and 930 seconds to segment and index the MLS test video of duration of 746 seconds as presented in Figure 2.6. The video files and their transcoded versions are stored at a bit rate of 210kbps.

2.9 **Conclusions**

Inspired by the success of modern continuous speech recognition, we used Hidden Markov Models (HMMs) to stochastically model the input video streams at both the semantic unit level and the video program level. For each semantic unit within the input video stream, an HMM was established to model the stochastic process of emission of image feature vectors within the scope of that semantic unit. In our work, the semantic units were associated only with clear semantic concepts. Transitional scenes and special visual effects that lacked clear semantic meaning were not modeled using HMMs. This not only simplified the training data collection process, but also improved the robustness of the video segmentation and video indexing procedure.

The 2-gram video program model was used to define the transition probabilities amongst the various semantic units. The high complexity of real video programs, often renders a domain knowledge dependent definition of the video program model practically untenable. In our approach, a data-driven maximum likelihood estimation of the 2-gram program model from training data was observed to yield very good results. The individual HMMs for the semantic units were concatenated based on the video program model. Determining the optimal path through the concatenation of the HMMs corresponding to the individual semantic units was shown to result in a data-driven single-pass video segmentation and video indexing algorithm.

Experimental results showed that the resulting video boundary detection and video segment classification were highly accurate. The proposed multi-level HMM-based scheme was observed to be scalable and extensible since the program model could be altered by addition, deletion and modification (via retraining) of the HMMs corresponding to the relevant semantic units without having to retrain or alter the HMMs corresponding to the other semantic units.

CHAPTER 3

MULTIPLE-CHOICE MULTI-DIMENSIONAL KNAPSACK PROBLEM-BASED VIDEO PERSONALIZATION

3.1 **Introduction**

The current proliferation of mobile computing devices and network technologies has created enormous opportunities for mobile device users to communicate with multimedia servers, using multimedia streams. As handheld mobile computing and communication devices such as personal digital assistants (PDAs), pocket-PCs and cellular devices have become increasingly capable of storing, rendering and display of multimedia data, the user demand for being able to view streaming video on such devices has increased. For example, a mobile handheld client may be interested in viewing traffic conditions on the road and browsing the weather forecast for his or her travel destination. One of the natural limitations of these handheld devices is that they are constrained by their battery power capacity, rendering and display capability, viewing time limit and in many situations, by the available network bandwidth connecting these devices to video data servers. Therefore, the original video content often needs to be personalized in order to fulfill the client's request under various client-side system-level resource constraints (henceforth termed as "client-side resource constraints", in the interest of brevity).

Given the client's preference(s) regarding the video content, and the various client-side resource constraints, the video personalization system should be able to gather and disseminate the most relevant video content to the mobile client(s) while simultaneously satisfying multiple

client-side resource constraints. In light of the above, a definition of video personalization can be given as follows:

Definition 3.1: Given the client's preferences regarding the video contents, and given the client-side resource constraints, video personalization is the process of compiling and disseminating the most relevant video contents to the mobile clients while simultaneously satisfying client-side system-level constraints.

In this work, we present a client-centered video personalization system which can optimally fulfill the client's requests while simultaneously ensuring optimal utilization of client-side resources. While there are many challenges to be addressed in the design and implementation of a comprehensive video personalization system, the work presented in this chapter focuses on the design and implementation of video personalization strategies. The personalization problem is modeled as one of constrained optimization, i.e., maximization of the "total value" of the video summary delivered to the client under multiple constraints that represent the client's content preferences and system-level resources. Various personalization strategies based on the classical Knapsack Problem (KP) have been proposed in the literature. The contribution of the work presented in this paper is the design and implementation of a Multiple-choice Multi-dimensional Knapsack Problem (MMKP)-based video personalization strategy which is shown to have significant advantages over the existing 0/1 Knapsack Problem (0/1KP)-based and the Fractional Knapsack Problem (FKP)-based video personalization strategies. The proposed MMKP-based personalization strategy is observed to include more relevant video content in response to a client's request compared to the existing 0/1KP-based and FKP-based personalization strategies. In contrast to the 0/1KP-based and FKP-based personalization strategies which can support only a single client-side constraint at a time, the proposed MMKP-based personalization strategy is

shown to be capable of supporting multiple client-side constraints simultaneously.

The proposed MMKP-based video personalization strategy is shown to compile and deliver an optimal (i.e., most relevant) (sub)set of the video contents while simultaneously satisfying multiple client-side system-level constraints. In the proposed scheme, videos are first segmented and indexed automatically in a single pass using the data-driven stochastic modeling approach described in chapter 2. The indexed video segments are summarized in a semantic manner resulting in video content summaries at multiple levels of abstraction. These video segments and their content summaries are stored in a video database which can facilitate semantic-level video retrieval. The client's request consists of the client's video content preference(s), and the client-side system-level constraints such as viewing time limit, battery power capacity, screen resolution etc. When the video server receives a request from a client, it first retrieves the relevant video contents from the video database. It then ascertains the client-side resource constraints and forwards the resulting client profile to a video personalization module. The personalization module optimally selects from the retrieved video segments, a subset of video segments or summaries at the appropriate levels of abstraction that best matches the client content preference(s) while simultaneously satisfying the various client-side resource constraints

The remainder of the chapter is organized as follows. Section 3.2 provides a brief review of related work in the fields of video summarization and personalization. Section 3.3 discusses Hierarchical content representation of video segment. The computation of the relevance values of the video segments and video summaries is detailed. In Section 3.4, various video personalization strategies based on variations of the classical Knapsack Problem (KP) are discussed and the proposed MMKP-based video personalization strategy is detailed. In Section 3.5, experimental evaluation results of the proposed MMKP-based video personalization are

compared with those of existing 0/1KP-based and FKP-based personalization strategies. Section 3.6 concludes the paper with an outline for future work.

3.2 **Related Work**

Video transcoding and summarization is an active field of research in computer vision, and constitutes the necessary step toward video personalization [Aigrain, 1996]. The overall goal of most video transcoding and summarization schemes is to reduce the amount of resources required to receive, render, play and view the video stream while preserving the desired level of detail of the original video contents.

Early work in video transcoding has typically focused on reducing the bit rate in order to meet the available channel capacity [Nakajima, 1995]. Despite increases in bandwidth availability, the fact that the underlying medium is shared necessitates media adaptation at the edges of the network. Transcoding schemes designed for bit rate reduction are usually Discrete Cosine Transform (DCT) based [Eleftheriadis, 2006],[Sun, 1996] whereas those designed for spatial resolution reduction are based on downscaling of the standard video frame size. A cascaded DCT-domain downscaling transcoder (CDDT) architecture is first proposed by Zhu et al. [Zhu, 1998], where a bilinear filtering scheme was used for spatial resolution downscaling in the DCT domain. Temporal transcoding schemes, on the other hand, are designed to reduce the number of video frames transmitted to the client [Chen, 2002]. Temporal resolution reduction techniques may be used to reduce the bit-rate requirements imposed by a network, to maintain a higher quality of coded frames, or to satisfy the viewing time limitations imposed by a client.

Based on the video transcoding techniques, video summarization is to create shortened video clips or video posters from an original video stream. The scheme of video summarization is divided into two categories. The first is to temporally compress the amount of video data to

generate a concise video summary. Some actual examples of this categories of scheme are movie trails and sports digests. Video skimming techniques using visual, auditory and textual features of video stream data are applied to summarize videos[Smith, 1997], [Lienhart, 1997], [Oh, 2000], [Babaguchi, 2000]. The second category of video summarization methods represents video contents using storyboard. It is suitable for at-a-glance presentation by laying out spatial visualization [Yeung, 1997], [Chang, 2000], [Toklu, 2000].

Various innovative key frame selection algorithms have been proposed in the literature to temporally compress video data. Doulamis et al. [Doulamis, 2000] use a content-sampling algorithm to extract a small set of key frames from a video stream. Kim et al. [Kim, 2000] take advantage of the objects of interest in the video along with their actions and the resulting events to generate a video abstraction. To facilitate content-based retrieval, video summaries are typically organized in a hierarchical manner. Jaimes et al. [Jaimes, 2000] propose a visual information indexing framework for systematic representation of image and video data based on syntax and semantics. In our client-centered video personalization system, content-aware key frame selection algorithm is used to generate video summaries. Summarized versions of the videos are labeled by semantic terms selected from a video description ontology.

Various personalization strategies have been proposed in the literature to generate the optimal response to the client's request while satisfying various client-side system-level constraints [Smyth, 2000], [Jasinschi, 2001], [Babaguchi, 2004]. The optimal response to the client's request is defined as a set of video summaries that is most relevant to the client's content preference(s). [Babaguchi, 2004] uses a domain-dependent heuristic personalization strategy for broadcasted American football videos. Merialdo et al. [Merialdo, 1999] demonstrate that the video personalization problem can be modeled as the classical 0/1 Knapsack Problem (0/1KP).

Tseng et al. [Tseng, 2003],[Tseng, 2004] propose a personalization strategy based on a combination of 0/1KP-based optimization and context clustering to collect successive similar shots. Context clustering is shown to be an enhancement of the scheme proposed in [Tseng, 2003] in that it considers the temporal smoothness of the generated video summary in order to improve the client's viewing experience. One of the drawbacks of 0/1KP-based video personalization strategies is that some of the video segments which are excluded in the response to the client's request may still contain some information that is potentially relevant or of interest to the client. Another drawback of 0/1KP-based personalization strategies is that 0/1KP-based optimization algorithms can support only a single client-side constraint, such as viewing time limit, at a time.

3.3 **Computation of Relevance Value**

3.3.1 **Hierarchical Video Content Representation**

In order to optimally satisfy the request of a resource-constrained mobile client, it is often necessary to transform the original video stream(s) into various transcoded versions based on the available resources. These transcoded versions have different requirements in terms of the various client-side resources needed to receive, transmit, render and view the transcoded video.

In the proposed system, each indexed video segment is summarized at multiple levels of abstraction using algorithms for content-aware key frame selection and motion panorama generation. We use a clustering algorithm to parse video segments into shots. The video parsing algorithm uses inter-frame histogram difference measures to identify the shot boundaries in a video segment. Since video segments are usually long, we use a temporally localized 2-class (i.e., binary) clustering algorithm to detect shots within a video segment. For a temporal window $w[t_1, t_2]$, we perform 2-class clustering to separate frames in the window into classes

c_1 and c_2. A rejection threshold *R* is set such that if the distance between c_1 and c_2 is less than *R*, then the width of the temporal window is increased to $[t_1, t_3]$ where $t_3 > t_2$ and the clustering redone.

In the key frame-based transcoding scheme, each shot is represented by a set of key frames. Frames within a shot are clustered into groups. For each group of frames, a key frame is selected. Groups with too few frames are merged with nearby groups. By selecting a threshold value for the group size, we can control the level of abstraction of the video summary. The smaller the threshold value, the more detailed the resulting summary and vice versa. The advantage of this approach is that we can preserve the dynamic nature of the content of the video frames within a shot. Not only are the commonly encountered frames within the shot selected as the key frames, but also those that deviate substantially from the commonly encountered frames in terms of their content.

Thus, in the case of key frame-based transcoding scheme, each video summary consists of a set of key frames. If the image frames are displayed at a fixed frame rate, the higher the level of abstraction, the shorter the duration of the video summary. The number of levels of abstraction associated with the transcoded video is set to three. The relative time durations of the transcoded video segments are set to 100%, 50% and 20% of the time duration of the original video segments. Due to the nature of the proposed MMKP-based personalization strategy, additional levels of abstraction and transcoding methods can be incorporated as and when necessary without any modifications to the other parts of the overall system.

In the case of video shots containing dominant panning camera motion (i.e., pan shots), motion panoramas based on image mosaicking are an efficient representation of the video shot [Bartoli, 2004]. Pan shots can be detected based on the underlying pattern of the motion vectors

(MVs) [Bhandarkar, 1999]. Let θ_{ij} be the direction of the MV associated with the ij^{th} pixel. Let $\theta_{avg} = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\theta_{ij}$ be the average of the MV directions in the frame. For a frame to qualify as a member of a pan shot, the variance $\sigma_{\theta}^{2} = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}[\theta_{ij} - \theta_{avg}]^{2}$ should be less than a predefined threshold.

The motion panorama construction algorithm consists of three major phases: static background generation, background-foreground segmentation (i.e., extraction of moving objects) and final panorama composition. During the first phase, the homographies corresponding to the motion of the camera are computed for certain frames. The static background for the entire scene underlying the video sequence is generated by stitching the individual frames into a single large wide-angle panoramic image using these homographies. In the second phase, the dynamic foreground, which includes regions corresponding to both, moving objects and false detections in the scene, is segmented by warping together three consecutive frames in the video sequence and consequently detecting the intensity discrepancy at each pixel location. Finally, the foreground objects or regions are pasted back onto the static background using the location information from the homographies computed in the first phase and the position coordinates computed during the second phase. If the motion panorama is encoded to have the same frame size as the original video shot, then there is a significant saving in terms of the amount of data to be transmitted and the required bandwidth or bitrate. Note that in the case of motion panorama-based transcoding, the frame(s) corresponding to the stationary background need(s) to be transmitted only once or very infrequently. Only the frames corresponding to the dynamic foreground need to be transmitted in the form of a motion overlay at the required frame rate [Bartoli, 2004]. Since the

dynamic foreground regions are relatively few in number, the bandwidth requirement of the motion overlay is much lower than that of the original video shot [Bartoli, 2004].

Video segments are labeled using semantic concepts selected from a video description ontology which, in our current implementation, is a three-level hierarchy of semantic terms, depicted in Figure 3.1. The first level in the ontology represents the video category (TV Broadcast, Sports, Surveillance, etc.), the second level represents the video program group (Broadcast News, Soccer, Basketball, Traffic Surveillance, etc.) and the third level defines the various semantic concepts (*Anchor*, *Weather Forecast*, *Commercial*, *Replay*, *Closeup*, etc.). The hierarchical nature of the video description ontology provides a structural framework for representation and storage of the video segments and their transcoded versions. The video description ontology enables the generation of a personalized response to a client's request; one that best matches the client's preference(s) with regard to the video content while simultaneously ensuring that the various client-side resource constraints are satisfied.

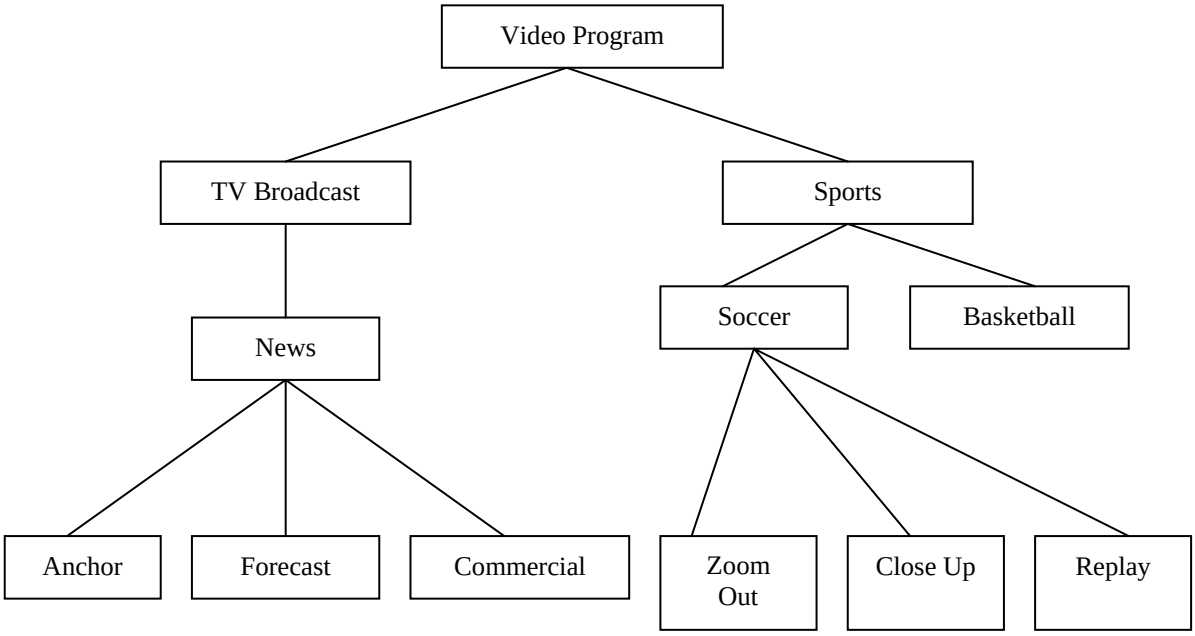


Figure 3.1 Three-level Hierarchy of Semantic Concepts

3.3.2 **Computation of Relevance Value**

In order to measure and evaluate the performance of various video personalization strategies, the relationship between the information content of the original video versus the amount of information retained in its various transcoded versions needs to be established. This is a complex task due to the inherent difficulty in quantifying the amount of information contained within the original video and due to the diverse nature of the various transcoded versions of the original video. In most cases, the amount of information contained within a video (including its transcoded versions) does not necessarily increase linearly with its duration. Although Shannon entropy has been used as a measure of pixel-level or feature-level information content of a video [Snoek, 2003], the relationship between the low-level feature-based entropy measure of a video stream and its high-level semantic content has not been firmly established.

In Section 3.3.2.1, we discuss how to compute the relevance value of a video segment based on the client's content preference(s). In Section 3.3.2.2 we discuss computation of the relevance value of a video summary of a video segment based on its relative duration and the relevance value of the original video segment.

3.3.2.1 **Relevance Value of a Video Segment**

Video segments are indexed using semantic terms. Each video segment is assigned a relevance value based on the client's preference with regard to video content. Assume video segment S_i is indexed by a semantic term T_i. In its request, the client specifies a preference for video content using a descriptive term labeled as *P*. The relevance value V_i assigned to the video segment S_i is then given by:

$$V_i = similarity(T_i, P), 0 \le V_i \le 1 \qquad (3.0)$$

In the current implementation the *similarity* is evaluated using the *lch* semantic similarity measurement algorithm [Leacock, 1998]. The *lch* algorithm measures the length of the shortest path between two concepts in the WordNet lexical database and scales the value by the maximum *is-a* path length.

3.3.2.2 **Relevance Value of a Video Summary**

We now discuss how to compute the relevance value of a video summary based on its relative duration and the relevance value of the original video segment computed using equation (3.0). Although there are many factors that determine the information content of a video, it is reasonable to assume that the amount of information or detail contained within a video summary is related to its duration. For each video segment, its original version is assumed to contain the greatest amount of detail; whereas its summary at the highest level of abstraction is assumed to contain the least amount of detail. Typically, the amount of information contained within a video summary (relative to original version) does not necessarily increase linearly with its relative duration. This is especially true when each indexed video segment is summarized at multiple levels of abstraction using algorithms for content-based key frame selection and motion panorama computation.

Each indexed video segment is summarized at multiple levels of abstraction using content-aware key frame selection and motion panorama computation algorithms. Each video summary consists of a set of key frames and motion panoramas. If the image frames are displayed at a fixed frame rate, the higher the level of abstraction, the shorter the duration of the video summary. This is so because at a higher level of abstraction, fewer image frames are included in the video summary. Since the FKP-based and the MMKP-based video personalization strategies could potentially include both the original video segments and their summaries, the relationship

between the relevance value of the original video segment and that of its summaries needs to be first established.

For each video segment, its original version is assumed to contain the greatest amount of detail; whereas its summary at the highest level of abstraction is assumed to contain the least amount of detail. It is reasonable to assume that the amount of information contained within a video summary (relative to original version) is related to its duration, i.e.

$$v_i = v_{i0} \cdot f(L_i / L_0) \qquad (3.1)$$

where v_{i0} is the relevance value of the original video segment, and L_0 and L_i are the time durations of the original video segment and the summarized (or transcoded) video segment respectively. The function $f(L_i / L_0)$ represents the relationship between the amount of information contained within a video summary relative to the original video segment.

We propose to use empirical laws to quantify the relationship between the amount of information contained in the transcoded videos relative to the original video. The *Zipf* function, *sigmoid* function and *Rayleigh* distribution are proposed as plausible mapping functions for quantifying the relationship between the amount of information in the transcoded video relative to the original video, and are shown to be suitable for different kinds of videos. The performance results of the personalization subsystem are shown to vary significantly when different empirical mapping functions are used to measure the amount of information contained in the transcoded video relative to the original video.

3.3.2.2.1 **The Zipf's Law-Based Mapping Function**

The first empirical mapping function discussed in this paper is based on *Zipf's law* [Wheeler, 2002]. For some categories of videos, such as broadcast news, most of the information is revealed in a video segment spanning the first 20%-30% of the video stream. For example,

consider a broadcast news video wherein a news anchor summarizes the news events at the beginning of the video clip followed by a detailed field news video. This observation justifies the use of the *Zipf* function to quantify the relationship between the amount of information contained in the transcoded (or summarized) videos relative to the original video. The mathematical definition of the *Zipf* function [Wheeler, 2002] is given by:

$$I = H_{k,s} / H_{N,s} \qquad (3.2)$$

where I (expressed as a percentage) is the amount of information contained within a video summary relative to the original video segment, N is the set of all possible discrete durations of the video summary, $k \in N$ is the duration of a video summary, $s > 0, s \in R$ is the characteristic parameter of the Zipf function and $H_{k,s}$ is the k^{th} generalized harmonic number. When $s = 0$, the information content of a video summary increases linearly (i.e., at a constant rate) with its duration.

Equation (3.2) is a definition of the discrete *Zipf* function. In our application, the relative (i.e., normalized) duration of a video summary is a continuous variable in the range [0, 1]. To use the *Zipf* function defined in equation (3.2), the following approximation and linear transform are used. Let L_{norm} denote the normalized and discrete video duration where $L_{norm} \in \{0.01, 0.02, ..., 0.99, 1.00\}$ and let $N = 100$. Then the following linear transform maps the values of L_{norm} to *k*, i.e.,

$$k = round(L_{norm} \times N) \qquad (3.3)$$

Figure 3.2 shows a plot of the relative information content of a transcoded video versus its normalized duration. The relative duration and relative information content of the transcoded video are normalized to lie within the range [0, 1] based on the duration and information content

of the original video respectively. The parameter s is set to values 0, 0.5, 1.0 and 1.5 respectively where s = 0 denotes a special case when the *Zipf* function degenerates to a linear mapping. The derivative of the *Zipf* function in equation (3.2) can be considered as the incremental information ΔI introduced by a video segment whose duration is incremented by ΔL and is given by:

$$I' = \lim_{\Delta L \to 0} \frac{I(L + \Delta L) - I(L)}{\Delta L} = \frac{1/k^s}{H_{N,s}} \qquad (3.4)$$

Figure 3.3 plots I' as a function of relative duration when the linear transform in equation (3.3) is performed. Figure 3.3 can be seen to be a depiction of the commonly observed law of diminishing marginal (incremental) return which further justifies our assumption that the Zipf's law-based mapping function is suitable for categories of videos wherein most of information is revealed in the first 20%-30% of the video segment.

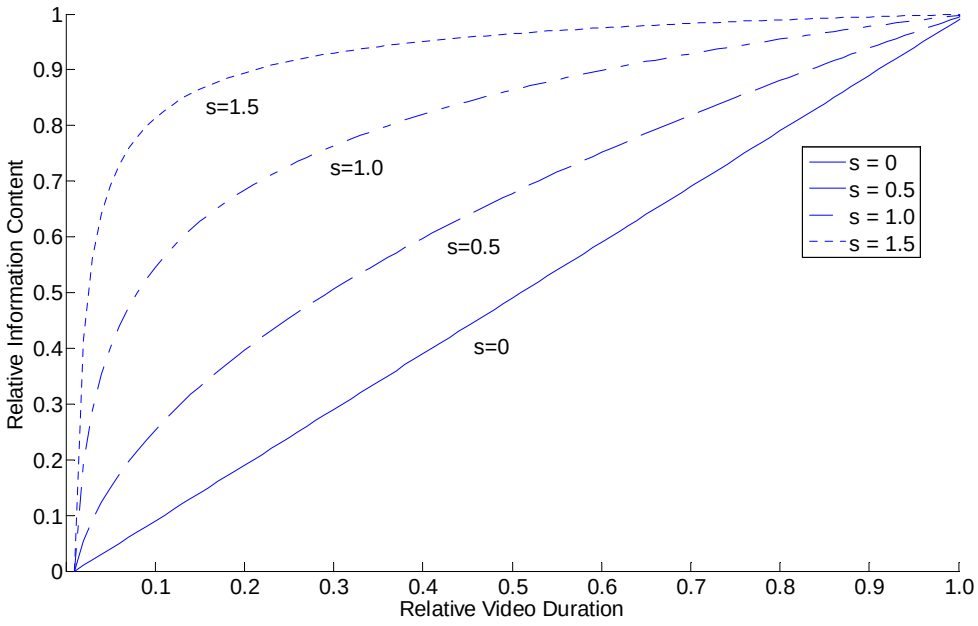


Figure 3.2 Relative Information Content of a Transcoded Video Segment versus Normalized Video Segment Duration: Zipf Function

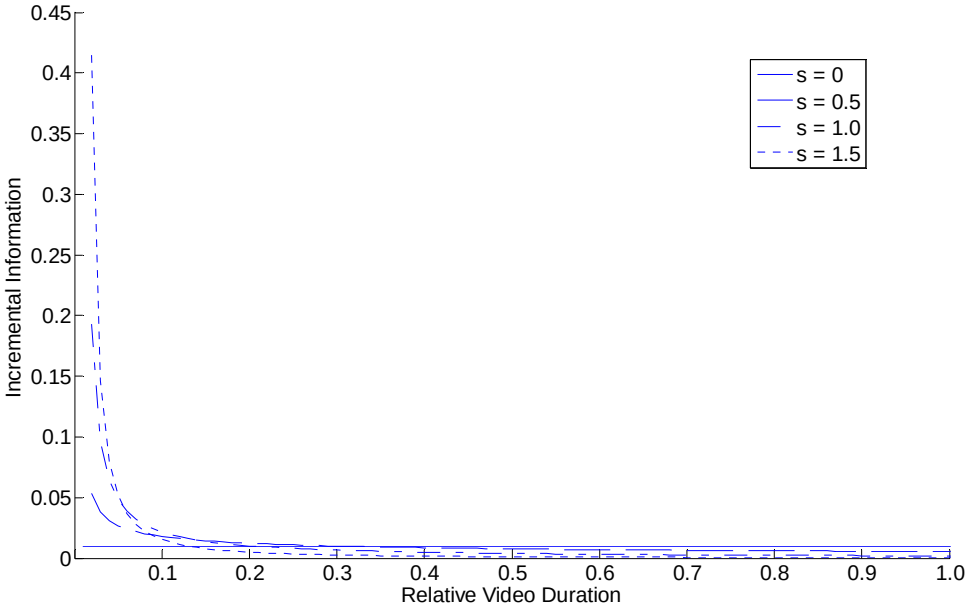


Figure 3.3 Incremental Information Content versus Normalized Video Segment Duration: Zipf Function

3.3.2.2.2 **The Sigmoid Mapping Function**

The family of *sigmoid* functions [Uykan, 2000] is another category of functions we propose to use in order to quantify the information content in a transcoded video segment relative to the original video segment. This family of functions is suitable for categories of videos wherein the middle 20%-30% of the video segment accounts for most of the information content.

Equations (3.5) and (3.6) describe the sigmoid function family and their corresponding derivatives respectively, where $\alpha > 0$ is the characteristic shape parameter of the sigmoid function and the parameter L is dependent on the duration of the video segment. Analogous to the parameter s in the Zipf's law, the parameter σ regulates the shape of the sigmoid function:

$$I = \frac{1}{1 + e^{-\alpha L}} \qquad (3.5)$$

$$I' = \frac{1}{1+e^{-\alpha L}}\left(1 - \frac{1}{1+e^{-\alpha L}}\right) \qquad (3.6)$$

When $L \notin [-6,6]$, the function values in equations (3.5) and (3.6) are observed to be less than 5% of their corresponding maximum values. Thus, in the context of our application, it suffices in terms of precision, to truncate the domains of the functions I and I' such that $L \in [-6,6]$. The duration of the video segment is normalized, such that the normalized duration $L_{norm} \in [0,1]$. In order to map the normalized duration $L_{norm} \in [0,1]$ to the parameter $L \in [-6,6]$ in equations (3.5) and (3.6), the following function is used:

$$L = 12 \times L_{norm} - 6 \qquad (3.7)$$

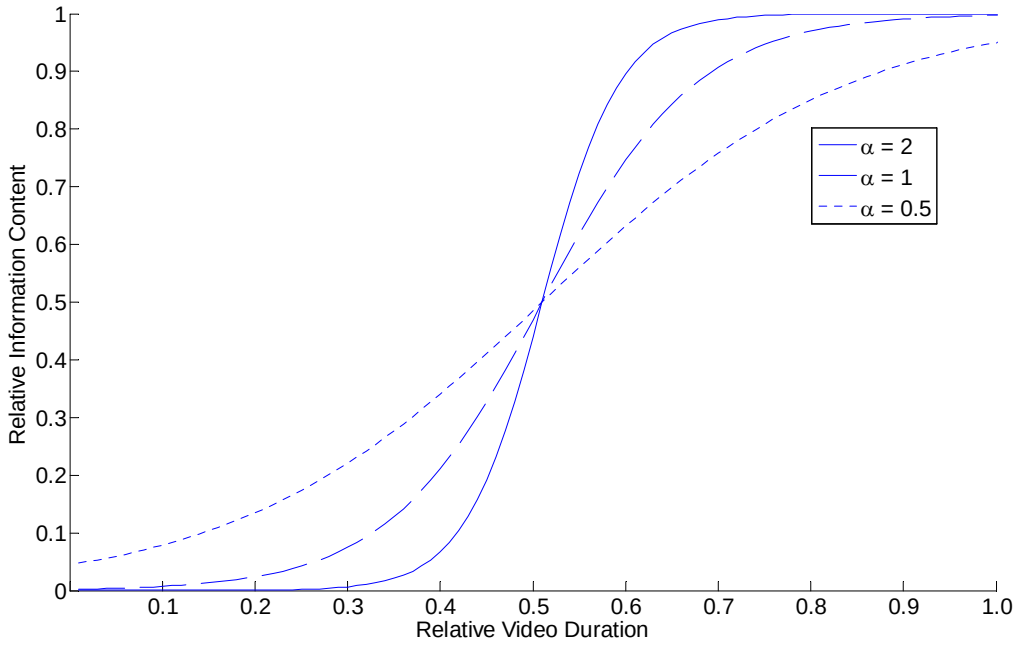


Figure 3.4 Relative Information Content of a Transcoded Video Segment versus Normalized Video Segment Duration: Sigmoid Function

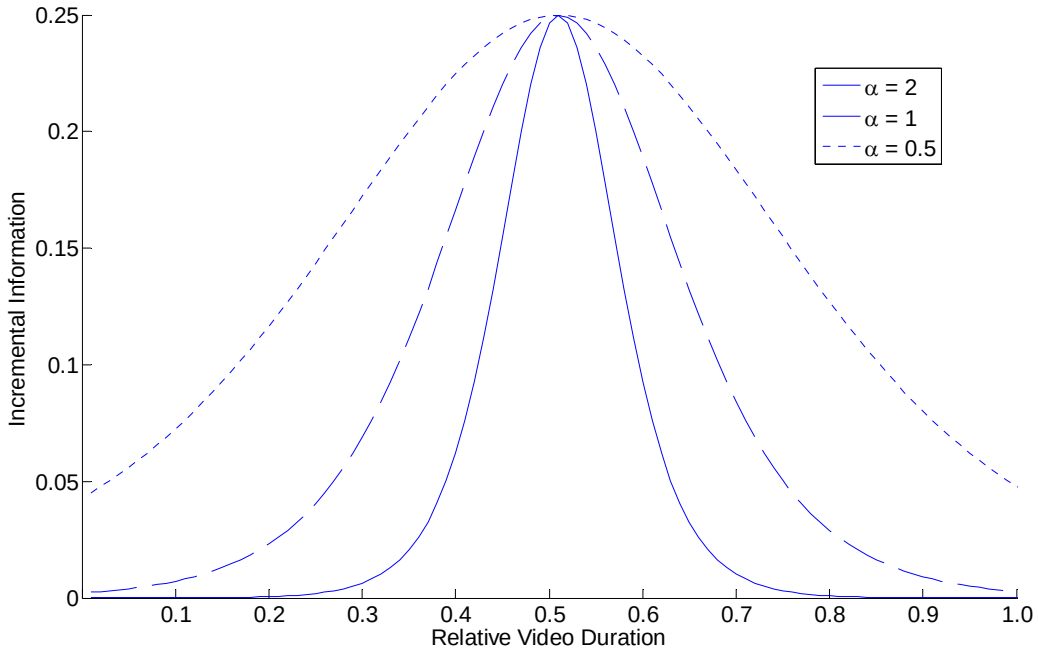


Figure 3.5 Incremental Information Content versus Normalized Video Segment Duration: Sigmoid Function

Figures 3.4 and 3.5 show the plots of the sigmoid function for different values of α (equation (3.5)) and the corresponding derivatives (equation (3.6)) respectively. In Figures 3.4 and 3.5, the *x* axis denotes normalized duration L_{norm} from which the corresponding value of L in equations (3.5) and (3.6) respectively is computed using equation (3.7).

3.3.2.2.3 **The Cumulative Rayleigh Distribution-based Function**

The Rayleigh distribution [Papoulis, 1984] is another plausible mapping function that can serve to quantify the relationship between the information content of the original video and that of its transcoded versions. Unlike the *sigmoid* family of functions, the incremental information governed by the *Rayleigh* distribution is skewed to the left. This characteristic makes the *Rayleigh* distribution suitable for videos in which most of the information is revealed in the earlier portions of the video segment, but not necessarily at the beginning. Equations (3.8) and

(3.9) give the definitions of the cumulative *Rayleigh* distribution and its derivative (i.e., the standard *Rayleigh* distribution) respectively where L is video segment duration and $\sigma > 0$ is the characteristic parameter of the *Rayleigh* distribution.

$$I = \frac{L \cdot e^{(-L^2/2\sigma^2)}}{\sigma^2} \qquad (3.8)$$

$$I' = 1 - e^{(-L^2/2\sigma^2)} \qquad (3.9)$$

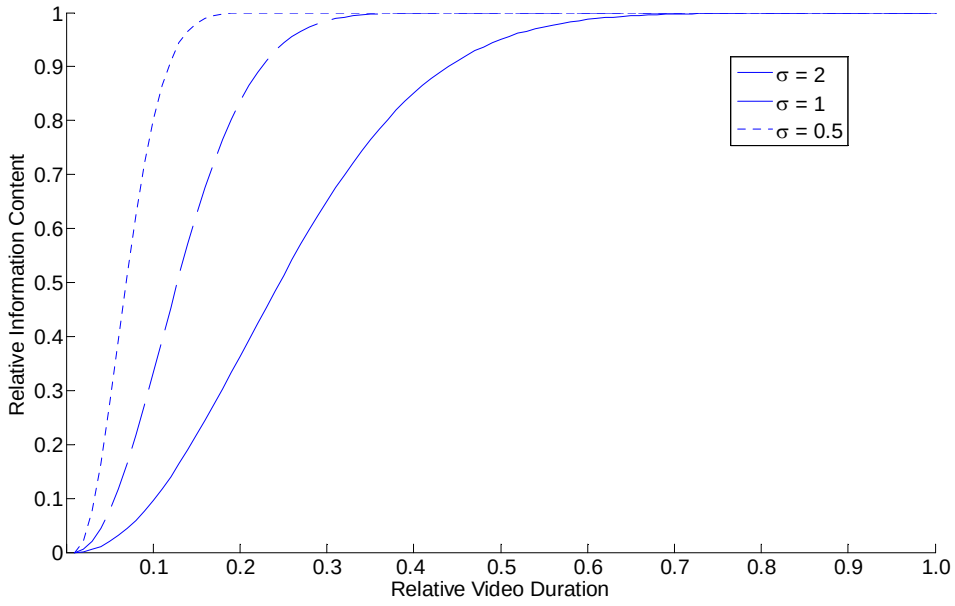


Figure 3.6 Relative Information Content of a Transcoded Video Segment versus Normalized Video Segment Duration: Cumulative Rayleigh Distribution

Figure 3.6 and Figure 3.7 plot the function curves of I (equation (3.8)) and I' (equation (3.9)) respectively where the video segment duration is normalized. The values of the functions I (equation (3.8)) and I' (equation (3.9)) are observed to be less than 5% of their corresponding maximum values for $L \notin [0,10]$. Thus in the context of our application, it suffices, in terms of

precision, to truncate the domains of functions I and I' such that $L \notin [0,10]$. The normalized video segment duration $L_{norm} \in [0,1]$ is mapped to the parameter $L \in [0,10]$ in order to compute the values of the cumulative Rayleigh distribution (I) and the Rayleigh distribution (I') using the following linear transform:

$$L = L_{norm} \times 10 \qquad (3.10)$$

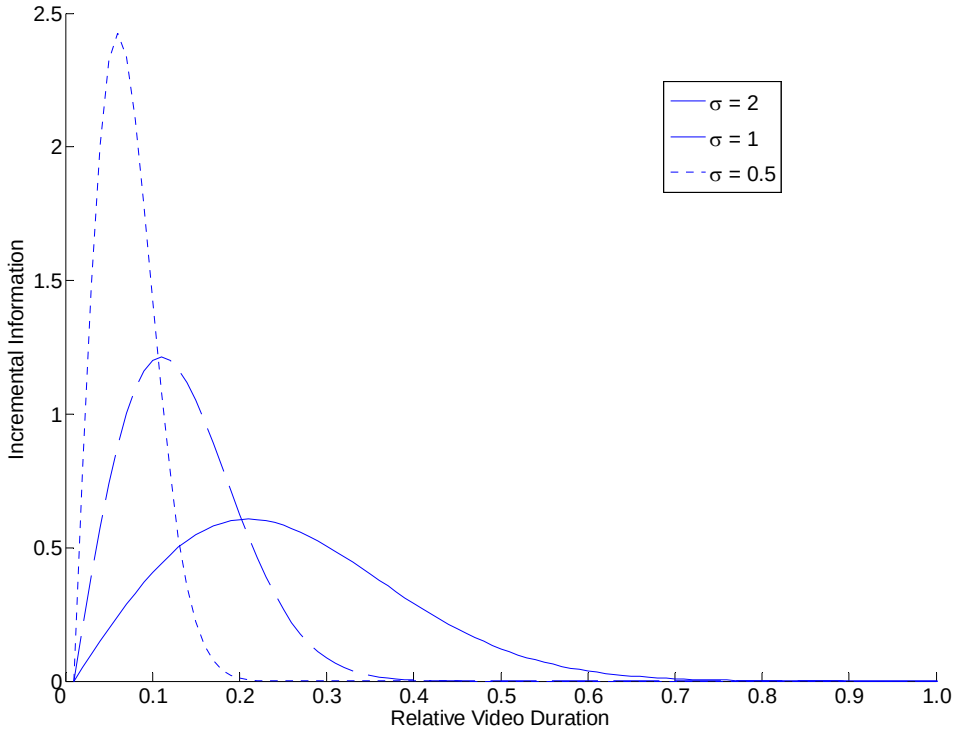


Figure 3.7 Incremental Information Content versus Normalized Video Segment Duration: Rayleigh Distribution

3.4 **Video Personalization Strategies**

The objective of video personalization is to present a customized or personalized video summary that retains as much of the semantic content desired by the client as possible but within the

resource constraints imposed by the client. The client typically wants to retrieve and view only the video content that matches his/her content preference(s). In order to generate the personalized video summary, the client preference(s), the client usage environment and client-side resource constraints need to be considered. The personalization engine compiles the optimal video content (i.e., the most relevant set of video summaries) for the client that satisfies his/her resource constraints. With a view towards optimizing the delivered information content while satisfying multiple client-side resource constraints, this paper presents the design and implementation of an MMKP-based video personalization strategy to generate a customized response to a client's request.

Compared to the 0/1KP-based and the FKP-based video personalization strategies presented in [Merialdo, 1999], [Tseng, 2003] and [Tseng, 2004], the proposed MMKP-based video personalization strategy is shown to include more relevant information in its response to the client's request. The MMKP-based personalization strategy is also shown to satisfy multiple client-side resource constraints, in contrast to the 0/1KP-based and the FKP-based personalization strategies which can only satisfy a single client-side resource constraint at a time.

3.4.1 **Video Personalization as Constrained Optimization**

The input videos are first segmented and indexed using semantic terms selected from a video description ontology as described earlier. Each video segment is assigned a relevance value based on the client's preference with regard to the video content. Let the set $S = \{S_1, S_2, ...S_n\}$ denote the video segments that are stored in the video database, where S_i denotes the i^{th} video segment and *n* is the total number of candidate video segments to be included in the response to the client's request. Video segment S_i is indexed by a semantic term T_i selected from the video description ontology. In its request for video content, the client specifies a preference for video

content using a semantic term *P*. A relevance value V_i is assigned to the video segment and is given by $V_i = similarity(T_i, P), 0 \le V_i \le 1$. In the current implementation the *similarity* function is computed using the *lch* similarity measure algorithm [Leacock, 1998]. The *lch* similarity measure algorithm measures the length of the shortest path between two semantic concepts, and scales the value by the maximum *is-a* path length in the WordNet lexical database [Fellbaum, 1998]

In the following sections, we describe the 0/1KP-based, the FKP-based and the MMKP-based video personalization strategies and compare their relative performance.

3.4.2 **Video Personalization Modeled as the 0/1KP**

Merialdo et al. [1999] propose that video personalization be formalized as a constrained optimization problem that is modeled as the classical 0/1 Knapsack Problem (0/1KP) given by:

$$\max_{i \in \{1,2,\ldots,n\}} (\sum_i V_i),$$

subject to

$$\sum_i L_i \le T \qquad (3.11)$$

where L_i is the duration of video segment *i* and *T* is the client video viewing time limit. To solve the 0/1KP, a dynamic programming algorithm can be used. The dynamic programming algorithm for the 0/1KP has a time complexity of $O(nT)$. Video segments included in the server's response to the client's request are of the original (i.e., non-transcoded) quality. However, some of the video segments which are excluded in the server's response may still contain some information of potential interest or relevance to the client. The 0/1KP-based video personalization algorithm does not include this information.

3.4.3 **Video Personalization Modeled as the FKP**

A fractional portion of a video segment could be included in the set of video segments compiled by the personalization module. The video segment is transcoded to enable it to fit within the limits of the available viewing time. In this case, video personalization is formulated as a constrained optimization problem that is modeled on the following fractional knapsack problem (FKP).

$$\max_{i\in\{1,2,\ldots,n\}}(\sum_{i} x_i V_i),$$

subject to

$$\sum_{i} y_i L_i \le T \qquad (3.12)$$

where T is the client video viewing time limitation, L_i is the temporal length of video segment S_i, and $x_i, y_i \in [0,1]$. The above FKP can be solved by using a greedy algorithm. Video segments are sorted in decreasing order of their *Value_Intensity* as computed in equation (5.3), where V_i is the relevance value and L_i is the time duration of video segment S_i.

$$VauleIntensity = V_i / L_i \qquad (3.13)$$

Video segments with high *Value_Intensity* values are selected first. A fractional portion of a video segment may be included in the set of video segments compiled by the personalization module. Although the FKP-based optimization scheme can include transcoded video segments, some potentially relevant videos could excluded in the server's response. This can be attributed to the basic nature of the constrained optimization problem posed by the FKP and the greedy algorithm used to solve it. The complexity of FKP is $O(n)$.

3.4.4 **Video Personalization Modeled as the MMKP**

Multimedia content can be represented at different levels of abstraction. The various levels of abstraction describe semantic video content in a hierarchical fashion. In the context of video personalization, the original video segment is considered to be associated with a discrete set consisting of its various transcoded versions. Each transcoded version is deemed to represent the semantic information content of the original video segment at a certain predefined level of abstraction. Furthermore, client-side resource constraints are typically multi-dimensional, i.e., in addition to constraints on the client's viewing time, there typically are constraints on other client-side resources, such as available battery energy, client bandwidth and video display quality. Thus, the amount of relevant information included in the personalized video in response to a client's request needs to be maximized subject to multiple resource constraints. This version of the video personalization problem is modeled along the Multiple-choice Multi-dimensional Knapsack Problem (MMKP) [Khan, 1998] [Akbar, 2001], [Hernandez, 2005] and is formulated as follows:

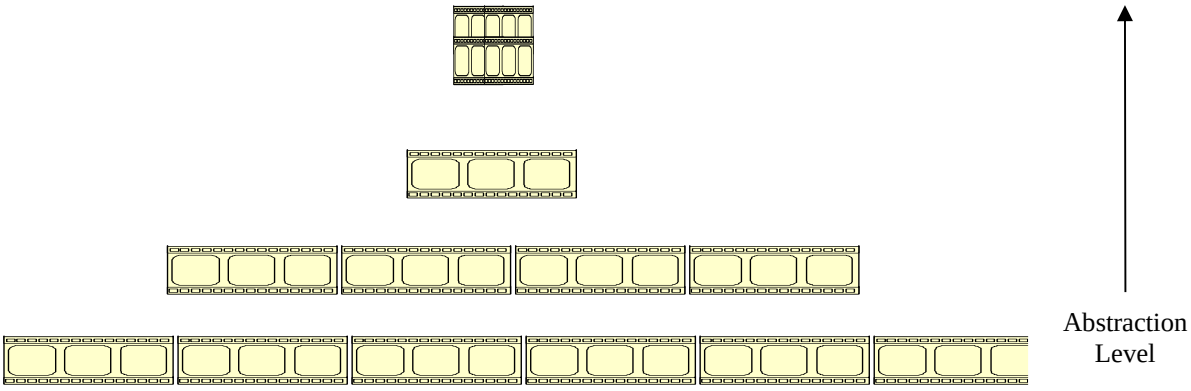


Figure 3.8 Representation of a Content Group at Multiple Levels of Abstraction

Each video segment S_i is transcoded into l_i versions, denoted as S_{ij}, $j \in \{1,2,...,l_i\}$. The original video segment and its transcoded versions constitute a *content group* at multiple levels of abstraction, as shown in Figure 3.8. Each *item* within a *content group* is a video segment. Each transcoded version is associated with a relevance value and is deemed to require *m* resources. The objective of the MMKP-based video personalization strategy is to select exactly one *item* from each *content group* in order to maximize total relevance value of the selected segments, subject to *m* resource constraints determined by the client. Let v_{ij} be the relevance value of the j^{th} version of the video segment S_i, $\vec{r}_{ij} = (r_{ij1}, r_{ij2},..., r_{ijm})$ be the required resource vector for the j^{th} version of the video segment S_i and $\vec{R} = (R_1, R_2,..., R_m)$ be the resource bound of the knapsack representing the *m* resources. The problem therefore is to determine

$$V = \max(\sum_{i=1}^{n}\sum_{j=1}^{l_i} x_{ij} v_{ij})$$

subject to

$$\sum_{i=1}^{n}\sum_{j=1}^{l_i} x_{ij} r_{ijk} \le R_k, k = 1,2,..., m$$

and

$$\sum_{j=1}^{l_i} x_{ij} = 1, x_{ij} \in \{0,1\} \qquad (3.14)$$

Based on the above formulation of the MMKP-based video personalization strategy it is obvious that more *items* can be added to a *content group* without requiring any other changes to the video personalization system. The modularity and extensibility of MMKP-based video personalization strategy is one of its salient features.

The MMKP is known to be an NP-hard problem [Hernandez, 2005]. The exact solution to the MMKP can be obtained using a brand-and-bound integer programming (BBIP) algorithm [Vanderbei, 1997]. In order to use the BBIP algorithm for solving the MMKP, we follow the approach of Hernandez et al. [2005] and cast the MMKP as a multi-dimensional knapsack problem (MKP). Let us define $q = \sum_{i=1}^{n} l_i$, $L_h = \sum_{i=1}^{h} l_i$, and $L_0 = 0$. Let us also define $z = L_{i-1} + j$. Hence $x_{ij} = x_z^{'}$ and $r_{kij} = r_{kz}^{'}$. We define the coefficients for the equality constraints as follows. In the h^{th} equality constraint, if the new variable $x_z^{'}$ is observed to belong to the h^{th} *content group*, then the coefficient $a_{hz}^{'} = 1$; otherwise $a_{hz}^{'} = 0$. In this case the MMKP can be rephrased as:

Determine

$$V = \max(\sum_{z=1}^{q} v_z^{'} x_z^{'})$$

subject to

$$\sum_{z=1}^{q} r_{kz}^{'} x_z^{'} \le b_k^{'}, k = 1,2,..., m$$

and

$$\sum_{z=1+L_{h-1}}^{L_h} a_h^{'} x_z^{'} = 1, h = 1,2,..., m, x_z^{'} \in \{0,1\} \qquad (3.15)$$

For the sake of simplicity and in order to follow standard notation, we will denote variables and coefficients by dropping the accent sign. The MMKP is then expressed as:

Determine

$$V = \max(\sum_{z=1}^{q} v_z x_z)$$

subject to

$$\sum_{z=1}^{q} r_{kz} x_z \le b_k, k = 1,2,..., m$$

and

$$\sum_{z=1+L_{h-1}}^{L_h} a_h x_z = 1, h = 1,2,..., m, x_z \in \{0,1\} \qquad (3.16)$$

In order to use the BBIP algorithm in Matlab, we reformulate the MMKP using vector and matrix notations as follows:

Determine

$$V = \max(v^T \cdot x)$$

subject to

$$R \cdot x \le b \text{ and } A \cdot x = 1 \qquad (3.17)$$

Both v and x are vectors of size $q \times 1$, R is a $m \times q$ array such that $R_{kz} = r_{kz}$ and A is a $n \times q$ array such that $A_{hz} = a_{hz}$. The MMKP as formulated above can be solved using the BBIP algorithm in Matlab [Vanderbei, 1997]. Since the MMKP known to be an NP-hard problem, the worst-case time complexity of the BBIP algorithm used to solve the MMKP is exponential in *n*, *m*, and l_i where *n* is the number of content groups, *m* is the number of resource constraints and l_i is the number of items in the *i*-th content group of the MMKP problem [Hernandez, 2005], [Khan, 1998].

3.4.5 **Performance Metric**

In order to measure and compare the performance of the various video personalization strategies, the sum of relevance values of all video segments included in the response is used as the performance metric, i.e. $\sum_{i \in response} v_i$ For a transcoded video segment, its relevance value v_i is defined in equation (3.1).

3.5 **Experiment Results**

3.5.1 **Video Database**

The CNN Headline News video was first segmented and indexed using the stochastic multi-level HMM-based algorithm. Each video segment was labeled with terms selected from a predefined video content description ontology. Video segments were transcoded at multiple levels of abstraction and then stored in a hierarchical video database. In order to use the proposed personalization evaluation metric, key frame-based transcoding was performed such that the original video and its transcoded versions have the same spatial resolution, although their time durations are different. Based on the client's request and client-side resource constraints, the video personalization system was designed to select an optimal set of video segments from the database in response to the client's request. The sum of relevance values of the video segments included in the server's response was used to measure and compare the performance of the various video personalization strategies.

3.5.2 **Relevance Values of Video Items**

In order to use the proposed personalization evaluation metric defined in equation (3.19), it is necessary to assign relevance values to *video items*. For the original video segment, the relevance value is computed using equation (3.0). The *lch* semantic similarity algorithm [Leacock, 1998] is used to compute the semantic similarity between the video segment's label and the client's content preference. As discussed in Section 3.3, for video summaries we use the empirical mapping functions defined in equations (3.2), (3.5) and (3.8) to compute their relevance values. The sum of relevance values of the *video items* included in the response to the client's request is used to quantify and compare the performance of the various video personalization strategies.

3.5.3 **Experimental Results**

For the purpose of experimentally comparing the performance of the various aforementioned video personalization strategies, we set the client's content preference to a fixed semantic concept, i.e., *News*. The video personalization module is designed to generate a response consisting of a set of *video items* using the various aforementioned personalization strategies. The Zipf's law-based, the sigmoid function-based and the cumulative Rayleigh distribution-based mapping functions are used to compute the relevance values of the *video items* included in the response to the client's request. We choose the semantic concept *News* primarily, because our video database contains a large number of fairly diverse *News* video segments. This ensures that our experiments do not exhibit a bias for or against any of the proposed empirical mapping functions.

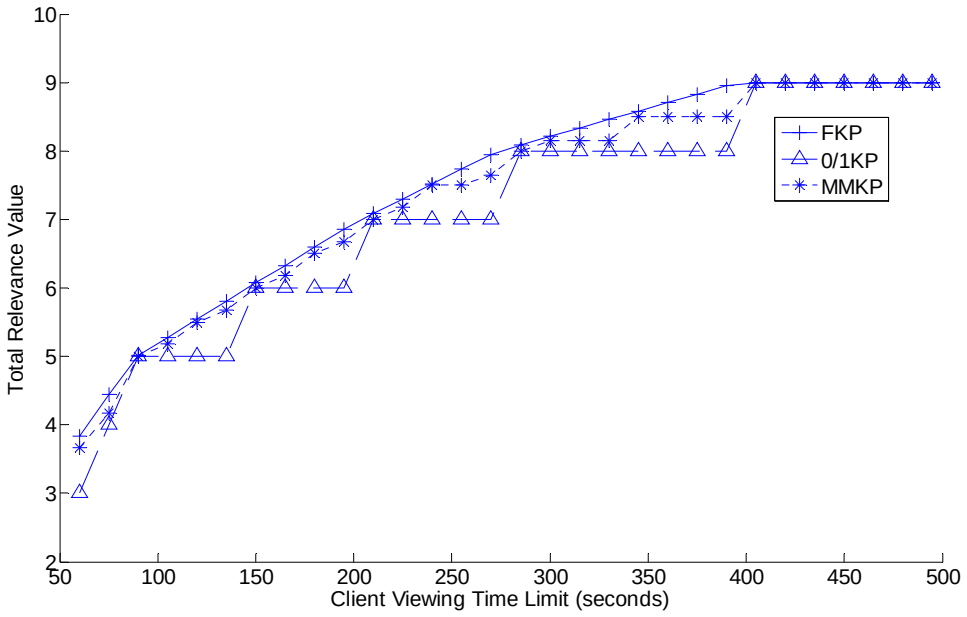
In the client's request to the server, the client specifies its content preference, i.e., the semantic concept *News* in our experiments. The client also specifies its viewing time limit as a client-side system-level constraint. It should be noted that although the MMKP-based video personalization strategy can support multiple client-side system-level constraints, the 0/1KP-based and the FKP-based video personalization strategies can only handle a single client-side system-level constraint at a time. In order to experimentally compare the performance of the 0/1KP-based and the FKP-based video personalization strategies with that of the MMKP-based video personalization strategy, the client viewing time limit is the single client-side system-level constraint used for all three video personalization strategies in all of our experiments.

In Figure 3.9(a), the Zipf function and linear transform defined in equations (3.2) and (3.3) respectively are used to compute the relevance values of the video summaries based on their time durations and the relevance values of the original video segments. The total relevance value of

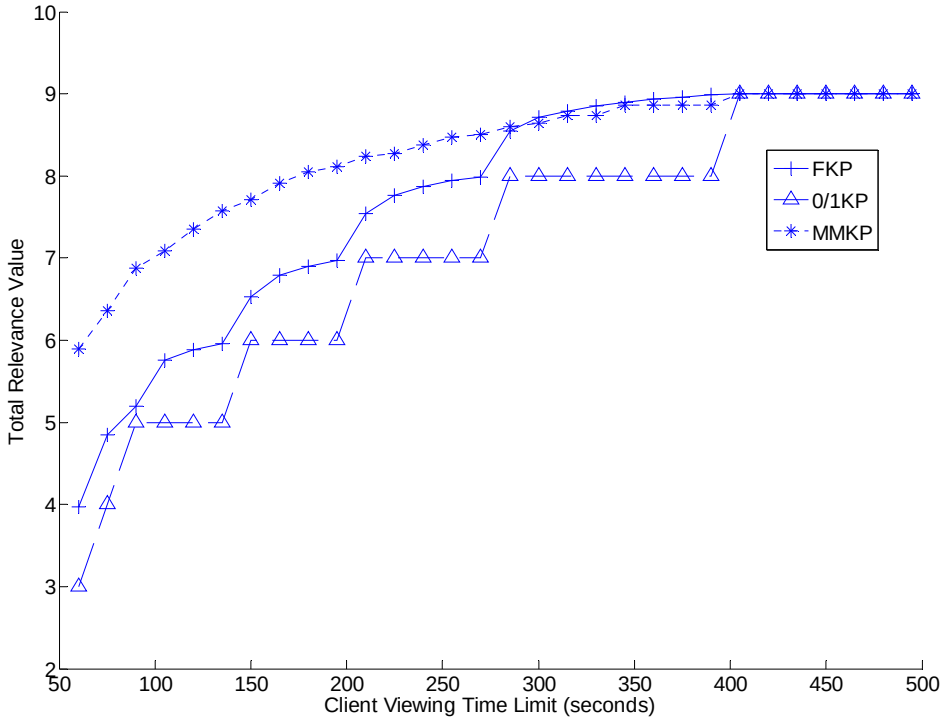
the response to the client's request is plotted against the client's video viewing time limit. In Figure 3.9(a), the characteristic parameter *s* of the Zipf function is set to zero. As defined in equation (3.15), the MMKP-based video personalization strategy selects one *item* from each *content group* where an *item* is defined as one of the video segments or video summaries in a *content group*. Hence more *video items* are included in the response generated by the MMKP-based video personalization strategy compared to the 0/1KP-based and FKP-based video personalization strategies. Note that in the response generated by the MMKP-based video personalization strategy, $size(VI_{selected}) = numGroups$. This implies that when the number of candidate *content groups* is large, the time durations of the video segments and/or video summaries included in the response are short, since more *video items* are included by the MMKP-based strategy within a prespecified viewing time limit. It is observed in Figure 3.1, that when $s = 0$, the relative information content of a video summary increases linearly with its time duration under the Zipf mapping function. In this case, the total information content of these short video segments or video summaries in the response to the client's request generated by the MMKP-based video personalization strategy is generally less than that of those included in the response generated by the FKP-based video personalization strategy. However, if we assume that beginning portion of a video segment contains the major portion of its information content, for example when $s = 1.0$, then the short video segments selected by MMKP-based personalization strategy contain more relevant information than those contained in the responses generated by the FKP-based and 0/1KP-based personalization strategies, as shown in Figure 3.9(b).

The proposed system also provides a test bed for various video personalization strategies. In Figures 3.9(c) and 3.9(d), we use the sigmoid function-based and the cumulative Rayleigh

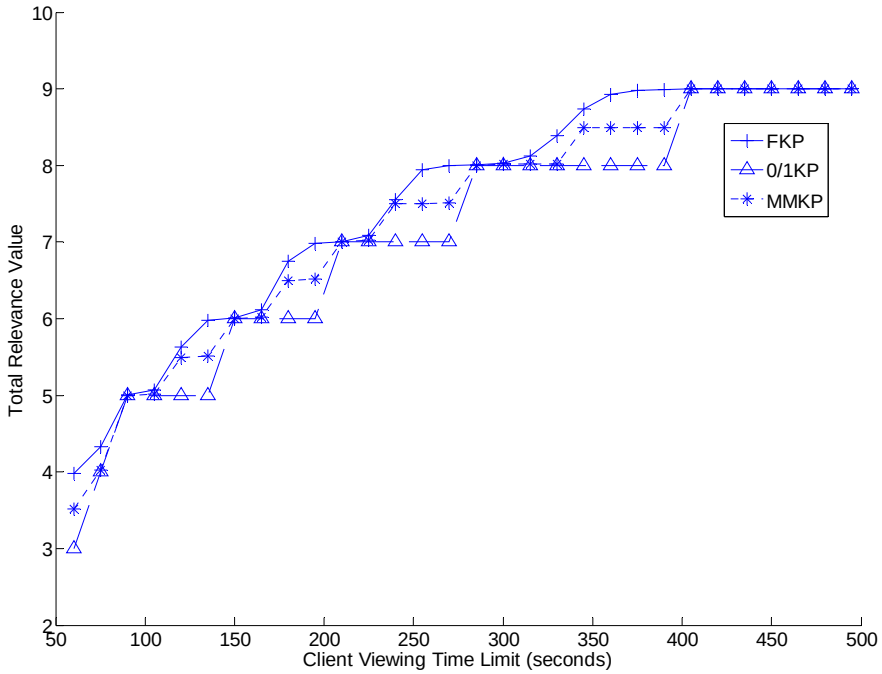
distribution-based mapping functions defined in equations (3.5) and (3.8) respectively to quantify the relative information content of transcoded (summarized) videos with respect to their original versions. Figure 3.9(c) depicts the performance of the various video personalization strategies when the sigmoid function-based mapping function is used to model the relative information content of the transcoded (summarized) video segments. The parameter α of the sigmoid function is set to 1. As shown in Figure 3.4, in the case of the sigmoid function, the central portion of a video segment is considered to contribute more information than the remaining portions of a video segment. The short video segments or video summaries selected by the MMKP-based personalization strategy contain less total information than the longer video segments or video summaries included in the responses generated by the FKP-based personalization strategy. When the cumulative Rayleigh distribution-based mapping function is used to model the information content in video summaries, as illustrated by Figure 3.6, more information is contained in the beginning portions of a video. Thus, more information is contained in the shorter video segments or video summaries in this case. In Figure 3.9(d), since more video segments or video summaries are included in the response generated by the MMKP-based personalization strategy than the responses generated by the FKP-based and 0/1KP-based personalization strategies, the MMKP-based personalization strategy yields a response with the highest total relevance value. In summary, the MMKP-based video personalization strategy is observed to generate a response with a higher overall relevance value when the Zipf law-based and Rayleigh-based mapping functions are used to model the information content of transcoded (summarized) video segments relative to the original video segments.



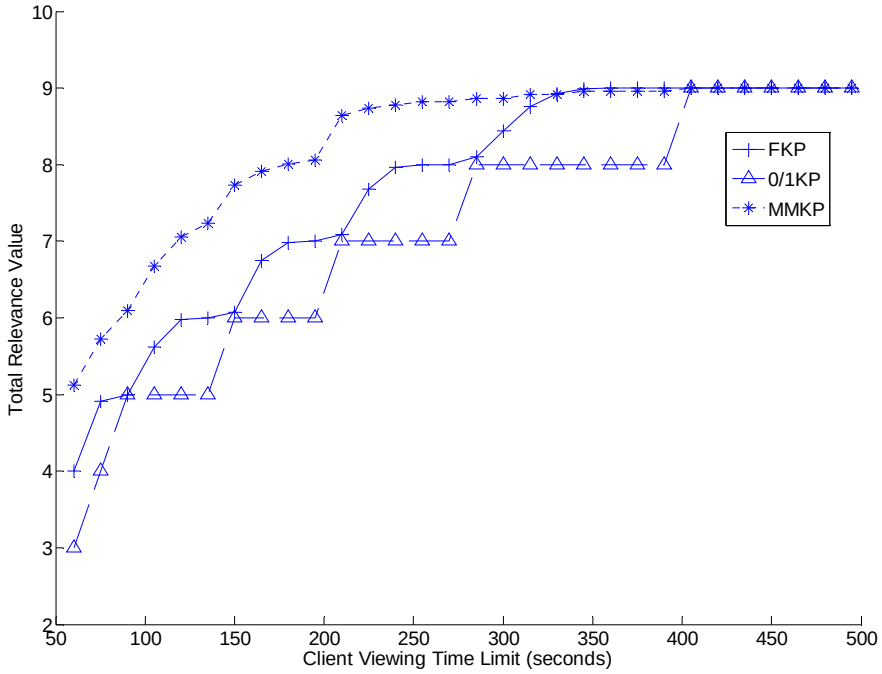
(a) **Using the Zipf Function,** $s = 0$



(b) **Using the Zipf Function,** $s = 1.0$



(c) **Using the Sigmoid Function,** $\alpha = 1.0$



(d) Using the Rayleigh Distribution, $\sigma = 2.0$

Figure 3.9 Total Relevance Value of the Response versus the Client Viewing Time Limit

A principal advantage of the proposed MMKP-based video personalization strategy is that it can satisfy multiple client-side resource constraints simultaneously whereas both the FKP-based and 0/1KP-based video personalization strategies can only satisfy a single client-side resource constraint at a time. In Figures 3.9 (a)-(d), the client-side resource constraint is the client's viewing time limit. Figure 3.10 shows the experimental results of the MMKP-based personalization strategy when the client has two resource constraints, i.e., a viewing time limit and a limit on the total amount of data received. The *Zipf* function with $s = 0$ is used in this case. The received data is limited to at most 3 KBytes for each second of the received video stream. The data limit constraint is held constant (at 3 KBytes for each second of the received video stream) whereas the viewing time limit constraint is varied. As seen in Figure 3.10, when the viewing time is less than 150 seconds, the response to the client's request contains no video segment, resulting in a null response. This is so because in each of the *content groups*, there is no *video item* of size less than 450 Kbytes (= 3 Kbytes per second × 150 seconds). When the client's viewing time limit is large enough (greater than 150 seconds in our experiment), it is possible to include video segments or video summaries which satisfy the data limit constraint in the response to the client's request. It is clear that when both constraints need to be satisfied, the response contains less video information compared to the case wherein the client viewing time is the only constraint. Thus, the solutions obtained when only a single resource constraint is satisfied at a time and when both resource constraints are simultaneously satisfied, are substantially different. The FKP-based and 0/1KP-based personalization strategies cannot handle such multiple constraints simultaneously and are forced to satisfy individual constraints one at a time. Since different resource constraints, when employed individually, yield different solutions, determining the optimal combination of these solutions to satisfy multiple resource constraints

simultaneously becomes an important (and difficult) issue in the case of the FKP-based and 0/1KP-based personalization strategies. This issue is obviously moot in the case of the MMKP-based personalization strategy since it is inherently equipped to satisfy multiple client-side resource constraints.

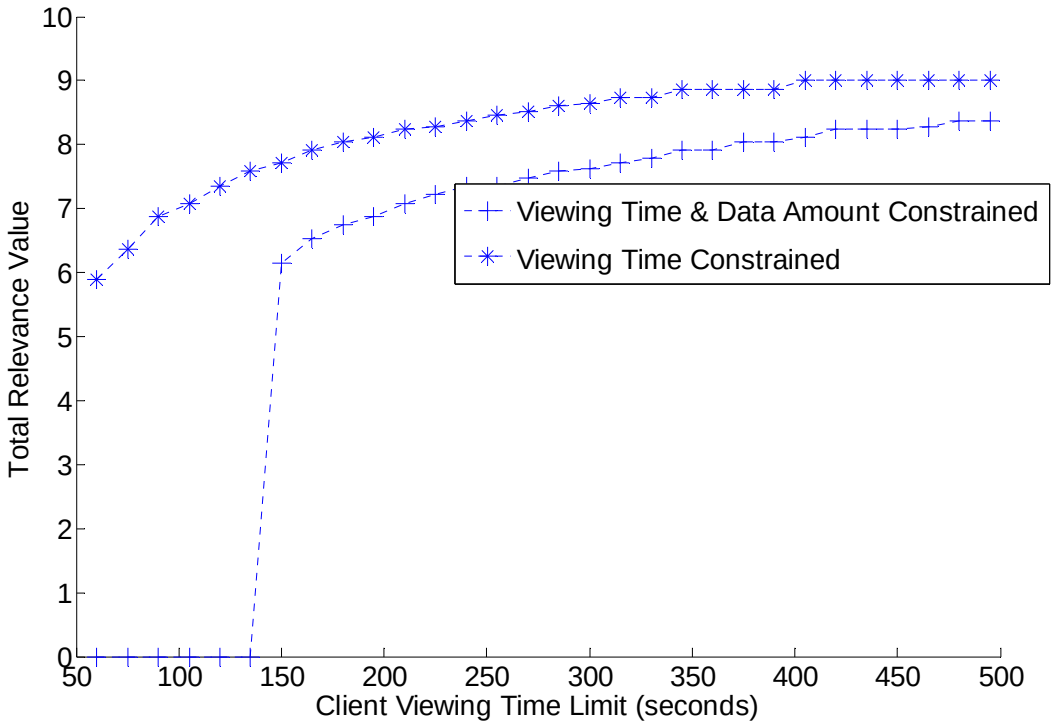


Figure 3.10 Performance of the MMKP-based Personalization Scheme under the Viewing Time Limit Constraint and under both, the Viewing Time Limit Constraint and the Data Limit Constraint (≤ 3 Kbytes for each second of received video)

The proposed 0/1KP, FKP and MMKP-based video personalization algorithms are implemented on a Dell Precision workstation with dual 3.19GHz CPUs and 2.0 GB of RAM. In the video database, there are 172 video content groups. Each video content group has three items. When the client specified viewing time is 90 seconds, it takes 31 milliseconds, 16 milliseconds

and 1234 milliseconds for the video personalization algorithm modeled along the 0/1KP, FKP and MMKP respectively.

3.6 **Conclusions**

In order to provide a resource-constrained client with its preferred video contents, it is often necessary to perform video personalization. The original video contents are personalized or adapted in order to best fulfill the client's request while simultaneously satisfying various client-side system-level constraints, such as viewing time limit, data limit, transmission bandwidth etc. In the proposed scheme, video segments are indexed and summarized at multiple levels of abstraction. In order to compare the information content of the transcoded or summarized video segments relative to their original versions, certain empirical mapping functions are employed. These mapping functions share a common characteristic, i.e. the incremental gain in the information content of a video segment diminishes with its total duration. This is in conformity with the commonly observed law of diminishing marginal return. The Zipf's law-based, the sigmoid function-based and the cumulative Rayleigh distribution-based mapping functions are used to compute the relevance values of video summaries given their time durations and the relevance values of the corresponding original video segments.

The primary task of the video personalization subsystem is to generate an optimal response to the client's request, i.e., one which maximizes the total relevance value of the response, while simultaneously satisfying multiple client-side system-level constraints. The video personalization problem is formulated as one of constrained optimization that is modeled along various versions of the classical Knapsack Problem (KP). The 0/1 Knapsack Problem (0/1KP)-based and Fractional Knapsack Problem (FKP)-based video personalization strategies have been described in the literature. In this paper, we have formulated and implemented a Multiple-Choice Multi-

Dimensional Knapsack Problem (MMKP)-based video personalization scheme as a means to maximally include as much relevant information as possible in response to the client's request, while satisfying *multiple* client-side system-level constraints. Experimental results show that when the beginning portions of a video segment contain more information than the rest of the video, the proposed MMKP-based optimization strategy yields a response with higher total relevance value compared to the 0/1KP-based and FKP-based video personalization approaches. Furthermore, this work represents a first attempt to provide a numerical performance metric for the evaluation of various video personalization strategies. Furthermore, the proposed MMKP-based video personalization strategy is shown to be capable of simultaneously handling multiple client-side system-level constraints. In constrast, the existing FKP-based and 0/1KP-based personalization strategies can only handle a single client-side constraint at a time.

Future work would include studies to validate the selection of the proposed empirical mapping functions. Future work would also seek to apply the proposed client-centered video personalization system to a wide collection of video data to test its robustness and extensibility. Furthermore, human subject evaluation of the generated responses to the client's request also needs to be explored in order to further validate this work.

CHAPTER 4

MULTIPLE CLIENT REQUEST AGGREGATION

4.1 **Introduction**

In the video personalization system, requests are from multiple clients. To fulfill each of these requests, the server needs to consume a certain amount of its resources, such as computing time and network bandwidth. When there are a large number of clients sending their requests to the server, the average client-experienced latency is long if every client request is processed individually.

The proposed client request aggregation strategy clusters similar client requests together such that the number of requests sent to the server is reduced, reducing the average client latency. The client requests are heterogeneous in multiple dimensions, i.e. they are different in their video content preferences, and in client-side constraints. A multi-stage clustering strategy is proposed to group similar request together one dimension at a time.

Existing video personalization research in the literature only addresses the single-client situation. We proposed the multi-stage clustering-based request aggregation strategy to reduce the server load and hence to improve the client-experienced latency. Client service request aggregation techniques have been discussed in large scale interactive multimedia service systems [Bommaiah, 2000], [Wu, 2005], [Venkatesh, 1995], [Bradshaw1, 2003]. These applications are characterized by large client population and high access intensity. Most of the existing works address efficient utilization of network channel resources [Hua, 1997], [Sen, 1999]. In the

context of video personalization, the process of finding and compiling the personalized video content consumes significant server computing power. Hence it is necessary to reduce the number of client requests the server needs to process in order to improve the client experienced mean latency. We propose the multi-stage client request aggregation strategy to group similar requests to reduce the number of requests the multimedia server needs to process. Our work serves as a high-level extension of the existing video-on-demand works. The number of multimedia streams is reduced such that network-level delivery techniques can be used to efficiently deliver multimedia streams to clients.

Different approaches to cluster data have been investigated [Jain, 1999]. Cluster analysis is a well-established research area in pattern recognition that has numerous applications in the areas of archaeology, astronomy, biology, medicine, market research, and psychiatry, among others [Willett, 1988],[Everitt, 2001]. Cluster analysis is also considered as one of the unsupervised machine learning methods, which do not use class information in learning. Most of the cluster analysis research in the area of information retrieval has considered the clustering of terms [Salton, 1989], and query expansion [Baeza, 1999]. K-means and hierarchical clustering are the most commonly used algorithms.

The hierarchical clustering algorithm has been applied in information retrieval [Everitt, 2001], and its time complexity is $O(n^2)$. The k-means algorithm is popular because it is easy to implement, and its time complexity is $O(n)$, where n is the number of patterns [Faber, 1994]. In order to measure the dissimilarity between semantic terms, cosine dissimilarity is commonly used in text segmentation and computational linguistics [Malioutov, 2006].

The remainder of the chapter is organized as follows. Section 4.2 provides details of the proposed multi-state client request aggregation strategy. In Section 4.3, experimental results of

the proposed multi-stage client request aggregation are provided as well. Section 4.4 concludes the paper with an outline for future work.

4.2 Client Request Aggregation

4.2.1 K-mean Clustering Algorithm

The k-means is the most commonly used clustering algorithm since it is easy to implement and effective in many applications. Its complexity is $O(n)$, where n is the number of patterns. K-means clustering tries to partition data points such that the within-cluster distance is minimized. The k-means algorithm is described as follows.

1. Randomly initialize N cluster centers.
2. For each data point, find the closet cluster center to the point.
3. Calculate the centroid of each cluster.
4. Use the centroids as new cluster centers.
5. Repeat step 2, until the difference of centroid of clusters is less than a threshold.

Inputs to the k-means clustering algorithm include dissimilarity measures, clustering criterion and number of clusters. The dissimilarity measure is fundamental to the definition of a cluster. It measures the distance of every pair of elements. In our application, we propose to use the Euclidean and cosine distance for client constraint and video content preference clustering respectively. Consine distance has been used to measure dissimilarity of linguistic terms [Malioutov, 2006]. Client video content preferences are represented by linguistic terms too. Hence cosine distance is suitable for the case. Client constraints such as viewing time and bandwidth limits are numerical. It is suitable to use the Euclidean distance to measure the dissimilarity of them.

The k-means aims to minimize the sum of squared distances of elements to centroid of clusters by mapping elements to clusters (patterns). Since silhouettes can be calculated with any clustering algorithm and any dissimilarity, it is suitable in the context of client request aggregation [van der Laan, 2002] to measure the performance of clustering. It is a real valued function of the cluster labels that measures how similar elements are within clusters and how different elements are between clusters. The silhouette for a given element is calculated as follows. For each element *j*, calculate the average dissimilarity a_j of element *j* with other elements of its cluster. For each element *j* and each cluster *l* to which it does not belong, calculate the average dissimilarity b_{jl} of element *j* with the members of cluster *l*. The silhouette of element *j* is defined by the formula:

$$S_j = \frac{b_j - a_j}{\max(a_j, b_j)} \qquad (4.1)$$

The k-means clustering requires a user-specified number of clusters. Currently available methods of selecting the number of clusters consist of optimizing the average silhouette [Kaufman, 1990]. Heuristically, the average silhouette measures how well matched an element is to the other elements in its own cluster versus how well separated the clusters are. This measure ranges from +1 to -1. +1 indicates elements are very distant from neighboring clusters, while -1 indicates elements are probably assigned to wrong clusters.

4.2.2 **Hierarchical Clustering Algorithm**

A hierarchical clustering is a sequence of partitions in which each partition is nested into the next partition in the sequence. The hierarchical clustering algorithm is described as the following pseudo code. Assume there are N data points,

1. Initially assign the N data points to N clusters.

2. Calculate pair-wise distances of all clusters.

3. Find the pair of clusters with the closet distance, and combine them into one cluster.

4. Repeat steps 2 and 3 until there is only one cluster.

A hierarchical cluster structure is converted into a partition by selecting one of the fusion or division levels of the structure, which is equivalent to "cutting" the hierarchical linkage structure at a particular height. In order to decide the optimal number of clusters, we apply the inconsistent coefficient [Kawa, 2005] to the dendrograms created by the hierarchical clustering. The inconsistent coefficient for the i-th linkage level is defined as follows.

$$c_i = \frac{\alpha_i - \overline{\alpha}}{\sigma_i} \qquad (4.2)$$

where $\overline{\alpha}$ and σ_i are the mean and standard deviation of the heights of linkages below level *i* respectively, α_i is the height of the linkage at the *i-th* level. Cutting the hierarchical linkage structure at height between level $\max(c_i)$ and the level immediately before $\max(c_i)$ yields the best partition. Figure 4.1 shows an example of finding the best cut of the hierarchical linkage structure. Data points are (1,1), (1.2, 1.2), (0.95,0.95), (2,2), (2.3,2.3), (1.9,2.1), and (1.8, 1.9). The maximum inconsistency coefficient (=1.1406) is found at the first level. Thus the best cut is to divide the set of data points into two clusters.

Similar to the k-means clustering algorithm, the dissimilarity measure is fundamental to the definition of a cluster. In our application, we propose to use the Euclidean and the cosine distance for client constraint and video content preference clustering respectively.

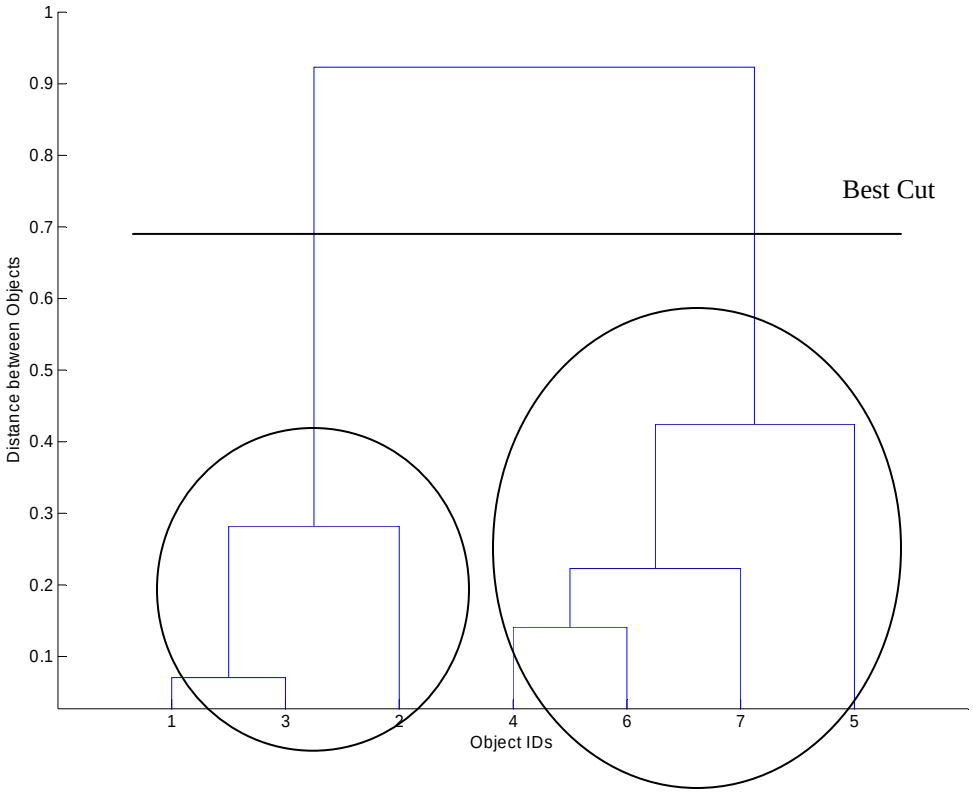


Figure 4.1. An Example of Finding the Best Cut of Clusters

4.2.3 **Multi-stage Client Request Aggregation**

Client requests for personalized videos are heterogeneous in nature. Heterogeneity can appear in the following three forms.

1. Arrival time heterogeneity: Client requests tend to arrive at different time instances.
2. Video object heterogeneity: The client video content preferences are different.
3. Client-side constraint heterogeneity: Each client request is associated with a set of client-side constraints. These client-side constraints tend to be different.

The goal of the proposed multi-stage client request aggregation strategy is to reduce the number of requests sent to the server. Since the multi-dimensional heterogeneous nature of the

client requests, it is more feasible to cluster them one dimension at a time. The proposed multi-stage client request aggregation strategy consists of the following steps, shown in Figure 4.2.

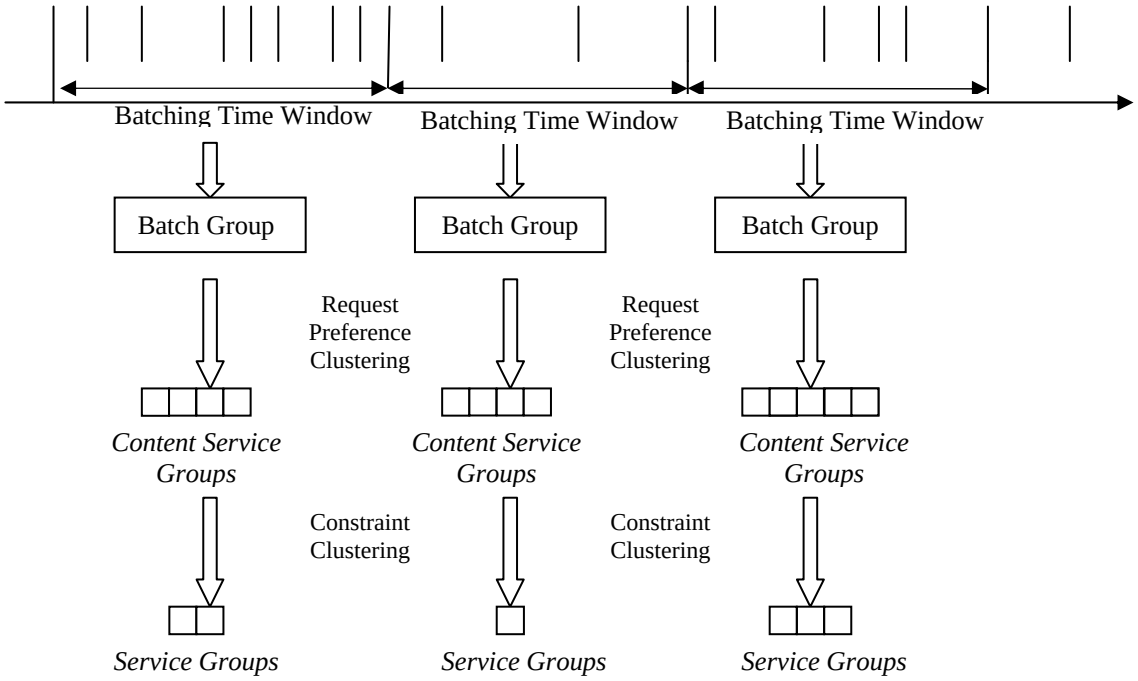


Figure 4.2 Multi-stage Client Request Aggregation

Step 1: *Batching by time*. Batching by time is to group multiple client requests arrived within a time window to form a group, termed *batch group*.

Step 2: *Preference clustering*. Client requests in a *batch group* with similar video content preferences are clustered into *content service groups* at this stage. Details of *preference clustering* are as follows.

Let q denote the ordered set of semantic terms used to index video segments, Q denote the ordered set of semantic terms clients can use in their requests to represent the clients' video

content preferences, and $q \subseteq Q$. Also assume the similarity matrix $S_{size(q)\times size(Q)}$ defines the similarity of semantic terms, where $0 \le s_{ij} \le 1$ is the semantic similarity of term $t_i \in q$ and term $t_j \in Q$. s_{ij} is calculated using the same *lch* algorithm. Let T be the semantic term used in the client video content preference, and the index of term T in the ordered set Q is *k*, then the client *content preference vector* P_c is defined as follows.

$$P_c = (s_{1k}, s_{2k}, ...s_{size(q)k}) \qquad (4.3)$$

where $s_{ik}, 1 \le i \le size(q)$ is the semantic similarity between term $t_i \in q$ and term $t_k \in Q$, and can be obtained from the similarity matrix $S_{size(q)\times size(Q)}$. The *preference clustering* algorithm uses the cosine dissimilarity of a pair of client query content preference vectors P_{c1} and P_{c2} to measure the distance of the pair of client video content preferences, as defined in equation (4.4).

$$sim(P_{c1}, P_{c2}) = \frac{P_{c1} \bullet P_{c2}}{\|P_{c1}\| \times \|P_{c2}\|} \qquad (4.4)$$

The *k-means* and hierarchical clustering algorithm is used to cluster client requests with similar content preference into a group, termed as *content service group*.

The number of the content service groups is adjusted in experiments. We propose to use the average silhouette as the criterion to determine the optimal number of content service groups the k-means clustering algorithm generates. For every $2 \le k < N_{unique}$, where N_{unique} is the number of unique client video content preferences in a *batch group*, the k-means partitions *content preference vectors* into *k* groups, and the average silhouette is calculated. The *k* which yields the maximum average silhouette value by k-means is denoted as $k_{optimal}^{preference}$.

In experiments, in order to control the performance of the client aggregation system, the number of preference clusters generated from a *batch group* is defined as follows:

$$N_{pc} = k_{optimal}^{preference} \times PAF \qquad (4.5)$$

where N_{pc} is the number of preference clusters generated from a *batch group*, k_c is the optimal number of client preference clusters, and PAF stands for Preference Aggregation Factor, $0 \le PAF \le 1$

Step 3: *Client-side constraint clustering*. Client requests within a *content service group* are clustered further based on client-side constraints to form a set of *service groups*. In the experiments presented in this work, client requests with close video viewing time limit are clustered together using the clustering algorithm.

In the context of client constraint clustering, the dissimilarity measure of client viewing time limit is the Euclidean distance. The optimal number of *service groups* generated by the client constraint clustering algorithm is denoted as $k_{optimal}^{VT}$, and is determined by the average silhouette of the k-means partition. In experiments, the number of *service groups* (viewing time limit clusters) is adjusted as follows.

$$N_{vtc} = k_{optimal}^{VT} \times CAF \qquad (4.6)$$

where N_{vtc} is the number of *service groups* (viewing time limit clusters) generated for a *content service group*. *CAF* stands for Constraint Aggregation Factor, $0 \le CAF \le 1$.

4.2.4 **Service Group Representation**

The multimedia server takes care of the set of client requests in a *service group* as a whole, i.e. only one video response is generated for all clients' requests in a *service group*. In order to represent the set of client requests in *a service group*, the video content preference which is the closest to the centroid of the *service group* is selected. The representative client viewing time and

allowed bandwidth constraint is the mean values of these parameters specified by the set of client requests in a *service group*.

4.2.5 **Performance Metrics**

In order to measure the performance of the proposed client request aggregation strategy, the overall client-server relationship is modeled as the following single queue with one server.

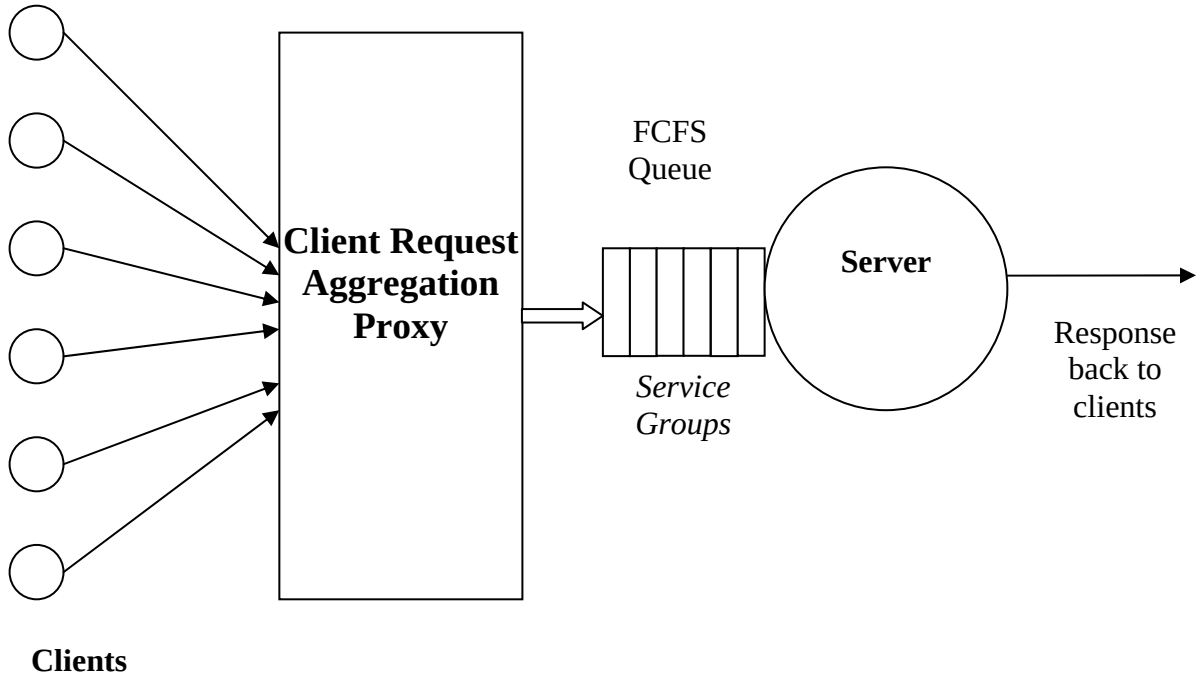


Figure 4.3 Single Queue Single Server Model for Client Request Aggregation Performance Evaluation

The following metrics are used to evaluate the performance of the proposed client request aggregation strategy.

1. Mean client-experienced latency (in seconds): The latency is defined as the duration of the interval between the time when the client sends a request to the server and the time when the client starts to receive video feedback from the server.

2. Mean client-experienced preference dissimilarity: Client-experienced preference dissimilarity is defined as the semantic dissimilarity between the video content preferences the clients send and receive. The *lch* semantic similarity measurement algorithm discussed in section 3.1 is used to calculate the dissimilarity. This metric measures how different the received video content is to the client's content preference.

3. Mean client-experienced viewing time difference (in seconds): The client-experienced viewing time difference is defined as the absolute value of the difference between the client-specified viewing time and the duration of the video feedback received from the server. The viewing time is the most important constraint. This metric measures how close the duration of the received video is to the client-specified duration.

4. Amount of data the server generates (in MB): This parameter measures the total video data amount the server processes in order to fulfill the clients' requests.

The duration of the batch windows, PAF and CAF are system adjustable parameters, will influence the overall performance of the proposed client request aggregation strategy. Experiment results show the relationships of the proposed performance metrics and the system adjustable variables.

4.3 **Client Request Aggregation Evaluation Results**

The single queue single server model discussed in section 4.2.5 is used to evaluate the performance of the proposed client request aggregation strategy. In our experiments, for every simulation test, 100 clients connect to the video personalization server via IEEE802.11b wireless networks. Each test is repeated 10 times. After a random delay, a client sends a video personalization request to the server. The durations of the random delays follow Poisson distribution $P(\lambda)$, $\lambda = 50$ seconds. In our experiments, clients specify viewing time and allowed

bandwidth limits in their constraints. In order to simulate mobile devices of different categories, the values of viewing time and allowed bandwidth limits follow a mixture of 5 normal distributions respectively. Means of the normal distributions of viewing time are 50, 100, 150 200 and 250 seconds respectively. The standard deviations of the normal distribution components for viewing time are set to 25 seconds. Means of the normal distribution components for allowed bandwidth are 20, 60, 100, 140 and 180 Kb/second respectively. The standard deviations of the normal mixture components are set to 20 Kb/second. The clients pick content preference terms from the set Q. Every semantic term in ^{Q} has the same probability to be selected. We measure the client-experienced latency, preference dissimilarity and viewing time difference for every client. On the server end, we measure the amount of video data generated in every test.

The system parameters, i.e. values of PAF and CAF, influence the performance of the client request aggregation strategy. When both PAF and CAF are set to one, clients get the closest match of what they preferred under their constraints, while the client-experienced latency is long. Adjustment of PAF and CAF can achieve desired balance between precision and client-experienced latency. We change the value of PAF and CAF systematically and measure the aggregation strategy's performance. On the server side, the duration of the batch windows is set to 20 seconds. Both k-means and hierarchical clustering algorithms are used in the multi-stage client request aggregation. In the following sections, we present and compare experimental results of the client request aggregation strategy with the two clustering methods.

4.3.1 **Client Aggregation using K-means**

Figures 4.4-4.8 show the relationships of the system parameters and the mean client-experienced viewing time difference, allowed bandwidth difference, the mean client-experienced video

content preference dissimilarity, the mean client-experienced latency and the amount of video data the server generates respectively.

It is observed in Figures 4.4-4.8 that when both the values of *PAF* and *CAF* are set to 100%, clients will get exactly their preferred contents with the duration of their viewing time limits and the with the allowed bandwidth, i.e. the mean client experienced preference dissimilarity, viewing time and allowed bandwidth difference are zero. However, as shown in Figures 4.7-4.8, the mean client-experienced latency is long and the amount of video data the server needs to process is large, since every single client request needs to be processed by the server individually.

In Figures 4.4 and 4.5, it is observed that the mean client-experienced viewing time difference and allowed bandwidth difference is primarily a decreasing function of *CAF* respectively. When the value of *CAF* is relatively small, the mean client-experienced viewing time and allowed bandwidth difference are large. This is because with small number of client constraint clusters, in a *service group*, the distance among client specified constraint parameters and client-experienced feedback parameters are big. When the number of client constraint clusters increases, i.e. when the value of *CAF* increases, it is shown in Figures 4.4 and 4.5 that both of the mean client-experienced viewing time difference and allowed bandwidth drop. However, the client-experienced viewing time difference and allowed bandwidth difference is not a monotonic decreasing function primarily of *PAF*. This is client preferences are clustered at a different stage. It is the 2-stage clustering that allows us to control the desired client-experienced preference dissimilarity and constraint differences separately.

As to the client-experienced preference dissimilarity, it is observed in Figure 4.6 that it drops when the value of *PAF* increases. Increasing the number of client constraint clusters does not

drop the client-experienced preference dissimilarity effectively. Figures 4.7-4.8 shows that either increasing the number of client constraint clusters or the number of client preference clusters increase the mean client latency and the amount of video data on server. It needs to be noted here that when both the numbers of client viewing time clusters and client preference clusters are small, the mean client latency is short. However, the prices of the short mean client latency are the larger client viewing time difference and client preference dissimilarity as well as larger amount of video data on the server.

In addition to the mean values of the client-experienced performance metrics, we also measure the standard deviations of these metrics in order to quantify the spread ranges of the metrics. Tables 4.1-4.4 list the mean and standard deviations of the client experienced viewing time, allowed bandwidth, preference dissimilarity, and latency. Mean values of these parameters represent the average client experiences, while the standard deviations of these parameters indicate the possible spread ranges of client experiences. For the sake of compactness, results of hierarchical clustering also are listed in the second line of each cell in the tables. Discussions of the meaning of them are provided in the following section.

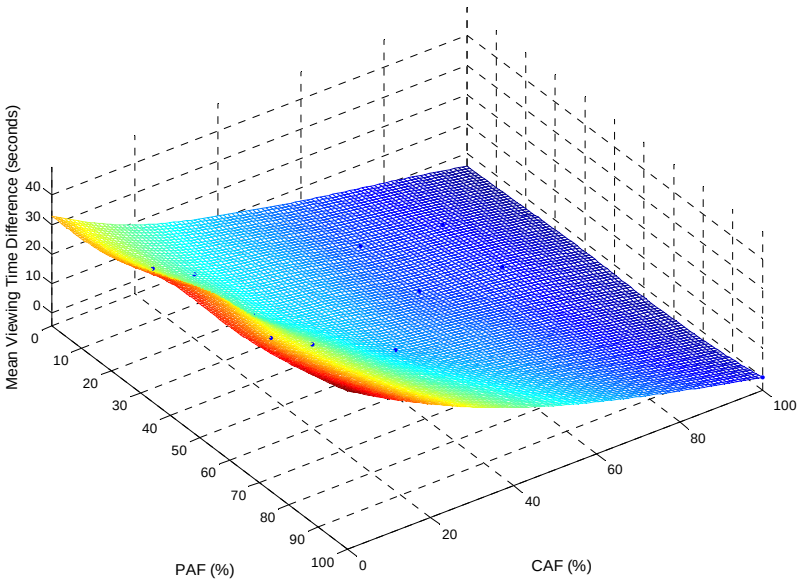


Figure 4.4 Mean Client Viewing Time Difference (seconds) vs. CAF and PAF: K-means Clustering

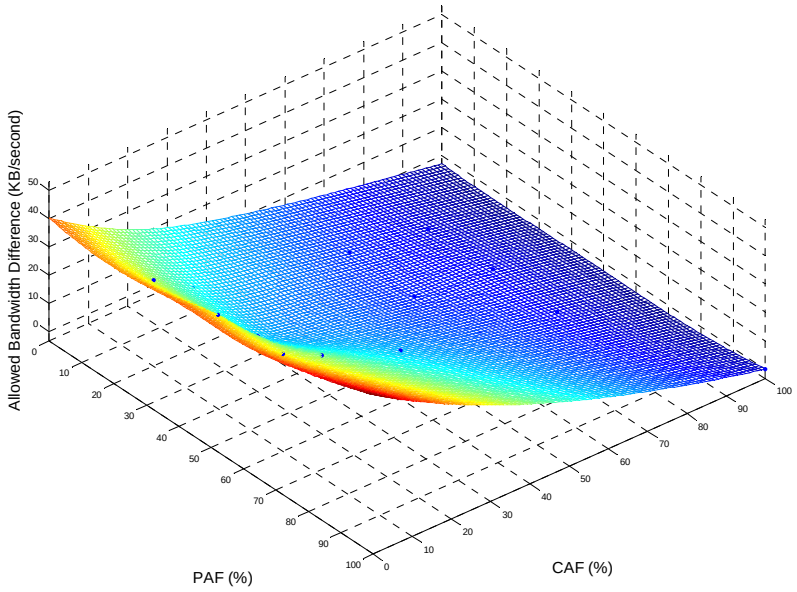


Figure 4.5 Mean Client Allowed Bandwidth Difference (KB/second) Difference: K-means Clustering

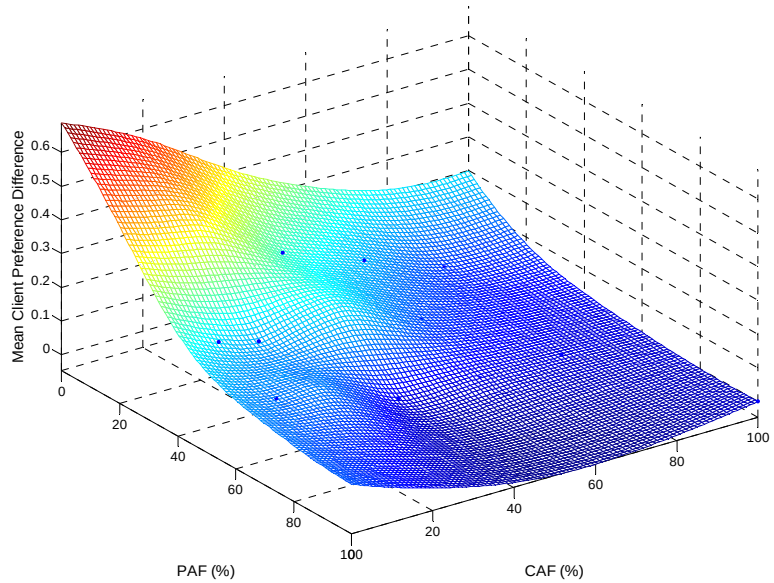


Figure 4.6 Mean Client Preference Difference vs. CAF and PAF: K-means Clustering

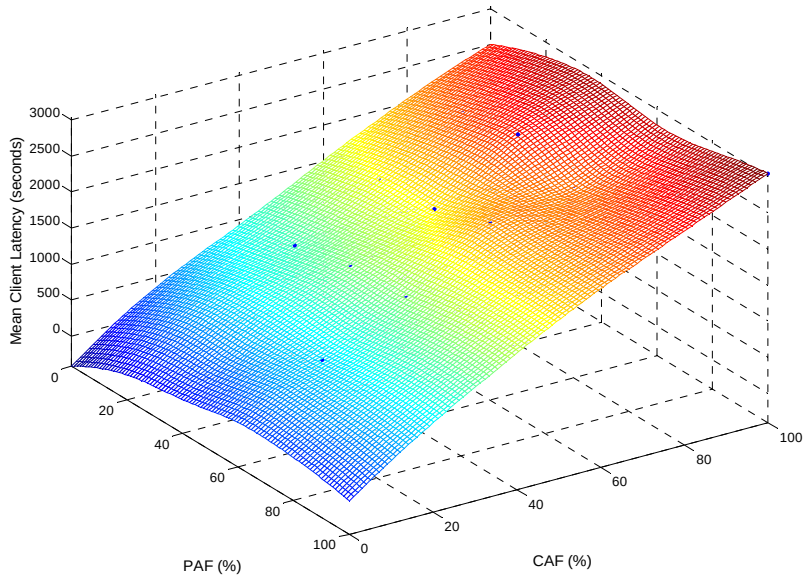


Figure 4.7 Mean Client Latency (seconds) vs. CAF and PAF: K-means Clustering

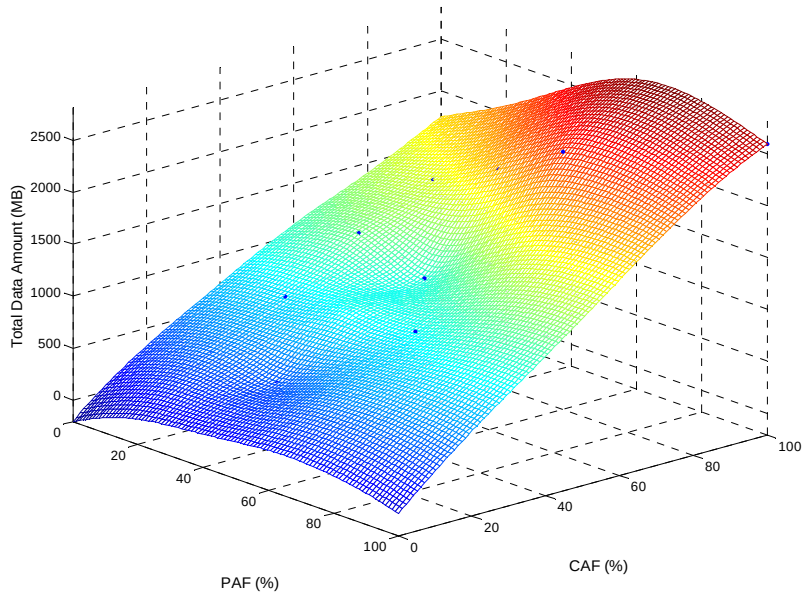


Figure 4.8 Amount of Data (MB) vs. PAF and CAF: K-means Clustering

Table 4.1: Mean and Standard Deviations of the Client-experienced Viewing Time Difference (seconds): Mean/Stdv

CAF / PAF	**10**	**20**	**40**	**60**	**80**
20	24.8/19.4 39.5/28.7	17.6/14.1 36.8/27.7	11.3/9.8 33.6/26.0	5.3/8.3 27.9/21.5	1.8/3.6 25.0/19.3
40	31.8/23.8 41.4/30.5	18.9/14.4 31.2/22.2	11.1/10.5 36.6/26.8	5.3/8.0 31.1/24.5	2.6/5.3 21.9/17.0
60	31.7/24.1 34.3/25.7	24.2/18.6 34.9/25.9	11.4/11.2 30.0/22.1	7.4/9.3 28.8/23.5	2.1/4.4 25.0/20.0

Table 4.2: Mean and Standard Deviations of the Client-experienced Allowed Bandwidth Difference (KB/second) : Mean/Stdv

CAF / PAF	10	20	40	60	80
20	27.1/22.7 40.1/28.5	18.1/15.2 42.8/29.2	10.9/9.8 38.2/27.9	5.6/6.7 27.0/22.2	1.5/3.3 24.2/20.0
40	29.7/24.0 36.6/28.4	16.1/12.4 41.2/29.1	9.6/9.3 38.7/28.6	5.0/6.8 35.6/26.3	2.6/4.9 25.1/22.0
60	31.1/26.1 39.9/27.7	24.4/20.5 38.7/27.9	13.6/14.6 36.0/27.6	7.4/9.7 31.0/24.8	2.4/5.0 23.7/20.6

Table 4.3: Mean and Standard Deviations of the Client-experienced Preference Dissimilarity: Mean/Stdv

CAF / PAF	10	20	40	60	80
20	0.44/0.98 0.50/0.97	0.42/0.91 0.29/0.78	0.26/0.75 0.33/0.82	0.17/0.62 0.47/0.94	0.08/0.44 0.41/0.89
40	0.20/0.65 0.14/0.54	0.17/0.62 0.11/0.47	0.20/0.66 0.05/0.31	0.09/0.46 0.05/0.34	0.04/0.30 0.07/0.39
60	0.13/0.52 0.09/0.42	0.13/0.54 0.04/0.28	0.02/0.22 0.04/0.28	0.02/0.21 0.01/0.15	0.01/0.17 0.07/0.37

Table 4.4: Mean and Standard Deviations of the Client-experienced Latency (seconds) Mean/Stdv

CAF / PAF	**10**	**20**	**40**	**60**	**80**
20	346/192 244/115	688/418 323/120	1106/645 372/164	1723/988 411/196	2325/1386 417/175
40	400/204 496/192	749/393 456/201	1300/695 331/164	1771/984 431/205	2491/1446 762/419
60	548/251 577/295	762/442 534/290	1342/756 509/265	2062/1128 566/344	2211/1294 787/451

4.3.2 **Client Request Aggregation using Hierarchical Clustering**

In order to compare the performance of the client request aggregation strategies using k-means and hierarchical clustering technique respectively, experimental evaluation results of the client aggregation using hierarchical clustering is presented in this section.

The single queue single server model described in Section 4.2.5 is used. Other experimental settings are kept the same as the settings for the client request aggregation using k-means clustering technique. We measure the influence of systems parameters, i.e. *PAF* and *CAF* , on the performance of the client request aggregation strategy using hierarchical clustering technique.

Figures 4.9-4.13 show the relationships of the system parameters and the mean client-experienced viewing time difference, allowed bandwidth difference, preference dissimilarity, latency and the amount of video data the server generates respectively.

When comparing figures 4.9-4.13 with figures 4.4-4.8, it is observed that the overall performances of the client request aggregation strategies using k-means and hierarchical clustering techniques are similar. When both *PAF* and *CAF* are set to 1.0, a closer comparison of figures 4.4 and 4.9, figures 4.5 and 4.10, and figures 4.6 and 4.11 shows that k-means clustering yields lower mean client-experienced viewing time difference, lower client-experience allowed bandwidth difference, lower client-experienced preference dissimilarity. The reason is that hierarchical clustering tends to generate bigger clusters than k-means [Korenius]. The maximum possible silhouette value is 1.0, which means each cluster generated by k-means consists of only one data point (equation 4.1). In the meantime, equation 4.2 shows that the smallest possible size of a cluster generated by hierarchical clustering is three. This is the reason why the k-means tends to generate smaller clusters than the hierarchical clustering. Comparisons of figures 4.7 and 4.12, and figures 4.8 and 4.13 confirm that since k-means generates smaller but more clusters, it results in smaller client-experienced viewing time and allowed bandwidth differences, but higher latency and greater server generated video data. On the other hand, the hierarchical clustering results in larger but fewer clusters. Hence less precision in client specified viewing time and allowed bandwidth, but lower latency and less server generated video data. Hence, if precision is preferred, tiny clusters would have been considered better than large ones, which are preferable in information retrieval. In this case, k-means is a better choice. If lower client latency and less generated video data is preferred, then hierarchical clustering is better.

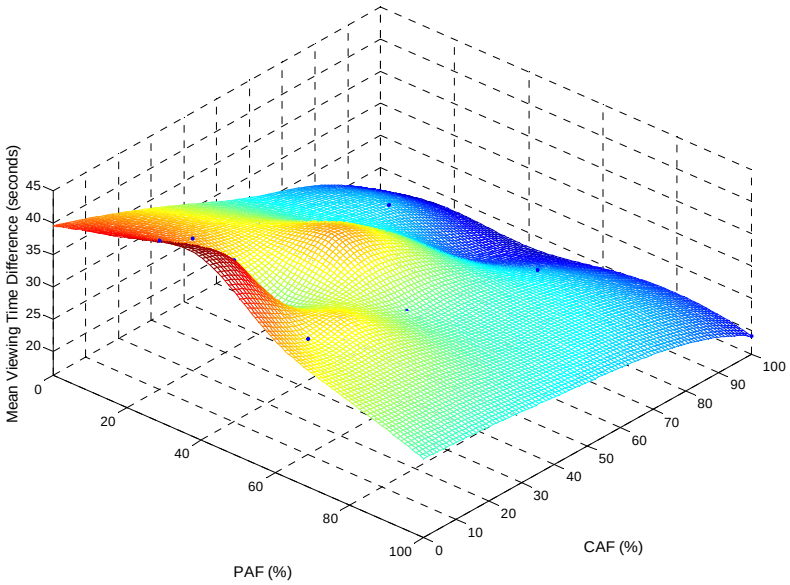


Figure 4.9 Mean Client Viewing Time Difference (seconds) vs. CAF and PAF: Hierarchical Clustering

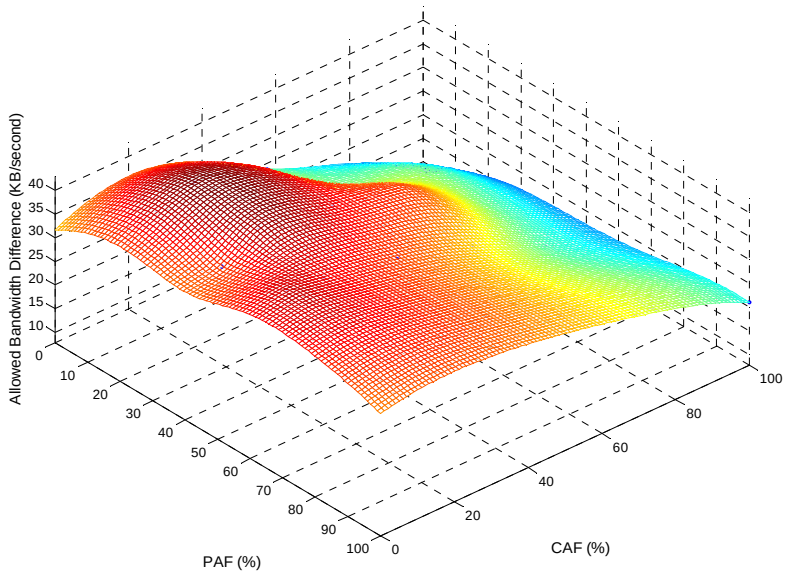


Figure 4.10 Mean Client Allowed Bandwidth Difference (KB/second) Difference: Hierarchical Clustering

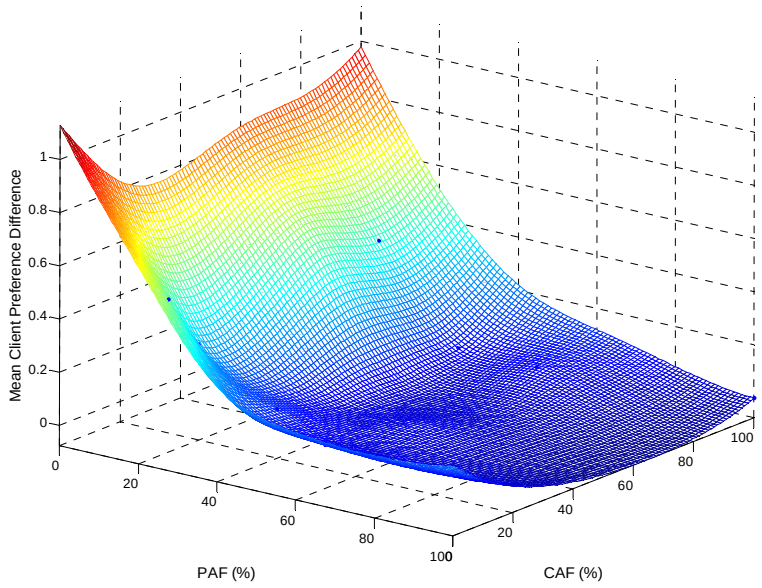


Figure 4.11 Mean Client Preference Difference vs. CAF and PAF: Hierarchical Clustering

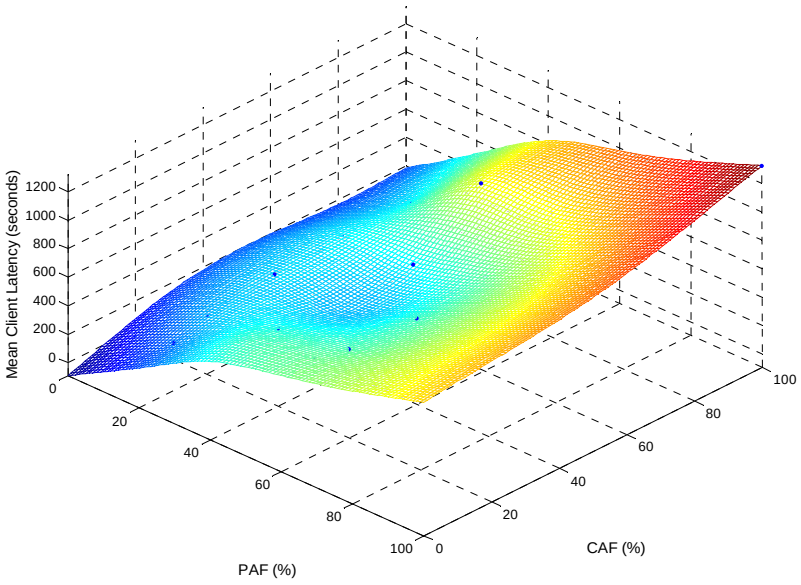


Figure 4.12 Mean Client Latency (seconds) vs. CAF and PAF: Hierarchical Clustering

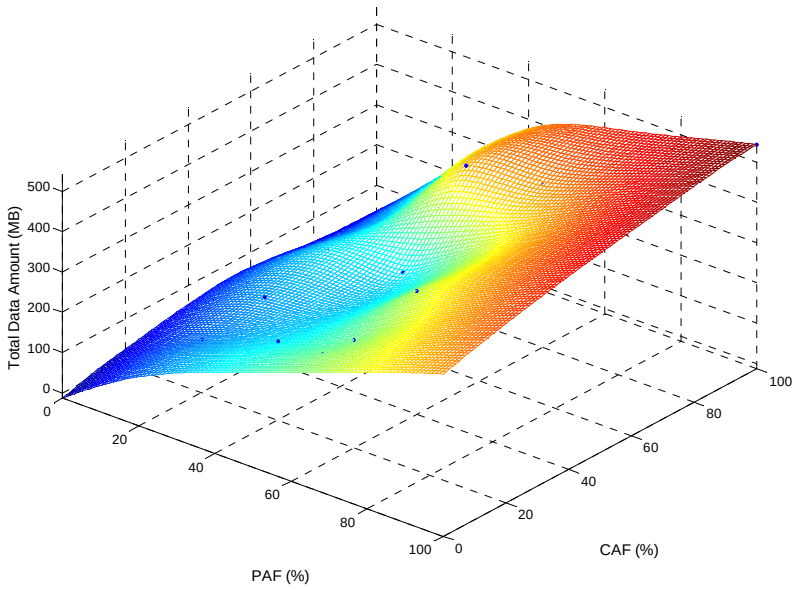


Figure 4.13 Amount of Data (MB) vs. CAF and PAF: Hierarchical Clustering

4.4 **Conclusions**

In the video personalization system, requests are from multiple clients. To fulfill each of these requests, the server needs to consume a certain amount of its resources, such as computing time and network bandwidth. When there are a large number of clients sending their requests to the server, the average client-experienced latency is long if every client request is processed individually. The client requests are heterogeneous in multiple dimensions, i.e. they are different in their video content preferences, and in client-side constraints. A multi-stage clustering strategy is proposed to group similar request together one dimension at a time. The proposed multi-stage client request aggregation strategy clusters similar client requests together such that the number of requests sent to the server is reduced, reducing the average client latency. Adjustments of Preference Aggregation Factor (PAF) and Constraint Aggregation Factor (CAF) can achieve desired balance among client-experienced latency and deviations of parameters of video

feedback from client specified constraints, i.e. content preference, viewing time and allowed bandwidth limitations. The multi-stage clustering strategy allows adjustments along individual dimensions separately.

Both k-means and hierarchical clustering techniques are used in the client request aggregation strategy. Comparison of the experimental performances of the client request aggregation using k-means and hierarchical clustering technique respectively shows that k-means tends to generate smaller but more clusters, and hence yields higher precision in client-experienced content preference, viewing time and allowed bandwidth limitation in video feedback, but longer latency and more video data generated. K-means is preferable when precision is important. On the other hand, when hierarchical clustering is used in the client request aggregation strategy, it tends to generate larger but fewer clusters, and hence yields lower precision in client-experienced content preference, viewing time and allowed bandwidth limitations, but shorter latency and less video data generated. Hierarchical clustering is a better choice when short latency is important.

CHAPTER 5

CLIENT-SIDE ENERGY-AWARE MULTIMEDIA DATA STREAMING

5.1 **Introduction**

The recent proliferation of multimedia capable mobile computing devices and networking technologies have created enormous opportunities for mobile device users to communicate with one another using multimedia streams. A necessary criterion for the mass acceptance of mobile devices is acceptable battery life of these devices. There has been dramatic improvement in energy-aware design of systems, both, in terms of hardware and software. Unfortunately, advances in hardware and software are not matched by a corresponding increase in battery life. Thus, the usefulness of these mobile devices in watching and/or hearing streaming multimedia is restricted by battery capacity. Future trends in battery technology do not promise dramatic improvements in battery capacity that will make this issue disappear. Consequently, hardware or software solutions need to be developed at the system or application level to prolong battery life.

Previous work on power management for mobile devices includes spin-down policies for disks [Wilkes, 1992], [Kumpf, 1994], [Douglis, 1995], [Helmbold, 1996], scheduling policies for reducing CPU energy consumption [Weiser, 1994], [Govil, 1995] and managing wireless communications [Imielinski, 1995], [Datta, 1997], [Kravets, 1998]. An IEEE 802.11b Wi-Fi connection is a popular way for mobile consumers to access the Internet wirelessly. The energy consumption of the wireless network interface can be significant, especially for smaller devices. Since media streaming applications are typically long running, the power consumption of these applications needs to be taken care of. Early work by Stemm et al. [Stemm, 1996] reports that the

network interface draws a significant amount of power. Although dependent on the specific machine and wireless device, the energy consumption of wireless communication devices can represent over 50% of total system energy consumption for current handheld computing devices and up to 10% for high-end laptops [Stemm, 1996]. Feeney et al. [Feeney, 2001] also report the energy consumption measurements of an IEEE 802.11b WNIC in an ad hoc networking environment and show that the energy consumption of the IEEE 802.11b WNIC has a complex range of behavior. Hence, it is important to look at techniques to reduce the energy consumed by the network interface used to download the multimedia stream.

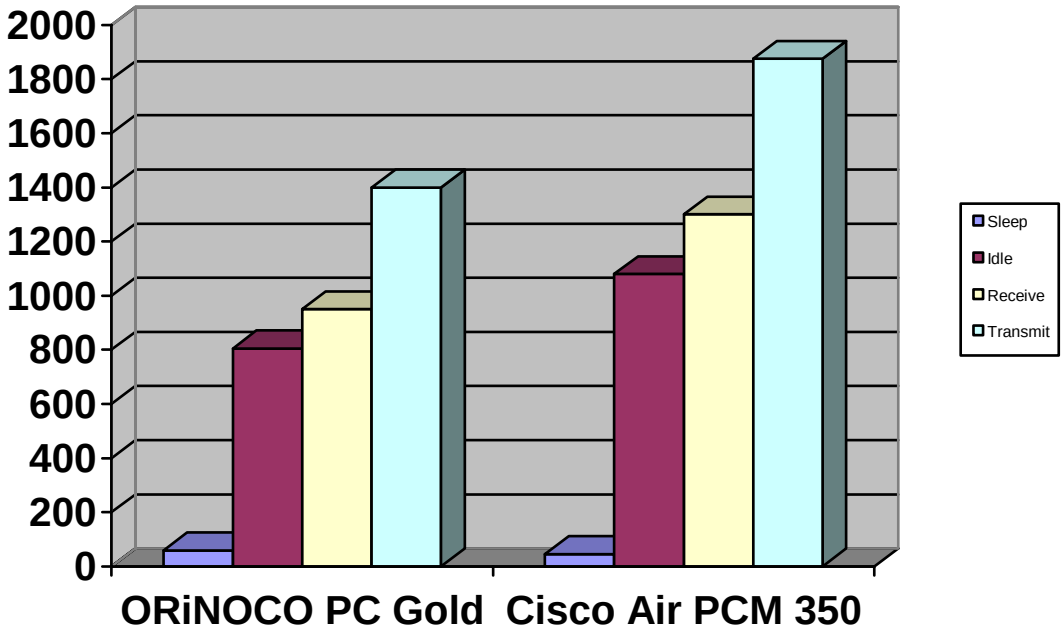


Figure 5.1 Energy Consumption Rates of Two WNIC's in Various States

The energy consumption rates of a wireless network interface card (WNIC) in the *sleep* state and in the *receive*, *transmit* or *idle* states are substantially different. Figure 5.1 shows the power

consumption rates of two popular WNIC's in the various aforementioned states [Shih, 2003]. The WNIC's energy consumption rate when receiving, transmitting data or when idling is substantially higher than when sleeping. Note that the WNIC cannot transmit, receive or buffer data in the *sleep* state.

Lorch et al. [Lorch, 1998] present a survey of software techniques for energy management. Havinga et al. [Havinga, 2000] present an overview of energy management techniques for multimedia streams. Aggarwal et al. [Aggarwal, 1998] describe techniques for processing video data for transmission under low power situations. A popular strategy to reduce the energy consumption of wireless network devices is by switching them to the lower power *sleep* state. Systems employing a strategy which enables switching of the WNIC to a low power consumption *sleep* state can achieve energy savings whenever possible without modifying the underlying application and without user-visible latency. Frequent switching to a low power consumption state also promises the added benefit of allowing the batteries to recover, thus exploiting the battery recovery effect [Chiasserini, 1999].

Media transcoding is a popular strategy used to reduce the stream fidelity. This strategy reduces the stream size, and hence reduces the amount of network traffic. Reducing the network traffic has the potential of reducing the total energy consumed. However, if care is not taken to return the WNIC to the low power-consuming state for as often and as long as possible, reducing the amount of transmitted data will have a negligible effect on the overall client energy consumption.

The basic principle underlying the proposed energy-saving approach is to predict the time durations during which to suspend communication by switching the WNIC to a *sleep* state. Our analysis of typical streams shows that the WNIC spends most of the time waiting for stream

packets in a higher energy consuming *idle* state. Even for a high bandwidth 2000Kbps stream, the WNIC spends over 56% of the time in the *idle* state; illustrating the potential for significant energy savings. Our policies operate on the multimedia client without explicit coordination or help from the multimedia server. Multimedia/video data is typically transmitted in the form of bursts of data packets with no-data periods between successive bursts. The bursty nature of the traffic is a consequence of the media streaming format and other network-related factors such as available network bandwidth, the buffering mechanism of the wireless access point and the traffic congestion control mechanism. Also, the bandwidth requirement of the multimedia stream is typically much less than what is provided by IEEE 802.11b, thus causing multimedia data traffic to appear bursty. The WNIC can be switched to its *sleep* state during these no-data intervals in order to save energy. Since we operate without explicit coordination with the server, this energy conservation strategy requires proper estimation of the time interval during which no data is expected to be received. If the WNIC is suspended too often or for too long a time duration during the wrong time periods, the users will miss the data sent to this client. On the other hand, if the WNIC is not suspended long and frequently enough, savings in energy consumption may not be appreciable.

Chandra [Chandra, 2003] describes a client-side history-based scheme that transitions the WNIC to a power saving *sleep* state during the no-data intervals of a multimedia stream. The history-based scheme predicts the length of the next no-data interval by computing the average of the lengths of the past *k* successive no-data intervals. The Microsoft Media format was observed to benefit immensely from this client-side history-based scheme on account of the fact that Microsoft Media transmits large data packets at fairly regular time intervals. The benefits in the case of the Real and Apple QuickTime media formats were less apparent on account of the

fact that the no-data time interval lengths were less regular. The work in this paper is a refinement of the client-side history-based scheme presented in [Chandra, 2003]. Specifically, we present a statistical linear prediction-based strategy to predict the occurrences and lengths of the no-data time intervals. These predictions are used to select the time periods during which the communication is suspended (i.e., the WNIC is powered down to its *sleep* state) in a client-server environment where multimedia streams are being transmitted from a server to a mobile, power-constrained client.

The remainder of the chapter is organized as follows. In Section 5.2, we describe the proposed linear prediction-based scheme and present a brief outline of the history-based scheme proposed in [Chandra, 2003] in the context of energy aware multimedia data streaming. In Section 5.3, we describe the experimental setup, evaluation methodologies, measurement metrics and experimental results that compare the performance of the proposed linear prediction-based scheme with that of the history-based scheme. In Section 5.4, we provide a detailed interpretation of the experimental results. In Section 5.5, we conclude the paper and outline directions for future research.

5.2 **Linear Prediction-based Approach**

In a client-server wireless network environment, data packets are transmitted by the server in the form of discrete bursts, as shown in Figure 5.2. This behavior is dependent on the particular multimedia streaming format used; for this work, we use the unmodified streaming formats of the Microsoft Media, Real and Apple QuickTime multimedia servers using the UDP protocol. Chandra [Chandra, 2003] provides more details on the video data transmission statistics. Between two successive bursts there is a time interval during which there is no data being transmitted by the server. We refer to this time interval as the no-data interval. In Figure 5.2, the

lengths of the two no-data intervals are $t_2 - t_1$ and $t_4 - t_3$ respectively. The sequence of these no-data interval lengths can be looked upon as a time series. Empirical observations suggest that the length of a no-data interval bears statistical correlation to previously observed no-data interval lengths. This provides the motivation for the formulation of a client-side statistical linear prediction-based scheme to predict future no-data interval length values based on previous observations.

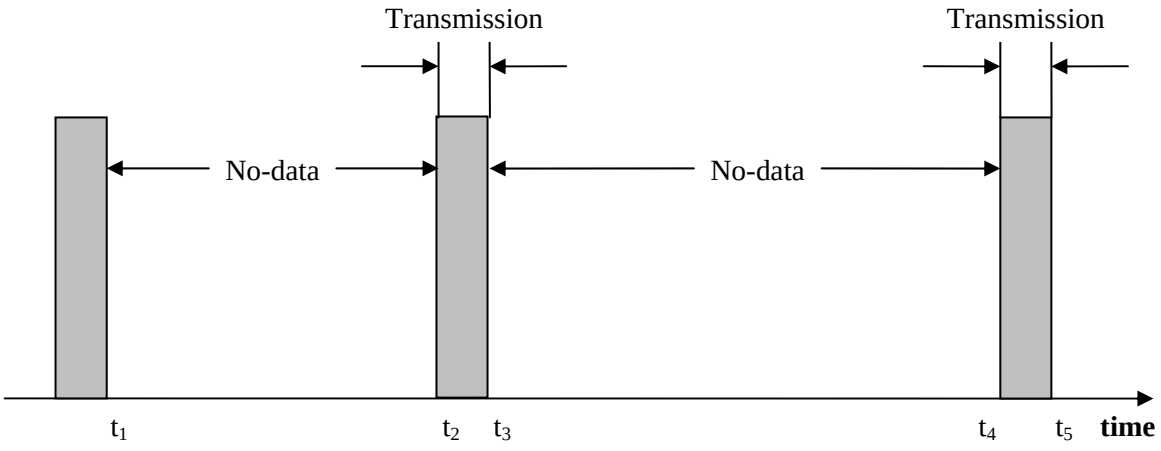


Figure 5.2 Simplified Stream Packet Transmission in a Wireless Network

Linear prediction is a mathematical operation where a future value of a time series is estimated as a linear function of previously observed samples [Hamilton, 1994]. A common representation of the linear prediction model is given by:

$$x'(n) = \sum_{i=1}^{p} a_i x(n-i) \qquad (5.1)$$

where $x'(n)$ is the estimated or predicted no-data interval length, $x(n-i)'s$ are the previously observed no-data interval length values, and $a_i's$ are the predictor coefficients. The error generated by this estimate is given by:

$$e(n) = x(n) - x'(n) \qquad (5.2)$$

where $x(n)$ is the true no-data interval length value and $x'(n)$ the predicted value of the no-data interval length. A linear predictor optimizes the estimate by minimizing the estimation error. The two adjustable parameters of a linear prediction model are the model order *p* and the width of the time window used for training. The algorithm used in our approach is the one proposed by Burg [Burg, 1978]. The appropriate values of *p* and the width of the training time window are chosen empirically.

In a client-server wireless network environment, if the predicted lengths of the no-data intervals are frequently longer than the actual ones, the user will experience packet losses in the data stream being downloaded because many data packets arrive at the client's WNIC while it is in the *sleep* state. On this account, it is useful to add a relatively small negative bias to the sleep interval lengths predicted by the linear prediction algorithm in order to lower the data drop rate. This ensures that the client's WNIC is transitioned to the *idle* state *before* the next data packet arrives. Thus, the biased estimate of the no-data interval length is given by

$$x_b'(n) = x'(n) - B = \sum_{i=1}^{p} a_i x(n-i) - B \qquad (5.3)$$

where $x_b'(n)$ is the biased prediction of the no-data time interval length. However, if the bias *B* is too large in magnitude, the resulting savings in battery energy may not be appreciable.

The history-based prediction scheme described in Chandra's previous work [Chandra, 2003] predicts the no-data interval length by averaging the observed no-data interval lengths over the

past *k* receive-idle cycles. It also varies the dependence of the prediction on past history by offsetting the predicted no-data interval length with a bias *B* as follows:

$$x'(n) = \frac{1}{k}\sum_{i=1}^{k} x(n-i) - B \qquad (5.4)$$

Note that the history-based prediction scheme can be looked upon as a special (i.e., degenerated) case of the linear prediction scheme where all the predictor coefficients are identical.

5.3 **Experimental Setup**

5.3.1 **System Description**

The experimental system consists of a multimedia server with a wireless access point, and a mobile client with a wireless network interface card (WNIC). The mobile client has a client-side proxy that is responsible for transitioning the WNIC to a low-power consumption *sleep* state during the predicted no-data time intervals. Ideally, since no data transfers are expected during the no-data time interval, there should be no loss of data. The traffic between the multimedia server's wireless access point and the mobile client is monitored by a monitoring station, which records the traffic flow in trace files. The multimedia stream used for our experiment is the *Wall* theatrical trailer. The *Wall* trailer is 1:59 minutes long and is digitalized to a high quality video stream.

Simulation of the client-side proxy is done using a typical WNIC power consumption model. We use the following published power parameters of a *Wavelan* 2.4 GHz wireless network interface card [Havinga, 2000]; *sleep* state: 177 mw, *idle* state: 1319 mw, *receive* state: 1425 mw and *transmit* state: 1675 mw. We assume that the transition from the *sleep* state to *idle* state takes 250 μseconds and the wireless network provides a useful bandwidth of 4 Mbps.

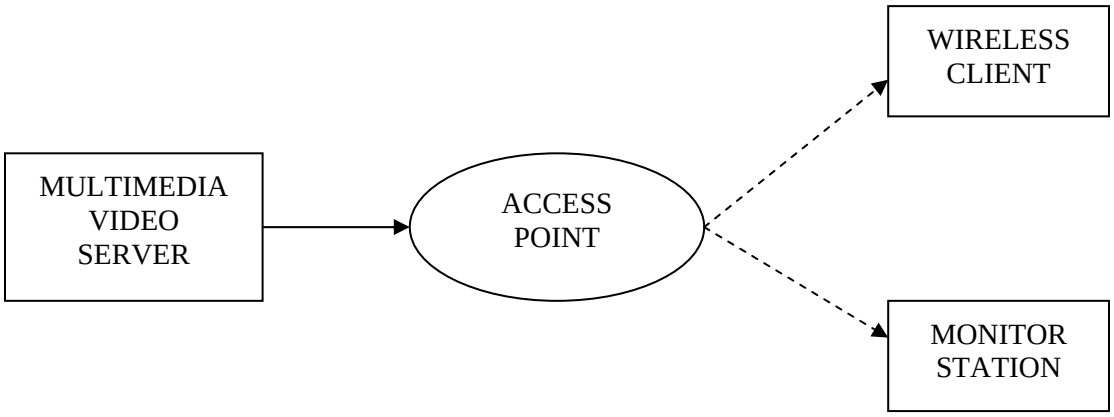


Figure 5.3 Experimental Setup

5.3.2 **Performance Metrics**

In order to measure the efficiency of our approach, the following performance metrics are used.

Total Energy Consumed: This is defined as the total amount of energy consumed (in mJoules) by the client-side WNIC to receive the streaming video data transmitted by the multimedia server. The goal is to minimize this metric when the client-side WNIC receives a video clip.

Energy Consumed per KB Received: This is defined as the amount of energy consumed (in mJoules) per Kilobyte of data received by the client-side WNIC. The goal of our experiment is to minimize this metric. As we will see in Section 5.4, due to the inaccuracy of client-side prediction, some of the streaming video data packets transmitted by the multimedia server will be dropped. This metric measures the energy efficiency of the client-side WNIC in terms of the energy expended for the amount of useful data it has received.

Drop rate: This is defined as the percentage of data dropped due to longer-than-actual predicted no-data interval lengths. The goal of our experiment is to minimize this metric as well.

5.4 **Experimental Results**

We use the wireless traffic trace files obtained by the monitoring station to perform the simulation. A WNIC cannot receive or buffer data when it is in the *sleep* state. If the predicted *sleep* interval is shorter than the actual one, the client-side WNIC wakes up at the end of predicted *sleep* interval and transitions to an *idle* state, ready to receive the burst of data packets. If the predicted *sleep* interval is longer than the actual one, the client-side WNIC sleeps through the end of the estimated *sleep* interval. Depending on the time when the WNIC wakes up, part of or the entire burst of data following the actual no-data interval is considered to be lost. In our experiments, we use the *Wall* theatrical trailer that is digitized to a high quality video stream. The Microsoft Media, Real and Apple QuickTime streaming formats are each used for the wireless transmission of the stream. Experimental measurements of the energy metric and drop rate are made as the value of the negative bias *B* is systematically varied.

To describe the power consumption model used to calculate the client-side WNIC energy consumption, we use the simplified stream data transmission model shown in Figure 5.2. The power consumption of the WNIC in each of the four states, i.e. *sleep*, *idle*, *receive* and *transmit* is denoted by P_{sleep}, P_{idle}, $P_{receive}$ and $P_{transmit}$ respectively. The predicted idle period is denoted by T_p, and the energy consumption of the WNIC is denoted by EC. Then the predicted *sleep* period falls in one of the following three cases, as shown in Figure 5.4.

Case 1: $T_p \le t_2 - t_1$, i.e. the predicted *sleep* period is shorter than or equal to the actual *sleep* period. During the time period T_p the WNIC's energy consumption is given by $T_p \times P_{sleep} = T_p \times 177mW$. The WNIC then wakes up and persists in the *idle* state until time instant t_2. The WNIC's energy consumption during this period is given by

$(t_2 - t_1 - T_p) \times P_{idle} = (t_2 - t_1 - T_p) \times 1319mW$. During the time period from t_2 to t_3, the WNIC receives data packets resulting in energy consumption given by $(t_3 - t_2) \times P_{receive} = (t_3 - t_2) \times 1425mW$. After having received the data burst, the WNIC goes back to the *sleep* state until the end of the next predicted *sleep* period. Hence the energy consumption in this case is given by

$$EC_1 = T_p \times P_{sleep} + (t_2 - t_1 - T_p) \times P_{idle} + (t_3 - t_2) \times P_{receiv} \qquad (5.5)$$

Case 2: $T_p > (t_2 - t_1)$ and $T_p \leq (t_3 - t_1)$, i.e. the predicted *sleep* period is longer than the actual no-data period but shorter than or equal to the no-data period plus the data transmission period. In this case, during the predicted *sleep* period T_p, the WNIC energy consumption is given by $T_p \times P_{sleep} = T_p \times 177mW$. Since the WNIC wakes up in the middle of data burst, part of the data in the burst is dropped. To receive the remainder of the data in the burst, the energy expended by the WNIC is given by $(t_3 - t_1 - T_p) \times P_{receive} = (t_3 - t_1 - T_p) \times 1425mW$. After having finished receiving the data burst, the WNIC goes back to the *sleep* state until the end of the next predicted sleep period. The total energy consumption in this case is given by

$$EC_2 = T_p \times P_{sleep} + (t_3 - t_1 - T_p) \times P_{receive} \qquad (5.6)$$

Case 3: $T_p > t_3 - t_1$, i.e. the predicted *sleep* period is too long. Consequently the WNIC wakes up during the next no-data interval and the data transmitted during the period $[t_2, t_3]$ is dropped. The WNIC persists in the *idle* state until the beginning of the following data burst. During the predicted *sleep* period T_p, the WNIC's energy consumption is given by $T_p \times P_{sleep} = T_p \times 177mW$. The WNIC then persists in the *idle* state until the beginning of the next

data burst. The energy consumption during this period is given by $(t_4 - t_1 - T_p) \times P_{idle} = (t_4 - t_1 - T_p) \times 1319mW$. The energy consumption in this case is given by

$$EC_3 = T_p \times P_{sleep} + (t_4 - t_1 - T_p) \times P_{idle} \qquad (5.7)$$

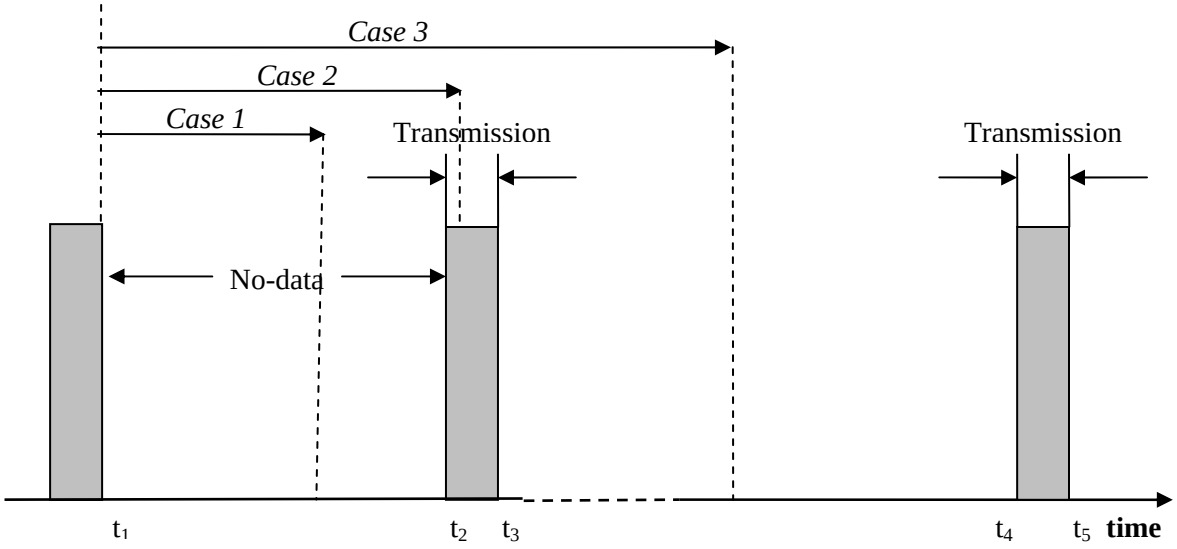


Figure 5.4 The Predicted No-data Period and the WNIC Energy Consumption Model

We compare the WNIC energy consumption results obtained using the linear prediction-based approach and the history-based approach described in [Chandra, 2003]. The results presented in this section are obtained when the *Wall* video segment is transmitted by the multimedia server in the Microsoft Media, Real and Apple QuickTime streaming formats. The multimedia server transmission bandwidth is set to result in a streaming bandwidth of 256Kbps and 512Kbps. The nature of the data bursts resulting from each of the aforementioned streaming formats is depicted in Figures 5.5(a), 5.6(a) and 5.7(a). As pointed out by Chandra [Chandra, 2003], due to differences in the characteristics of the underlying media stream formats, the time

series comprising of the lengths of the no-data intervals between successive data bursts exhibit different statistical properties. The Microsoft Media server transmits large data packets at fairly regular intervals. The Real and Apple QuickTime players tend to exhibit greater variation in the packet sizes and inter-packet arrival times. Figures 5.5(b), 5.6(b) and 5.7(b) show the histograms of the no-data interval lengths for each of the above streaming formats. To compare the performance of the history-based and linear prediction-based approaches, the histograms of the predicted no-data interval lengths obtained using the history-based approach and the linear prediction-based approach are presented in Figures 5.5(c), 5.6(c) and 5.7(c) and Figures 5.5(d), 5.6(d) and 5.7(d) respectively for each of the aforementioned streaming formats. A comparison of the distribution of the actual no-data interval lengths and the distributions of the predicted no-data interval lengths for each of the three streaming formats shows that the linear prediction-based approach yields no-data interval length distributions that are much closer to the actual distributions when compared to the history-based approach. When the distribution of the no-data interval lengths spans a broad range, the history-based approach is observed to be incapable of preserving the statistical properties of the actual no-data intervals.

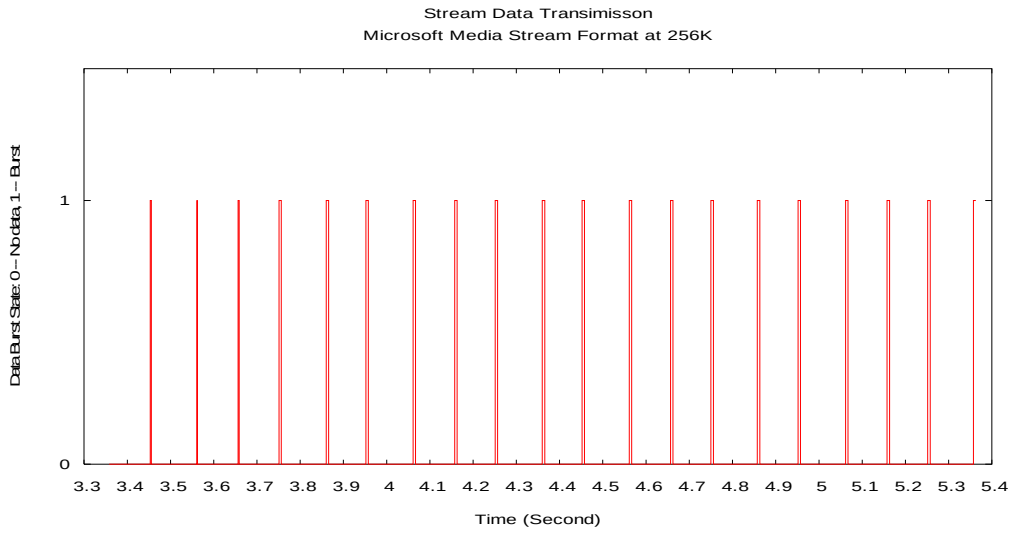


Figure 5.5(a) Data Burst Intervals for Microsoft Media Streaming Data at 256Kbps

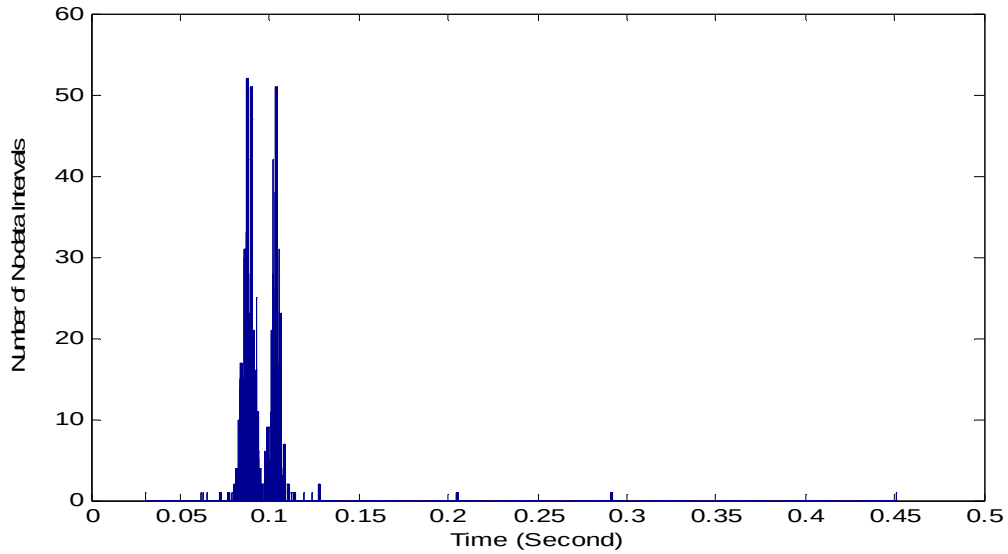


Figure 5.5(b) No-data Interval Length Histogram for Actual Data, Microsoft Media Format at 256Kbps

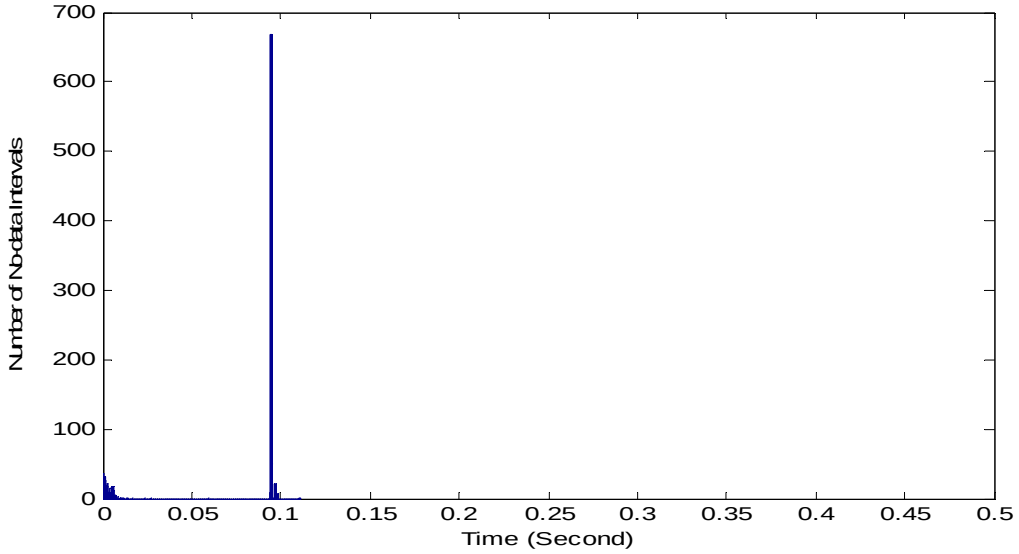


Figure 5.5(c) No-data Interval Histogram, History-based Estimation, Microsoft Media Format at 256Kbps

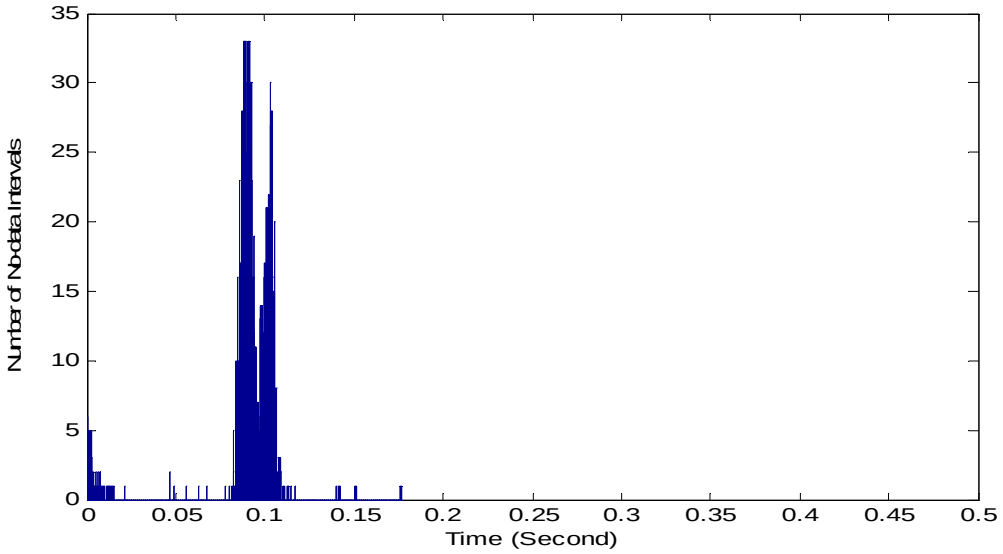


Figure 5.5(d) No-data Interval Length Histogram for Linear Prediction-based Estimation,Microsoft Media Format at 256Kbps

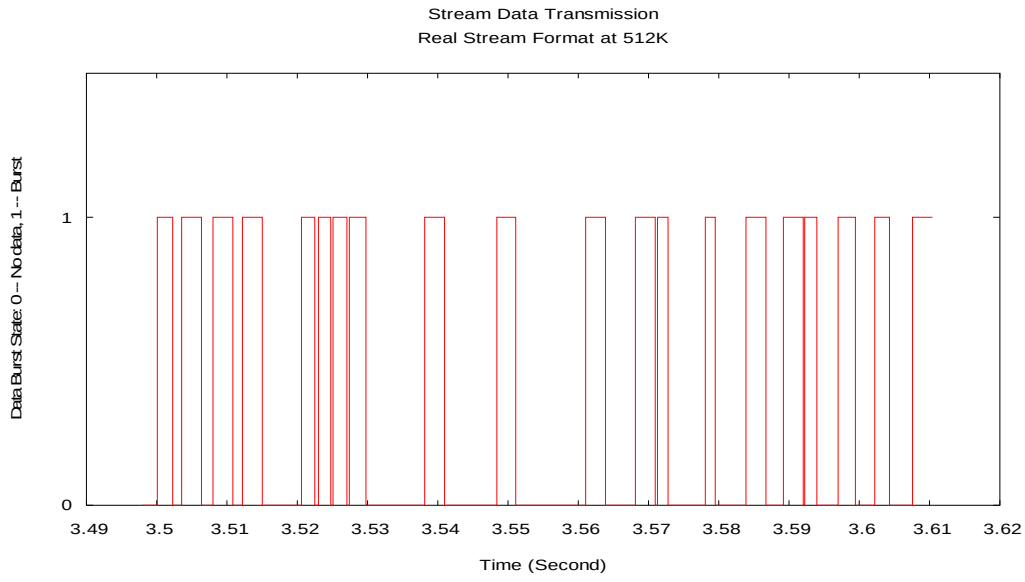


Figure 5.6 (a) Data Burst Intervals for Real Streaming Data at 512Kbps

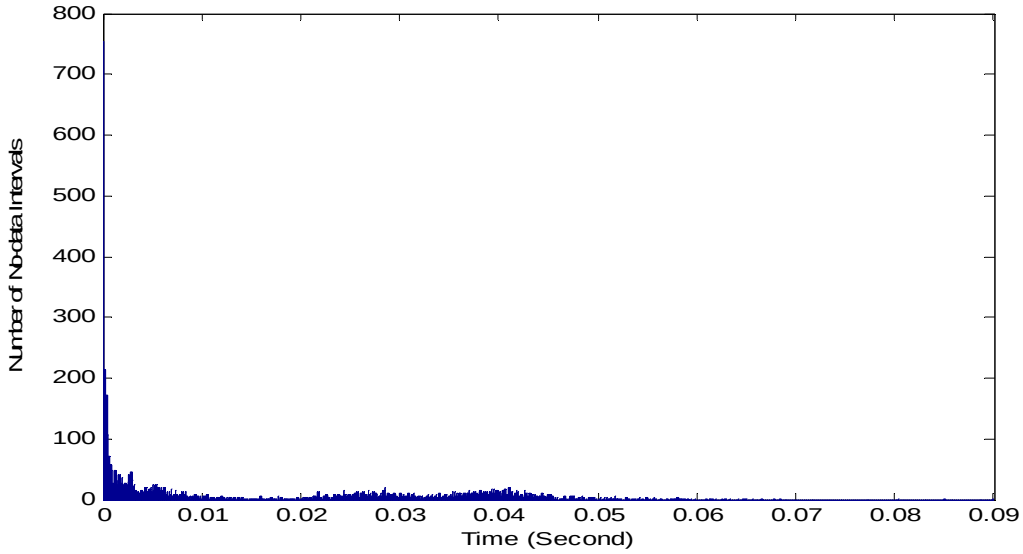


Figure 5.6(b) No-data Interval Length Histogram for Actual Data, Real Format at 512Kbps

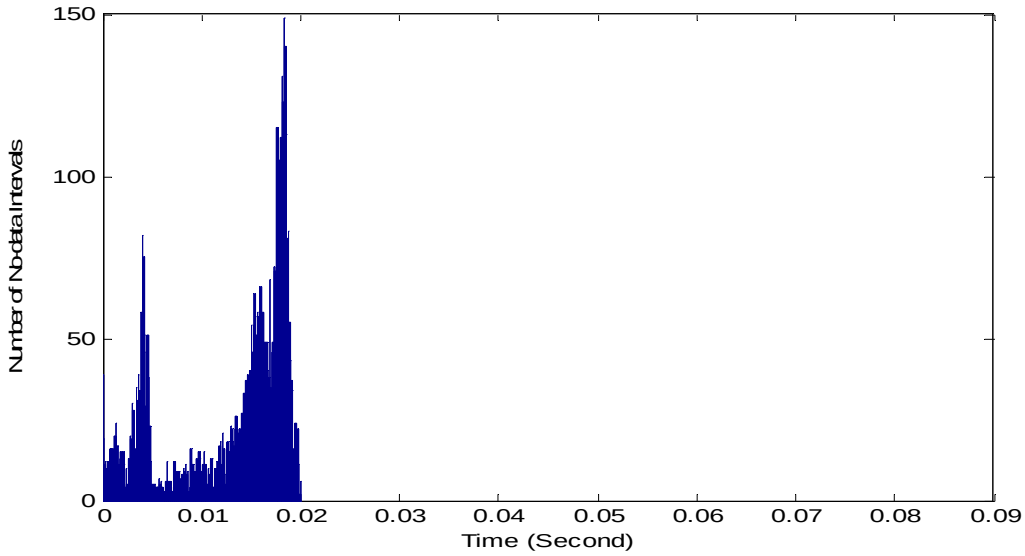


Figure 5.6(c) No-data Interval Length Histogram, History-based Estimation, Real Format at 512Kbps

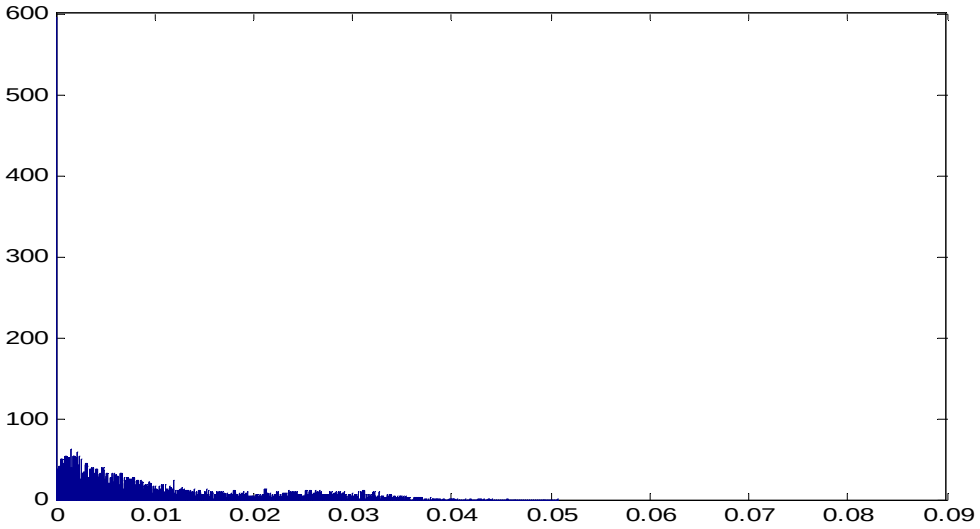


Figure 5.6(d) No-data Interval Length Histogram, Linear Prediction-based Estimation, Real Format at 512Kbps

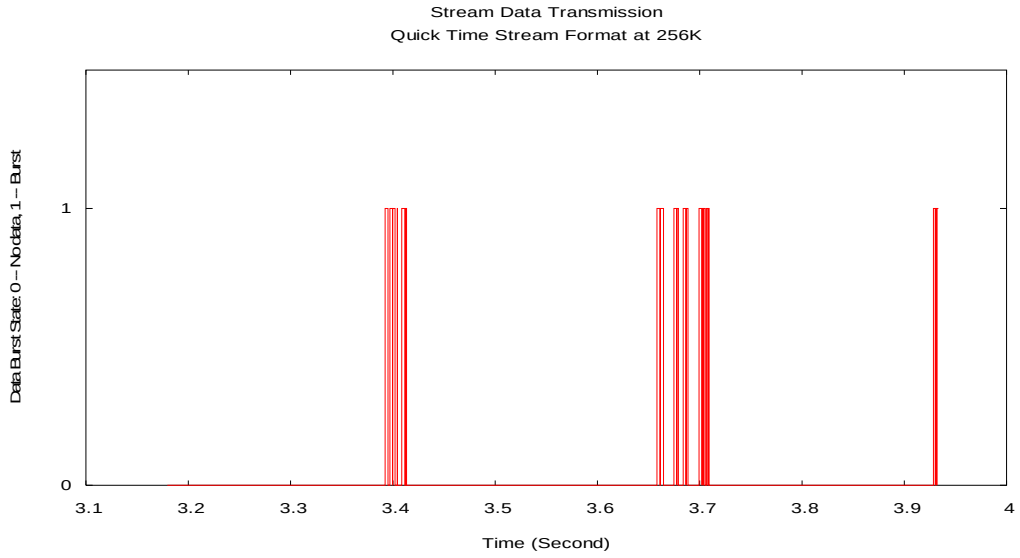


Figure 5.7(a) Data Burst Intervals for Apple QuickTime Streaming Data at 256Kbps

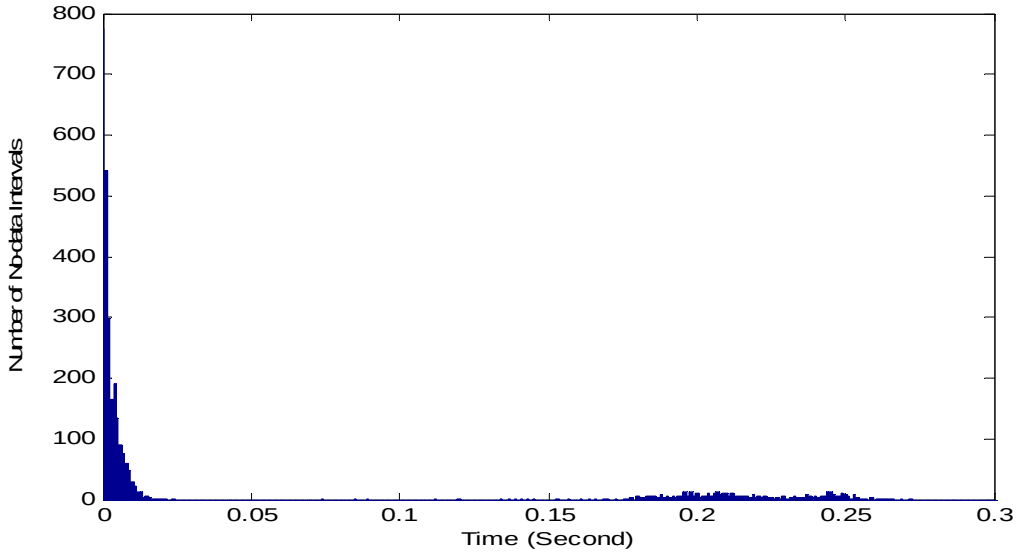


Figure 5.7(b) No-data Interval Length Histogram for Actual Data, Apple QuickTime Format at 256Kbps

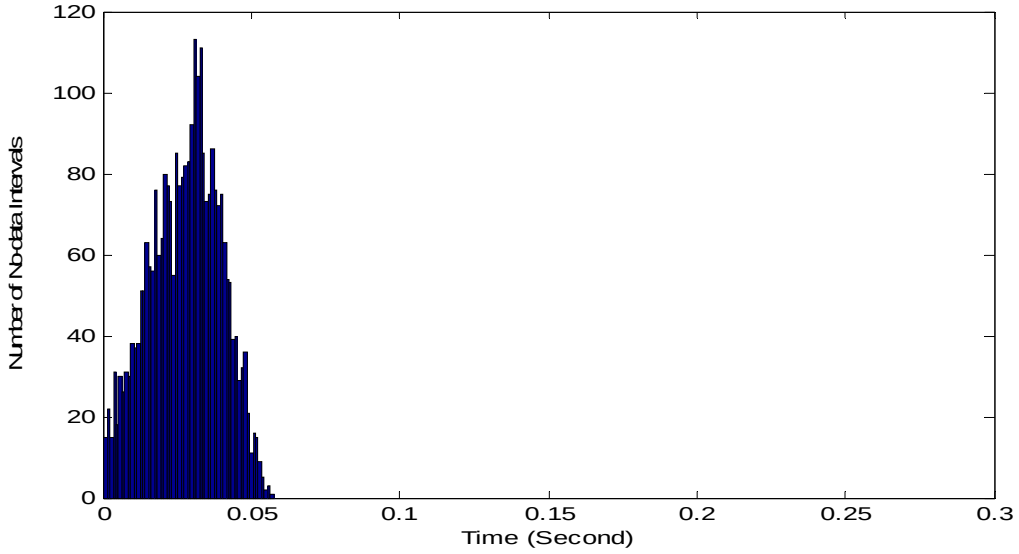


Figure 5.7(c) No-data Interval Length Histogram, History-based Estimation, Apple QuickTime Format at 256Kbps

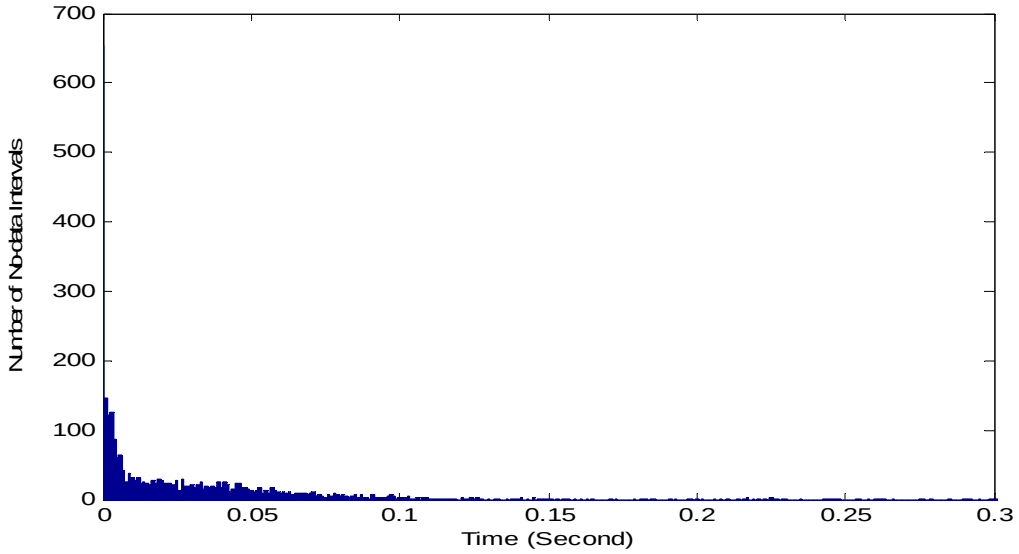


Figure 5.7(d) No-data Interval Length Histogram, Linear Prediction-based Estimation, Apple QuickTime Format at 256Kbps

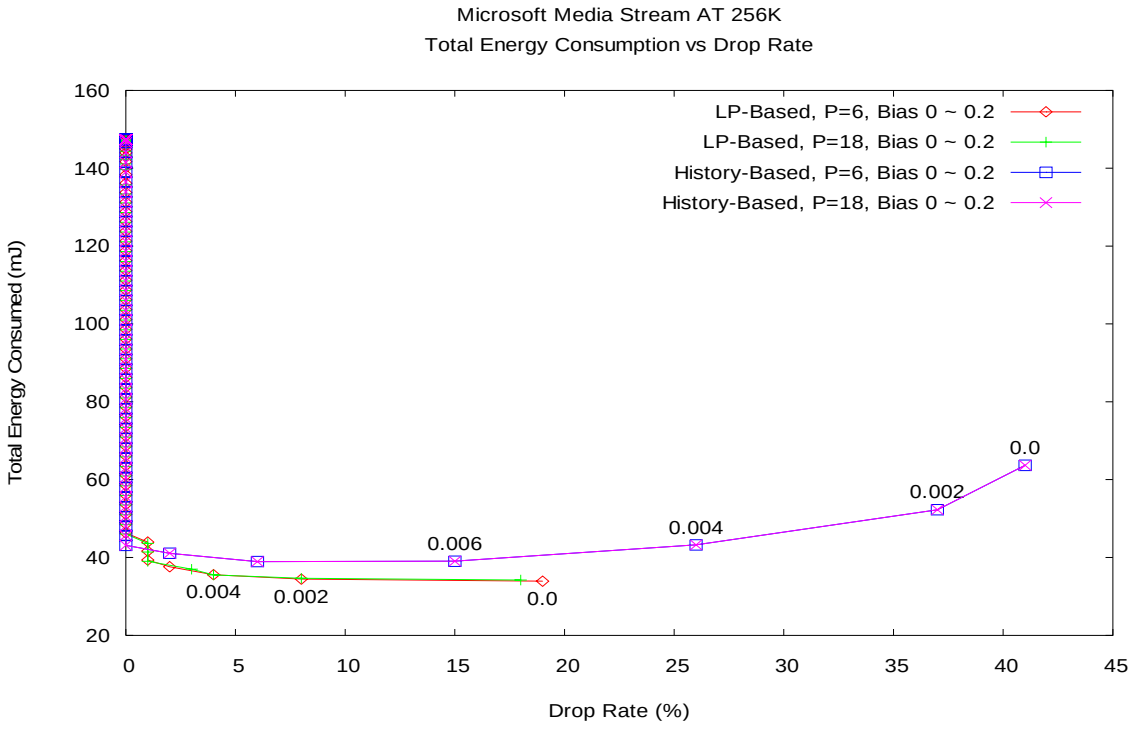


Figure 5.8 Total WNIC Energy Consumption vs. Drop Rate, Results of the Linear Prediction-based, and History-based Approach, Microsoft Media Format at 256Kbps.

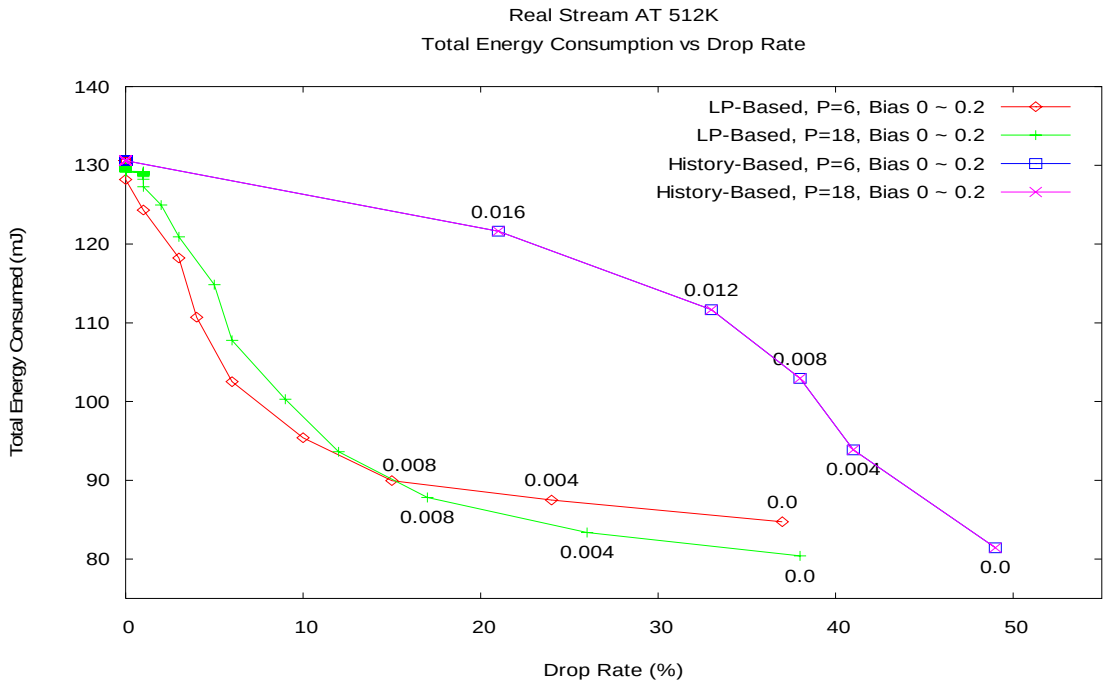


Figure 5.9 Total WNIC Energy Consumption vs. Drop Rate, Results of the Linear Prediction-based, and History-based Approach, Real Format at 512Kbps.

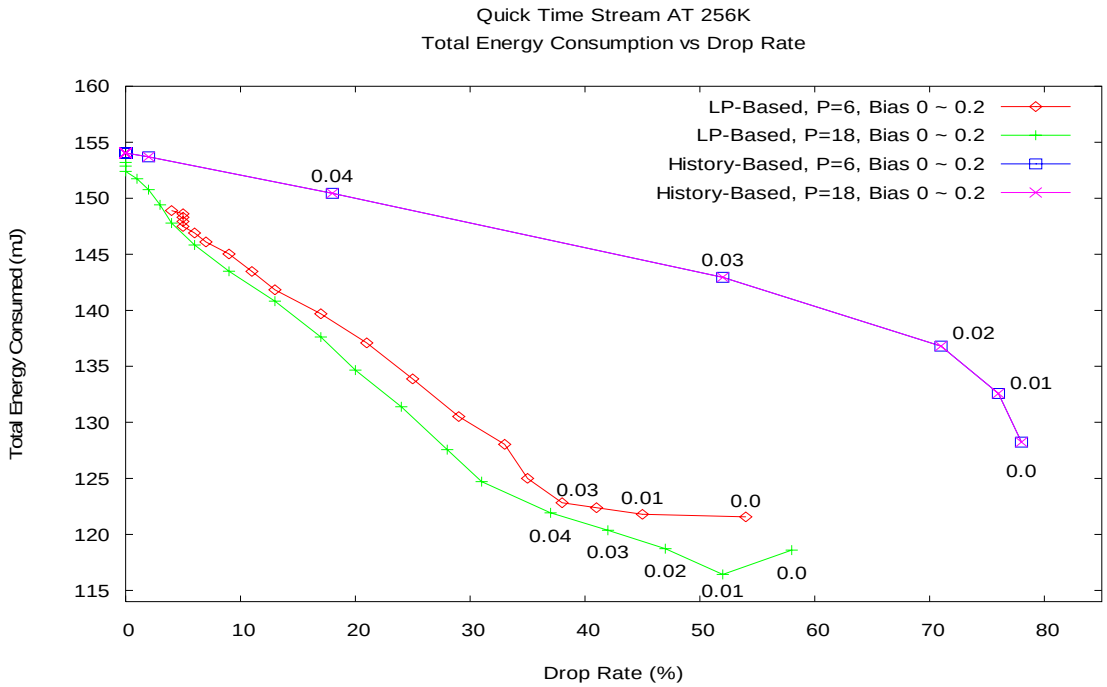


Figure 5.10 Total WNIC Energy Consumption vs. Drop Rate, Results of the Linear Prediction-based, and History-based Approach, Apple QuickTime Format at 256Kbps.

The values of the energy metric for the history-based and linear prediction-based approaches are compared for a given value of the drop rate. In order to obtain a fair comparison, the number of previous observations (number of previous actual no-data intervals) used in the estimation is set to be the same for both, the linear prediction-based approach and the history-based approach. The graphs in Figures 5.8 – 5.10 plot the client-side WNIC total energy consumption versus the drop rate. The magnitude of the negative bias added to the value of the predicted *sleep* interval length is indicated near the corresponding data point in each figure. Experiment results show that the linear prediction-based approach yields a lower total WNIC energy consumption compared to the history-based approach, for a given value of the drop rate when the number of previous observations used is the same. Moreover, when no negative bias is added, the linear prediction-based approach always yields a better performance than the history-based approach in terms of the drop rate and the total WNIC energy consumption. This implies that, statistically speaking, the linear prediction-based approach estimates the *sleep* interval length values for the client-side WNIC more accurately than does the history-based approach.

When a WNIC is in the *sleep* state, it neither can receive nor buffer the incoming data. Consequently, if the estimated *sleep* interval length is longer than its actual value, the client-side WNIC persists in the *sleep* state when the data burst arrives. Accordingly, this burst of data is considered lost. It is useful to decrease the data drop rate with a properly chosen negative bias that is added to the estimated length of the no-data interval. For a fixed magnitude of negative bias, the more accurate the estimation, the higher the percentage of predicted *sleep* intervals that are shorter than their actual counterparts. From Figures 5.8 – 5.10, we can see that for the same magnitude of negative bias that is added to the predicted *sleep* period, the linear prediction-based approach results in a lower data drop rate compared to the history-based approach. This can be

attributed to the fact that the *sleep* interval length value estimated by the linear prediction-based approach lies within a smaller neighborhood about the actual *sleep* interval length value. With the addition of a negative bias of relatively small magnitude, most of the estimated *sleep* intervals are observed to lie within their actual counterparts in the case of the linear prediction-based approach, thus resulting in a relatively low drop rate.

With no negative bias added to the predicted *sleep* interval length values, there is a higher probability that the predicted values of the *sleep* interval lengths are longer than their actual counterparts. Some of the predicted values are so long that entire data bursts are dropped by the client-side WNIC, as indicated by *Case 3* in Figure 5.4. In this situation, the client-side WNIC wakes up during the no-data interval, and persists in the *idle* state until the following data burst. The WNIC energy consumption in this situation is defined by equation (5.7). The energy consumed by the client-side WNIC during its *idle* state is given by $(t_4 - t_1 - T_p) \times P_{idle} = (t_4 - t_1 - T_p) \times 1319mW$ (Figure 5.4). Since the power consumption of the WNIC in the *idle* state is higher than in the *sleep* state if the no-data intervals between data bursts are long, the energy consumption due to overestimation of the *sleep* interval is large. A small amount of negative bias applied to the predicted values of the *sleep* interval lengths can reduce the probability of overestimation as defined by *Case 3* (Figure. 5.4) and therefore reduce the client-side WNIC power consumption and also the drop rate. The Microsoft Media and the Apple QuickTime streaming formats are characterized by long no-data intervals accompanying relatively short data bursts. As shown in Figure 5.8, in the case of the Microsoft Media format, when the bias magnitude lies in the range [0, 0.006], the total client-side energy consumption and the data drop rate exhibit a decreasing trend with increasing value of the bias magnitude for the history-based approach. As shown in Figure 5.10, a similar trend can be observed in the case of

the Apple QuickTime format when the bias magnitude values are in the range [0, 0.01] for the linear prediction-based approach that uses 18 previous observations.

When the bias magnitude value is chosen large enough such that all the predicted no-data interval length values comply with *Case 2* (Figure 5.4), then increasing the bias magnitude value results in a greater amount of data received by the client-side WNIC (i.e., a lower drop rate) at the expense of increased energy consumption. This explains the trend where the drop rate decreases but the total WNIC energy consumption increases with increasing bias magnitude value. This trend can be observed in Figure 5.9 in the case of the Real format for both, the history-based approach and the linear prediction-based approach. A similar trend can be observed in the case of the Apple QuickTime format (Figure 5.10) for the history-based approach and for the linear prediction-based approach when the bias magnitude value is greater than 0.01. However, once the bias magnitude value crosses a certain threshold value such that all the predicted no-data interval length values comply with *Case 1* (Figure 5.4) then any further increase in the bias magnitude value only increases the total WNIC energy consumption (since the client-side WNIC spends more time in the *idle* state waiting for the data burst to be received) without any decrease in the drop rate. This trend can be clearly observed in the case of the Microsoft Media format (Figure 5.8), for the history-based approach when the bias magnitude value exceeds 0.12 and for the linear prediction-based approach when the bias magnitude value exceeds 0.01. In the limiting case when the bias magnitude value equals the longest actual no-data interval length, the client-side WNIC will be in the *idle* or *receive* state for the entire duration of the streaming session. This is tantamount to the complete absence of any client-side prediction scheme. In this case the energy metric values for both, the history-based approach and

the linear prediction-based approach converge to same point, which corresponds to the absence of any client-side prediction whatsoever.

In order to compare the energy efficiency of the client-side WNIC when receiving video streams in different formats, the energy consumption of the client-side WNIC per KByte of data received is plotted as a function of the drop rate for all the three media streaming formats for both, the history-based approach and the linear prediction-based approach (Figures 5.11 – 5.13). The graphs in Figures 5.11 – 5.13 do not exhibit the same consistency as their counterparts in Figures 5.8 – 5.10 in terms of monotonicity of the function. This can be explained with the following analysis. If the total energy consumption of the client-side WNIC, EC_{total} (which is plotted versus the drop rate in Figures 5.8 – 5.10), is expressed as a function of the drop rate as follows:

$$EC_{total} = f(drop\_rate) \qquad (5.8)$$

then the energy consumed by the client-side WNIC for each KByte of data received, denoted by EC_{KByte}, is given by

$$EC_{KByte} = \frac{f(drop\_rate)}{B \times (1 - drop\_rate)} \qquad (5.9)$$

where B is the amount of data transmitted in the video stream. From equation (5.9) it can be observed that even if $f(drop\_rate)$ is monotonic with respect to $drop\_rate$, EC_{KByte} can still be non-monotonic with respect to $drop\_rate$. Nevertheless, the value of EC_{KByte} in the case of the linear prediction-based approach is observed to be much lower than that in the case of the history-based approach for a given value of $drop\_rate$ for all the three streaming media formats (Figures 5.11 – 5.13). Conversely, the drop rate in the case of the linear prediction-based approach is observed to be much lower than that in the case of the history-based approach for a

given value of EC_{KByte} for all the three streaming media formats (Figures 5.11 – 5.13). This shows that the linear prediction-based approach is more energy efficient than the history-based approach regardless of the streaming media format.

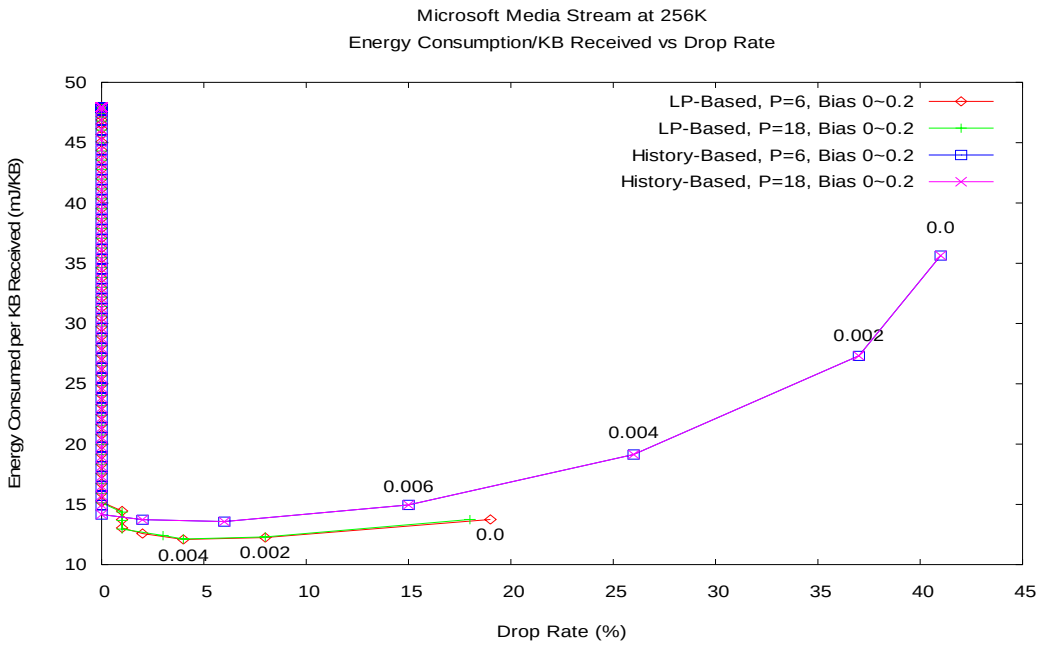


Figure 5.11 Energy Consumption per KByte of Data Received, Results of History- & Linear Prediction-based Approaches, Microsoft Media Format at 256Kbps

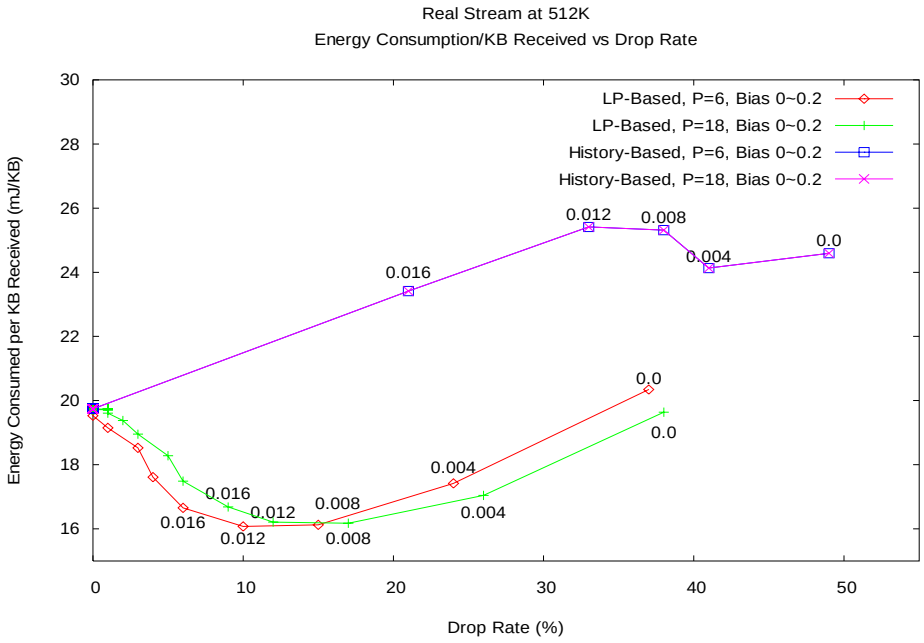


Figure 5.12 Energy Consumption per KByte of Data Received, Results of History- & Linear Prediction-based Approaches, Real Format at 512Kbps

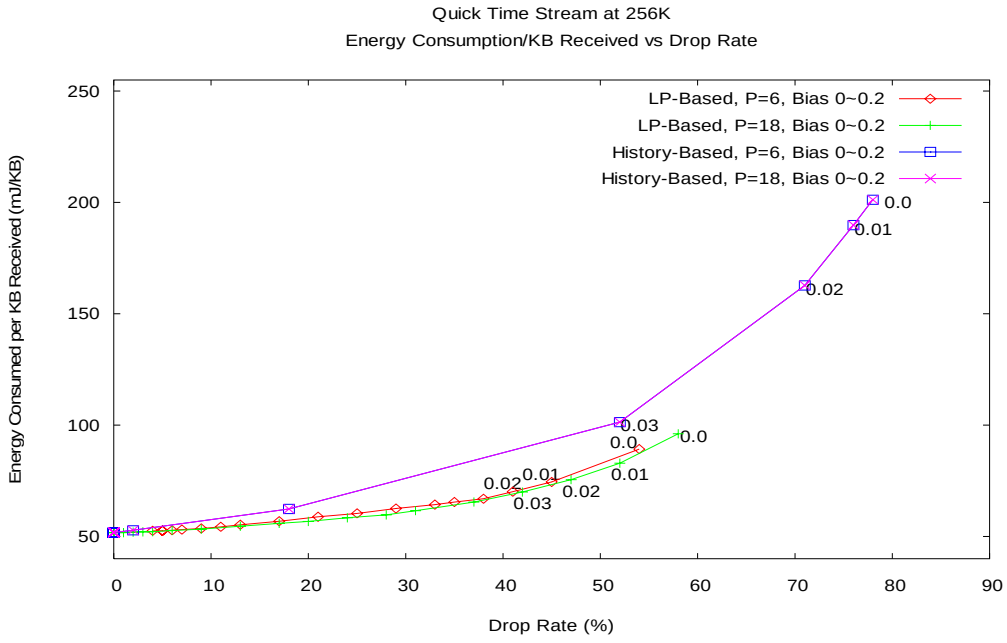


Figure 5.13 Energy Consumption per KByte of Data Received, Results of History- & Linear Prediction-based Approaches, Apple QuickTime Format at 256Kbps

5.5 **Conclusions**

The wireless network interface card (WNIC) of a mobile computing device accounts for a significant percentage of the overall client power consumption. In this paper, we have shown how linear prediction can be used to predict the length of the *sleep* time intervals for the client-side WNIC in order to reduce its energy consumption. The prediction model is trained using previously observed no-data intervals for a multimedia traffic stream. Experimental results show that, for a given value of additive (negative) bias, the statistical linear prediction-based approach yields, simultaneously, a lower data drop rate and a lower energy metric when compared to the history-based approach. In fact, the history-based approach can be looked upon as a special (i.e., degenerate) case of the linear prediction-based approach where all the predictor coefficients are identical in value.

Different popular multimedia streaming formats exhibit different data streaming characteristics. The Real and the Apple QuickTime media streams exhibit greater variation in the data packet sizes and inter-packet arrival times when compared to the Microsoft Media streams. This makes it hard for the prediction algorithm to reliably predict the lengths of no-data intervals. Nevertheless linear prediction-based approach is shown to be more robust than the history-based approach in its ability to predict the lengths of the no-data intervals, for all the three popular media stream formats explored in this work, namely Microsoft Media, Apple QuickTime and Real.

Future research will investigate more sophisticated time series modeling and prediction methods. Problems scenarios where both the server and the client are power constrained (such as in a peer-to-peer ad-hoc mobile network) will also be investigated.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 **Contributions**

At the outset of the dissertation, I outlined the research challenges that are considered important for mobile multimedia personalization. In the following paragraphs, I list the contributions this dissertation has made towards solving some of these problems in the video personalization domain. I first list the contributions to individual problems that this study has achieved. Then the overall contributions are summarized when the work of this dissertation is taken as a whole.

6.1.1 **A stochastic modeling approach to automatically segment and index video streams in a single pass**

Automatic semantics-based video segmentation and indexing is the first step towards client-center retrieval and personalization. Humans tend to use high-level features (concepts), such as keywords, text descriptors, to interpret images and measure their similarity, while the features automatically extracted using computer vision techniques are mostly low-level features (color, texture, shape, spatial layout, etc.). In general, there is no direct link between the high-level concepts and the low-level features. The proposed multi-level HMM-based approach can solve the multimedia annotation problem and bridge the gap between the high-level concepts and the low-level features. Furthermore, it has the following advantages compared to the existing approaches.

- **Video segmentation and video indexing are performed in a single pass.** This is extremely valuable when dealing with large amounts of video data to populate a video database.

- **No domain-dependent knowledge about the structure of video programs is used.** The probabilistic grammar used to define the video program is learned entirely from the training data. This allows the proposed approach to handle various kinds of videos in a modular and extensible manner without having to manually redefine the program model.
- **Semantic unit level HMMs are used to model video units with clear semantic meanings.** The proposed data-driven approach does not need to use HMMs to model video edit effects. This not only simplifies the collection and processing of training data, but also ensures that all video segments in the video database are labeled with concepts with clear semantic meanings in order to facilitate video retrieval based on semantic content.

6.1.2 A MMKP-based video personalization strategy to generate a customized response to a client's request

The objective of video personalization is to present a customized or personalized video summary that retains as much of the semantic content desired by the client as possible but within the resource constraints imposed by the client. In order to generate the personalized video summary, the client preference(s), the client usage environment and client-side resource constraints need to be considered.

Compared to the existing 0/1KP-based and the FKP-based video personalization strategies, the proposed MMKP-based video personalization strategy is shown to include more relevant information in its response to the client's request. The MMKP-based personalization strategy is also shown to satisfy multiple client-side resource constraints, in contrast to the 0/1KP-based and the FKP-based personalization strategies which can only satisfy a single client-side resource constraint at a time.

6.1.3 A proposed approach to use empirical rules to quantify the amount of information contained in the transcoded video summaries relative to the original video

In existing literatures, there is no quantitative technique proposed to measure the performance of video personalization strategies. This is because it is a complex task due to the inherent difficulty in quantifying the amount of information contained within the original video and due to the diverse nature of the various transcoded versions of the original video.

Although there are many factors that determine the information content of a video, it is reasonable to assume that the amount of information or detail contained within a video summary is related to its duration. In most cases, the amount of information contained within a video (including its transcoded versions) does not necessarily increase linearly with its duration. We propose to use empirical laws to quantify the relationship between the amounts of information contained in the transcoded videos relative to the original video. The *Zipf* function, *sigmoid* function and *Rayleigh* distribution are proposed as plausible mapping functions for quantifying the relationship between the amount of information in the transcoded video relative to the original video, and are shown to be suitable for different kinds of videos.

6.1.4 A multi-stage client request aggregation strategy to utilize server resources efficiently and to enhance client experience

When there are a large number of clients sending their requests to the server, the average client-experienced latency is long if every client request is processed individually. The proposed client request aggregation strategy clusters similar client requests together so that the number of requests sent to the server is reduced, along with the average client latency. The client requests are heterogeneous in multiple dimensions, i.e. they are different in their video content

preferences, and in client-side constraints. A multi-stage clustering strategy is proposed to group similar request together one dimension at a time.

6.1.5 A client-side energy-aware multimedia data streaming strategy to efficiently utilize client energy

A necessary criterion for the mass acceptance of mobile devices is acceptable battery life of these devices. Unfortunately, advances in hardware and software are not matched by a corresponding increase in battery life. The proposed client-side energy-aware multimedia streaming strategy predicts the time durations during which to suspend communication by switching the WNIC to a *sleep* state. Experimental results show that, for a given value of additive (negative) bias, the proposed approach yields, a lower data drop rate and a lower energy metric simultaneously.

6.1.6 The overall architecture of a client-centered multimedia personalization system

This dissertation proposes a framework of a multimedia personalization system for mobile clients. The system includes a video preprocessing subsystem, a multiple-level video transcoding and hierarchical video content representation subsystem, a client query and constraint decoding subsystem, a video personalization subsystem, and a client request aggregation subsystem. The framework is modular and extensible. New techniques for the individual tasks of mobile multimedia personalization can be introduced and incorporated into the corresponding subsystems in the framework without changing other parts and the overall architecture of the system.

6.2 Dissertation Conclusions

The results of the dissertation have demonstrated that the proposed framework of the multimedia personalization system for mobile clients is feasible. Every individual subsystem has the capacity

to include new techniques to accomplish subtasks of mobile multimedia personalization. Our multimedia personalization system has the following characteristics:

1. It saves time to build the visual content database.
2. It generates personalized video feedbacks which include more user preferred content.
3. It supports multiple client-side constraints simultaneously.
4. It efficiently utilizes server resources to serve multiple clients, and hence shortens the user-experienced latency.
5. It efficiently utilizes client battery energy for multimedia streaming.

As one of the dissertation achievements, the multi-level HMM-based video segmentation and indexing automates the process of building the visual database. Experimental results demonstrate that:

- Parsing a video file into semantic units enables video retrieval based on high-level semantic content.
- For each video semantic unit, the stochastic behavior of the sequence of feature emissions from the image frames can be modeled by a HMM.
- A universal left-to-right HMM topology with three Gaussian mixture components can model the stochastic behavior of the image feature sequence in a video semantic unit well.
- Video program grammar can be represented by an *n*-gram probabilistic language model.
- The data-driven maximum likelihood estimation of the *2*-gram program model from training data yields very good results.

Another achievement of the dissertation is the formulation of the Multiple-choice Multi-dimensional Knapsack Problem (MMKP)-based video personalization strategy. Compared to the

existing 0/1KP and FKP-based video personalization strategies, it is observed from the experimental results that

- when the beginning portions of a video segment contain more information than the rest of the video, the proposed MMKP-based approach yields a response with higher total relevance value compared to the existing FKP-based and 0/1KP-based approaches to video personalization.
- the MMKP-based video personalization strategy is capable of satisfying multiple client-side constraints simultaneously.
- although there are many factors that determine the information content of a video, it is reasonable to assume that the amount of information or detail contained within a video summary is related to its duration. This is especially true when each indexed video segment is summarized at multiple levels of abstraction using algorithms for content-based key frame selection and motion panorama computation.
- empirical laws such as the Zipf function, sigmoid function and Rayleigh distribution are plausible mapping functions for quantifying the relationship between the amount of information in the transcoded video and that in the original video, and are shown to be suitable for different kinds of videos.

In order to limit the client-experienced latency, it is necessary to perform client request aggregation on the server end. Experimental results show that

- the proposed client request aggregation strategy reduces the mean client-experienced latency without significant reduction in the average relevance of the delivered video content to the client’s request.

- k-means clustering tends to generate smaller clusters, and hence yields lower client-experienced difference in viewing time and preference, and higher total amount of video data.
- k-means is preferable when precision is important.

Furthermore, experimental results show that

- the wireless network interface card (WNIC) of a mobile computing device accounts for a significant percentage of the overall client power consumption.
- linear prediction can be used to predict the length of the *sleep* time intervals for the client-side WNIC in order to reduce its energy consumption.
- linear prediction-based approach is shown to be more robust than the history-based approach in its ability to predict the lengths of the no-data intervals, for all the three popular media stream formats explored in this work, namely Microsoft Media, Apple QuickTime and Real.

I believe that this dissertation will serve as a useful guidance for developing and extending mobile multimedia personalization systems and algorithms. It also provides a framework and testing bed for introducing new video indexing, annotation, video content representation, video personalization and delivery techniques.

6.3 **Future Research Directions and Extensions**

6.3.1 **Video Summarization and Content Representation**

In addition to the content-aware key frame selection and motion panorama techniques used to summarize video segments, it would be beneficial to investigate and incorporate further video content summarization and representation techniques into the system.

6.3.2 **Content-based Image Features**

Low level image features and Tamura features are used in the HMM-based video segmentation and indexing subsystem. There are a number of high-level semantic features of video segments that can be utilized in the process of video segmentation and indexing. People, object and semantic event detection can be used to improve the performance of video segmentation and indexing.

6.3.3 **HMM Model Selection**

Currently in the system, HMMs with a universal left-to-right topology with three Gaussian mixture components are used to model the stochastic behavior of image feature sequences of video semantic units. It would be beneficial to investigate the impact of model selection on the performance of video indexing. A data-driven model selection approach should be explored to automatically select the topology and the number of Gaussian mixture of a HMM.

6.3.4 ***N*-gram Language Model Investigation**

Investigation into the *n*-gram ($n > 2$) statistical language model to represent video program should be performed. For different categories of videos, the selection of *n*-gram language model needs to be validated.

6.3.5 **Empirical Mapping Function Validation**

For various categories of video programs and video content summarization strategies, it is necessary to validate the selection of the empirical mapping rules to quantify the relationship between the amount of information contained in the transcoded videos relative to the original video. Validation of empirical function parameters should also be preformed.

6.3.6 **Human Subject Evaluation of the Proposed MMKP-based Video Personalization**

In order to validate the proposed video personalization evaluation strategy, it would be beneficial to use human subjects to evaluate the generated video response to a client's request.

REFERENCES

Agrawal, P., Chen, J. C., Kishore, S., Ramanathan, P. and K. Sivalingam, Battery power sensitive video processing in wireless networks, *Proc. IEEE PIMRC98*, Boston, MA, pp. 116-120, 1998.

Aigrain, P., Zhang, H.J. and Petkovic, D., Content-based representation and retrieval of visual media: A state-of-the-art review, *Multimedia Tools Applicat.*, vol. 3, pp. 179–202, 1996.

Akbar, M.D., Manning, E.G., Shoja, G.C. and Khan, S., Heuristic Solutions for the Multiple-Choice Multi-dimension Knapsack Problem, Proc. Intl. Conf. Computational Science, pp. 659-668, 2001.

Akutsu, A., Tonomura, Y., Hashimoto, H. and Ohba, Y., Video indexing using motion vectors, *In SPIE Conference on Visual Communications and Image Processing*, Boston, MA, pp. 1522–1530, November 1992.

Ardebilian, M., Tu, X. and Chen, L., Robust 3d clue-based video segmentation for video indexing, *Journal of Visual Communication and Image Representation*, 11(1), pp.58–79, March 2000.

Ayars, J., Bulterman, D., Cohen, A., Day, K., Hodge, E., and Hoschka, P., Synchronized Multimedia Integration Language (SMIL 2.0) Specification, W3C Recommendation, URL: http://www.w3.org/TR/smil20/, Aug 2001.

Babaguchi, N., Toward abstracting sports video by highlights, *Proc. IEEE ICME*, pp. 1519–1522, 2000.

Babaguchi, N., Kawai, Y., Ogura, T. and Kitahashi, T., Personalized Abstraction of Broadcasted American Football Video by Highlight Selection, IEEE Trans. on Multimedia, Vol. 6, No.4, pp. 575-586, 2004.

Baeza-Yates, R., and Ribeiro-Neto, B., Modern Information Retrieval, ACM Press / Addison-Wesley, New York, 1999.

Bartoli, A., Dalal, N. and Horaud, R., Motion Panoramas, *Computer. Animation and Virtual Worlds*, Vol. 15, 501-517, 2004.

Baum, L.E., Peterie, T., Souled, G., and Weiss, N., A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains, *Ann. Math. Statist*., pp. 164-171, 1970.

Bhandarkar, S.M., Warke, Y.S., Khombhadia, A.A., Integrated Parsing of Compressed Video, *Lecture Notes In Computer Science*, Vol. 1614 , pp. 269 - 276, 1999.

Boll, S., Image and Video Retrieval from a User-Centered Mobile Multimedia Perspective, W.-K. Leow et al. (Eds.): *CIVR 2005, LNCS 3568*, pp. 18–27, Springer-Verlag Berlin Heidelberg 2005.

Boll, S., Klas, W. and Wandel, J., A Cross-Media Adaptation Strategy for Multimedia Presentations, *Proc. of the ACM Multimedia Conf. '99*, Orlando, Florida, USA, pp. 37–46, November 1999.

Bommaiah, E., Guo, K., Hofmann, M. and Paul, S., Design and Implementation of a Caching System for Streaming Media Over the Internet, *IEEE Real Time Technology and Applications Symposium(RATS)* , pp. 111-121, 2000.

Boreczky, J.S. and Wilcox. L.D., A Hidden Markov Model Framework for Video Segmentation Using Audio an Image Features, *Proc. IEEE ICASSP*, Seattle, May 1998.

Bradshaw1, M., Wang, B., Subhabrata Sen, S., Gao, L., Kurose1, J., Shenoy, P. and Towsley, D., Periodic broadcast and patching services – implementation, measurement and analysis in an internet streaming video testbed, *Multimedia Systems* 9, pp. 78–93, 2003.

Brown, P., deSouza, P. and Mercer, R., Class-Based n-gram Models of Natural Language, *Computational Linguistics Archive*, Vol 18 , Issue 4, pp 467 - 479, 1992.

Brusilovsky, P., Kobsa, A. and Vassileva, J., *Adaptive Hypertext and Hypermedia* Kluwer Acad. Publ., Dordrecht, 1998.

Bunningen, A., Context aware querying - Challenges for data management in ambient intelligence, *Technical Report TR-CTIT-04-51*, University of Twente, 2004.

Chaisorn, L., Chua, T.S., Koh, C.K., Zhao, Y., Xu, H., Feng, H. and Tian, Q., A Two-Level Multi-Modal Approach for Story Segmentation of Large News Video Corpus, in Proceedings of the Twelve Text REtrieval Conference (TREC-12)', Gaithersburg, MD, 2003.

Chandra, S., Wireless network interface energy consumption Implications for popular streaming formats, *Multimedia Systems*, Springer-Verlag, pp. 185-201, 2003.

Chang, S.F. and Sundaram, H., Structural and semantic analysis of video, *Proc. IEEE ICME*, pp. 687-690, 2000.

Chen, M.J., Chu, M.C., and Pan, C.W., Efficient Motion Estimation Algorithm for Reduced Frame-rate Video Transcoder, *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 12, No. 4, 269–275, 2002.

Chiasserini, C. F., and Rao, R. R., Pulsed battery discharge in communication devices, *Proc. 5th ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '99)*, Seattle, WA, pp. 88-95, August 1999.

Dailianas, A., Allen, R.B. and England, P., Comparison of automatic video segmentation algorithms, SPIE *Conference on Integration Issues in Large Commercial Media Delivery Systems*, volume 2615, pp. 2–16, Philadelphia, PA, October 1995.

Datta, A., Celik, A., Kim, J., VanderMeer, D.E. and Kumar, V., Adaptive broadcast protocols to support power conservant retrieval by mobile users, *Proc. Data Engineering Conf. (ICDE)*, pp. 124–133, 1997.

Davis, M., King, S., Good, N. and Sarvas, R., From Context to Content: Leveraging Context to Infer Media Metadata, *Proc. of the ACM Multimedia (MM2004)*, pp. 188-195, 2004.

Douglis, F., Krishnan, P., and Bershad, B., Adaptive disk spin down policies for mobile computers, *Proc. 2nd USENIX Symposium on Mobile and Location Independent Computing*, Monterey, CA, pp. 121-137, 1995.

Doulamis, N., Doulamis, A., Avrithis, Y., Ntalianis, K., and Kollias, S., Efficient Summarization of Stereoscopic Video Sequences, *IEEE Trans Circuits and Systems for Video Technology*, Vol. 10, No. 4, pp. 501–517, 2000.

Eickeler, S., and Müller, S., Content-based Video Indexing of TV Broadcast News using Hidden Markov Models. *Proc. of ICASSP*, pp. 2997-3000, March 1999.

Eickeler, S., and Rigoll, G., A Novel Error Measure for the Evaluation of Video Indexing Systems, *Proc. IEEE Intl. Conf. Acoustics, Speech and Signal Processing*, Istanbul, Turkey, Vol. 4, 1991-1994, 2000.

Eleftheriadis, A., and Batra, P., Dynamic Rate Shaping of Compressed Digital Video, *IEEE Transactions on Multimedia*, Vol. 8, No. 2, pp. 297 – 314, 2006.

Everitt, B.S., Landau, S. and Leese, M., Cluster Analysis, 4th edn. Arnold, London, 2001.

Faber, V., Clustering and the continuous k-means algorithm. Los Alamos Science 22, pp. 138–144, 1994.

Feeney, L.M. and Nilsson, M., Investigating the energy consumption of a wireless network interface in an ad hoc networking environment, *Proc. IEEE INFOCOM 2001*, Anchorage, Alaska, vol. 3: pp. 1548–1557, 2001.

Fellbaum, C. (Ed.), *WordNet – An Electronic Lexical Database*, The MIT Press, Cambridge, MA, 1998.

Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Gorkani, M., Hafner, J., Lee, D., Petkovic, D., Steele, D., and Yanker, P. 1995, Query by Image and Video Content: The QBIC System, *IEEE Computer Magazine*, pp. 23 – 32, September, 1995.

Forney, G.D., The Viterbi Algorithm, *Proceedings of the IEEE*, Vol. 61, No. 3, 268-278, 1973.

Gargi, U. and Kasturi, R., An evaluation of color histogram-based methods in video indexing, *International Workshop on Image Databases and Multimedia Search*, Amsterdam, Netherlands, pp. 75-82, August 1996.

Gong, Y. and Liu, X., Video shot segmentation and classification, *In IAPR International Conference on Pattern Recognition*, Barcelona, Spain, pp. 860–863, September 2000.

Govil, K., Chan, E. and Wasserman, H., Comparing algorithms for dynamic speed-setting of a low-power CPU, *Proc. 1st ACM International Conference on Mobile Computing and Networking (MOBICOM95)*, pp. 13–25, 1995.

Hamilton, J. D., *Time Series Analysis*, Princeton University Press, Princeton, NJ, 1994.

Havinga, P. J.M., *Mobile Multimedia Systems*, Ph.D. thesis, University of Twente, 2000.

Helmbold, D. P., Long, D. E., and Sherrod, B., A dynamic disk spin-down technique for mobile computing, *Proc. 2nd ACM International Conference on Mobile Computing (MOBICOM96)*, pp. 130–142, 1996.

Hernandez, R.P., and Nikitas, N.J., A New Heuristic for Solving the Multichoice Multidimensional Knapsack Problem, IEEE Trans. System, Man and Cybernetics, Part A, Vol. 35, No. 5, pp. 708-717, 2005.

Hua, K. and Sheu, S., Skyscraper broadcasting: A new broadcasting scheme for metropolitan video-on-demand systems. Proc. ACM SIGCOMM, Cannes, France, pp. 89-100, 1997.

Huang, J., Liu, Z., and Wang, Y., Joint Scene Classification and Segmentation Based on Hidden Markov Model, *Proc. of ICME*, New York, NY, pp. 538-550, August 2000.

Huang, J., Liu, Z., and Wang, Y., Joint Scene Classification and Segmentation Based on Hidden Markov Model, *IEEE Trans. Multimedia*, Vol. 7, No. 3, 538-550, 2005.

Huang, J., Liu, Z., Wang, Y., Chen, Y. and Wong, E.K., Integration of multimodal features for video scene classification based on HMM, *IEEE Workshop on Multimedia Signal Processing*, Copenhagen, Denmark, pp. 53-58, 1999.

Imielinski, I., Gupta, M., and Peyyeti, S., Energy efficient data filtering and communications in mobile wireless computing, *Proc. Usenix Symposium on Location Dependent Computing*, pp. 109-119, 1995.

Jaimes, A., and Chang, S.F., A Conceptual Framework for Indexing Visual Information at Multiple Levels. *IS&T/SPIE Internet Imaging*, Vol. 3964, San Jose, CA, pp. 2-15, Jan. 2000.

Jain, A.K., Murty, M.N. and Flynn, P.J., Data Clustering: A Review, ACM Computing Surveys, Vol. 31, No. 3, pp.264-323, September 1999.

Jasinschi, R., Dimitrova, N., McGee, T., Agnihotri, L. and Zimmerman, J., Video scouting: An architecture and system for the integration of multimedia information in personal TV applications, *Proc. IEEE ICASSP*, pp. 1405–1408, 2001.

Kasturi, R. and Jain, R.c., Dynamic vision, *Computer Vision: Principles*, IEEE Computer Society Press, Washington, pp. 469–480, 1991.

Kaufman, L. and Rousseeuw, P., *Finding Groups in Data: An Introduction to Cluster Analysis*, New York, John Wiley & Sons, 1990.

Kawa J, Pietka E, Image clustering with median and myriad spatial constraint enhanced FCM, Proc. Computer Recognition Systems, CORES 2005, Springer, pp. 211 – 218, 2005.

Khan, S., Quality adaptation in a multi-session adaptive multimedia system: model and architecture. PhD Thesis, Department of Electronical and Computer Engineering, University of Victoria, 1998.

Kim, C., and Hwang, J., An Integrated Scheme for Object-based Video Abstraction. *Proc. ACM Conf. Multimedia,* Los Angeles, CA, 2000, pp. 303–311, 2000.

Kopf, S., King, T., Fleming, L. and Effelsberg, W., Color Adaptatiok of Videos for Mobile Devices, *Proc. of the ACM Multimedia*, Santa Barbara, CA, pp. 963-964, Obtober 23-27, 2006.

Korenius, T., Juhola, M. and Laurikkala, J., On applying the principal components analysis and cosine similarity for information retrieval, Information Processing & Management, Submitted, Available from the authors by a request.

Kravets, R., and Krishnan, P., Power management techniques for mobile communication, *Proc. 4th International Conf. on Mobile Computing and Networking (MOBICOM98),* pp. 157–168, 1998.

Lahti, J., Westermann, U. and Palola, M., MobiCon–Integrated Capture, Annotation, and Sharing of Video Clips with Mobile Phones, *Proc. of the ACM Multimedia Modeling (MM'05),* pp.798-799, Singapore, Nov 2005.

Leacock, C., and Chodorow, M., Combining Local Context and WordNet Similarity for Word Sense Identification, *WordNet: An Electronic Lexical Database*, C. Fellbaum (Editor), MIT Press, Cambridge, MA, pp. 265-283, 1998.

Lee, S.M., Xin, J.H. and Westland, S., Evaluation of Image Similarity by Histogram Intersection, *COLOR research and application*, Vol. 30pp.265-274, , Number 4, August 2005.

Lee, M.S., Yang, Y.M. and Lee, S.W., Automatic video parsing using shot boundary detection and camera operation analysis, *Pattern Recognition*, 34(3), pp. 711–719, March 2001.

Lemlouma, T. and Layaïda, N., Adapted Content Delivery for Different Contexts, *Proc. of the. Symposium on Applications and the Internet (SAINT'03)*, Orlando, Florida, USA, January 27-31, 2003, pp. 190-197.

Li, B., and Sezan, M.I., Event Detection and Summarization in Sports Video. *Proc. of CBIVL*, No. 8, pp. 132–138, 2001.

Li, K., Kumpf, R., Horton, P. and Anderson, T. , A quantitative analysis of disk drive power management in portable computers, *Proc. USENIX Association Winter Technical Conf.*, pp. 279–291, 1994.

Lienhart, r., Pfeiffer, S. and Effelsberg, W., Video abstracting, *Commun. ACM*, vol. 40, no. 12, pp. 55–62, 1997.

Lin, C.Y., Tseng, B.L. and Smith, J.R., VideoAnnEx: IBM MPEG-7 Annotation Tool, *IEEE Intl. Conf. on Multimedia & Expo (ICME)*, Baltimore, July 2003.

Lorch, J. and Smith, A.J., Software strategies for portable computer energy management, *IEEE Personal Communications Magazine*, pp. 60–73, June, 1998.

Malioutov, I., and Barzilay, R., Minimum Cut Model for Spoken Lecture Segmentation, *Proc. of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*, Sydney, Australia, pp. 25-32, July 2006.

Manning, C.D. and Sch¨utze, H., Foundations *of Statistical Natural Language Processing*, The MIT Press, Cambridge, USA, 1999.

Merialdo, B., Lee, K.T., Luparello, D., and Roudaire, J., Automatic Construction of Personalized TV News Programs. *Proc. ACM Conf. Multimedia*, Orlando, FL, pp. 323—331, Sept. 1999.

Mohan, A., Papageorgiou, C. and Poggio, T., Example-based object detection in images by components, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 23, No. 4, pp. 349–361, 2001.

Mohan, R. Smith, J. and Li. C., Adapting multimedia internet content for universal access, *IEEE Trans. on Multimedia*, volume 1(1), March 1999, pages 104–114.

Nagasaka, A. and Tanaka, Y., Automatic Video Indexing and Full-video Search for Object Appearances, *IFIP Working Conference on Visual Database Systems*, Budapest, Hungary, pp. 113-127, October 1991.

Nakajima, Y., Hori, H., AND Kanoh, T., Rate Conversion of MPEG Coded Video by Requantization Process, *Proc. IEEE Intl. Conf. Image Processing*, Washington, DC, pp. 408-411, 1995.

Nam, J., Alghoniemy, M. and Tewfik, A.H., Audio-visual content-based violent scene characterization, Proc. *Of the IEEE International Conference on Image Processing*, Chicago, USA, Vol. 1, pp. 353–357, 1998.

Naphade, M.R. and Huang, T.S., Extracting semantics from audio-visual content: the final frontier in multimedia retrieval, *IEEE Trans. Neural Networks*, 13(4), pp. 793- 810, 2002.

Ney, H., and Ortmanns, S., Progress on Dynamic Programming Search for Continuous Speech Recognition. *IEEE Signal Processing Magazine*, pp. 64-83, 1999.

Nguyen, H.T., Worring, M. and Dev, A., Detection of moving objects in video using a robust motion similarity measure, *IEEE Transactions on Image Processing*, Vol. 9, No. 1, pp. 137–141, 2000.

Oh, J.U. and Hua, K.A., An efficient technique for summarizing videos using visual contents, *Proc. IEEE ICME*, pp. 1167 – 1170, 2000.

Papoulis, A., *Probability, Random Variables, and Stochastic Processes*, 2nd ed, McGraw-Hill, New York, NY, pp. 104 and 148, 1984.

Pham, T.V. and Worring, M., Face detection methods: A critical evaluation, Technical Report 2000-11, Intelligent Sensory Information Systems, University of Amsterdam, 2000.

Porter, S.V., Mirmehdi, M. and Thomas, B.T., Video cut detection using frequency domain correlation, *IAPR International Conference on Pattern Recognition*, vol. 3, pages 413–416, Barcelona, Spain, September 2000.

Pye, D., Hollinghurst, J., Mills, J. and Wood, R., Audio-visual segmentation for content-based retrieval, *International Conference on Spoken Language Processing*, Sydney, Australia, December 1998.

Rabiner, L.R., A tutorial on hidden markov models and selected applications in speech recognition, *Proc. of the IEEE*, Vol. 77, No. 2, pp. 257–286, 1989.

Rabiner, L. and Juang, B. H., *Fundamentals of Speech Recognition*, Prentice Hall, Englewood Cliffs, NJ, USA, 1993.

Rowley, H.A., Baluja, S. and Kanade, T., Neural network-based face detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 1, pp. 23–38, 1998.

Salton, G., Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer, Addison-Wesley, Reading, Massachusetts, 1989.

Sen, S., Gao, L., Rexford, J. and Towsley D., Optimal patching schemes for efficient multimedia streaming, Proc. 9 th Int'l Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'99), Basking Ridge, NJ, pp. 455-463, June 1999.

Schneiderman H. and Kanade, T., A statistical method for 3D object detection applied to faces and cars, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Hilton Head, USA, pp. 1746-1751, 2000.

Sethi, I. and Coman, I., Mining association rules between low-level image features and high-level concepts, *Proc. of the SPIE Data Mining and Knowledge Discovery*, vol. III, 2001, pp. 279–290.

Shih, E., Bahl, V. and Sinclair, M., Reducing energy consumption of wireless mobile devices using a secondary low-power channel, *MIT Laboratory for Computer Science*, pp. 37-38, 2003.

Shinoda, K., Bach, N.H., Furui, S. and Kawai, N., Scene recognition using Hidden Markov Models for video database, *Proc. Symposium on Large-Scale Knowledge Resources(LKR2005)*, Tokyo, Japan, pp.107-110, 2005.

Singh, S., Woo, M. and Raghavendra, C.S., Power-aware routing in mobile ad hoc networks, *Proc. 4th ACM/IEEE International Conference on Mobile Computing and Networking*, pp. 181–190, 1998.

Smith, M.A. and Kanade, T., Video skimming and characterization through the combination of image and language understanding techniques, *Proc. IEEE CVPR*, pp. 775–781, 1997.

Smyth, B. and Cotter, P., A personalized television listings service, *Commun. ACM*, vol. 43, no. 8, pp. 107–111, 2000.

Snoek, C.G.M., and WORRING, M., Time Interval Maximum Entropy Based Event Indexing in Soccer Video, *Proc. IEEE Int. Conf. Multimedia & Expo*, Vol. 3, Baltimore, USA, 481–484, 2003.

Stemm, M., Gauthier, P., Harada, D. and Katz, R. H., Reducing power consumption of network interface in hand-held devices, *Proc. 3rd International Workshop on Mobile Multimedia Communications (MoMuc-3)*, Princeton, NJ, pp. 103-112, September 1996.

Sun, H., Kwok, W., and Zdepski, J., Architectures for MPEG Compressed Bitstream Scaling, *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 6, pp. 191-199, 1996.

Swanberg, D., Shu, C.F. and Jain, R., Knowledge guided parsing and retrieval in video databases, *SPIE Conference on Storage and Retrieval for Image and Video Databases*, volume 1908, San Jose, CA, pp. 13–24, February 1993.

Tamura, H., Mori, S. and Yamawaki, T., Textural features corresponding to visual perception, *IEEE Trans on Systems, Man and Cybernetics*, vol. 8, pp. 460-472, 1978.

Taniguchi, Y., Akutsu, A. and Tonomura, Y., Panoramaexcerpts: Extracting and Packing Panoramas for Video Browsing, *ACM International Conference on Multimedia*, pp. 427-436, Seattle, WA, November 1997

Toklu, C., Liou, S.P. and Das, M., VIDEOABSTRACT: A hybrid approach to generate semantically meaningful video summaries, *Proc. IEEE ICME*, pp. 1333 – 1336, 2000.

Tonomura, Y. and Abe, S., Content Oriented Visual Interface using Video Icons for Visual Database Systems, *Journal of Visual Languages and Computing*, 1(2), pp. 183-198, June 1990.

Tseng, B.L., Lin, C.Y., and Smith, J.R., Video Summarization and Personalization for Pervasive Mobile Devices, *SPIE Electronic Imaging 2002 - Storage and Retrieval for Media Databases*, San Jose, January 2002.

Tseng, B.L., Lin, C.Y., and Smith, J.R., Video Personalization and Summarization System for Usage Environment, *Jour. Visual Communication and Image Representation*, Vol. 15, pp. 370–392, 2004.

Tseng, B.L., Lin, C.Y., and Smith, J.R., Using MPEG-7 and MPEG-21 for Personalizing Video, *IEEE Multimedia*, Vol. 11(1), pp. 42-53, 2004.

Tseng, B.L. and Smith, J.R., Hierarchical Video Summarization Based on Context Clustering, *Proc. SPIE*, Vol. 5242, pp. 14-25, Nov. 2003.

Uykan, Z. and Koivo, H.N., Unsupervised Learning of Sigmoid Perceptron, *Proc. IEEE Intl. Conf. Acoustics, Speech and Signal Processing*, Vol.6, Istanbul, Turkey, pp. 3486 – 3489, June, 2000.

van der Laan, M., Pollard, K., and Bryan, J. A new partitioning around medoids algorithm, *Technical Report 105, Group in Biostatistics*, University of California, February 2002.

Vanderbei, R.J., *Linear Programming: Foundations and Extensions*, Kluwer Academic Publishers, 1997.

Vasconcelos, N., Lippman, A., A multiresolution manifold distance for invariant image similarity, *IEEE Trans.on Multimedia* 7 (1), 2005, pp. 127–142.

Venkatesh, D. and Little, T. D. C., Dynamic Service Aggregation for Efficient Use of Resources in Interactive Video Delivery, *Lecture Notes in Computer Science*, Vol. 1018, Springer-Verlag, pp. 113-116, Nov. 1995.

Viola, P. and Jones, M.J., Robust real-time face detection, *International Journal of Computer Vision*, 57(2), pp. 137-154, May 2004.

Viterbi, A.J., Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, *IEEE Transactions on Information Theory* 13(2), pp. 260–269, April 1967.

Wactlar, H.D., Kanade, T., Smith, M.A. and Stevens, S.M., Intelligent Access to Digital Video: Informedia Project, *IEEE Computer*, pp. 46-51, May 1996.

Wei, Y., Bhandarkar, S.M. and Li, K., Client-Centered Multimedia Content Adaptation, ACM Trans. on Transactions on Multimedia Computing, Communications and Application (TOMCCAP), submitted.

Weiser, M., Welch, B., Demers, A. and Shenker, S., Scheduling for reduced CPU energy, *Proc. 1st Symposium on Operating Systems Design and Implementation (OSDI)*, Monterey, CA, pp. 13-23, 1994.

Wheeler, E.S., Zipf's Law and Why it Works Everywhere, *Glottometrics*, vol.4, pp. 45-48, 2002.

Wilkes, J., Predictive Power Conservation, *Technical Report HPL-CSP-92-5*, Hewlett-Packard Labs, 1992.

Willett, P., Recent trends in hierarchic document clustering: A critical review, *Information Processing & Management*, 24(5), pp.577-597, 1988.

Wu, S., Jin, H., Chu, J. and Fan, K., A Novel Cache Scheme for Cluster-based Streaming Proxy Server, *25th IEEE Intl. Conf. Distributed Computing Systems Workshops*, pp. 727 – 733, June 2005.

Yang, M.H., Kriegman, D. and Ahuja, N., Detecting faces in images: A survey, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 1, pp. 34–58, 2002.

Yeung, M.M. and Yeo, B.L., Video visualization for compact presentation and fast browsing of pictorial content, *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 5, pp. 771–785, Oct. 1997.

Yilmaz, A. and Shah, M., Shot detection using principle coordinate system, *In IASTED International Conference on Internet and Multimedia Systems and Applications*, Las Vegas, CA, November 2000.

Yu, C. and Meng W., *Principles of Database Query Processing for Advanced Applications*, Data Management Systems, Morgan Kaufmann, 1998.

Zabih, R., Miller, J. and Mai, K., A Feature-based Algorithm for Detecting and Classifying Production Effects, *Multimedia Systems*, 7(2), pp. 119–128, March 1999.

Zhu, W., Yang, K., and Beacken, M., CIF-to-QCIF Video Bitstream Down Conversion in the DCT Domain, *Bell Labs Technical Journal*, Vol. 3, No. 3, 21-29, 1998.