Advances in Multimedia

Collaboration and Optimization for Multimedia Communications

Guest Editors: Jianwei Huang, Zhu Li, and Qian Zhang

Collaboration and Optimization for Multimedia Communications

Advances in Multimedia

Collaboration and Optimization for Multimedia Communications

Guest Editors: Jianwei Huang, Zhu Li, and Qian Zhang

Contents

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2008, Article ID 720685, 2 pages
doi:10.1155/2008/720685

Editorial

Collaboration and Optimization for Multimedia Communications

Jianwei Huang,[1] Zhu Li,[2] and Qian Zhang[3]

[1] *Department of Information Engineering, The Chinese University of Hong Kong, New Territory, Hong Kong SAR, China*
[2] *Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong SAR, China*
[3] *Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Kowloon, Hong Kong SAR, China*

Correspondence should be addressed to Jianwei Huang, jwhuang@ie.cuhk.edu.hk

Received 27 February 2008; Accepted 27 February 2008

The advances of Internet and wireless access technologies have opened up new opportunities to serve high quality, on-demand multimedia applications. Applications like mobile TV, IPTV, on-demand streaming, and peer-to-peer video sharing have fundamentally changed the content distribution landscape, and have been accelerating a social and engineering revolution in media distribution and consumption. To achieve the ultimate goals of total freedom in self-expression, seamless mobile access, and anytime anywhere media consumption, technology advances in various areas need to be reexamined and jointly utilized under a coherent optimization framework to reach an efficient end-to-end media delivery solution. New models, metrics, and methodologies in source, and channel coding, distributed and collaborative communications are needed to intelligently adapt the multimedia content to suit user preferences, meet device and network constraints, and achieve better communication resource utilization. The source coding and adaptation decisions of media sources need to be reconciled with the limited network resources, end-user preferences, and resource allocation schemes at network nodes. Distributed optimization schemes like pricing and game theoretical approaches are needed to improve resource allocation and management efficiency. In this special issue on multimedia networking, we present several papers that address such issues.

The first paper of this special issue, "A collaborative wireless access to on-demand services" by Z. Naor, presents a collaborative access scheme that exploits the broadcast nature of the wireless communications in order to achieve better multicast content delivery. The proposed method is particularly suitable for sessions of long time durations, for applications where clients can subscribe to ahead of time, and for applications in which the clients receive the same information simultaneously.

The second paper of this special issue, "A stream tapping protocol involving clients in the distributions of videos on demand" by S. Kulkarni et al., presents a stream tapping protocol that involves clients in the video distribution process. Compared with the traditional taping protocol, the proposed one greatly reduces the workload of the video server by delegating part of the content distribution process to the clients who are watching the video.

The third paper of this special issue, "Automatic bandwidth adjustment for content distribution in MPLS networks" by D. Moltchanov, discusses a new algorithm for dynamic resource adaptation to temporarily changing traffic conditions in multiprotocol label switching (MPLS) networks. The major advantage of the proposed approach is that it is fully autonomous, takes into account statistical characteristics of traffic patterns, and is independent of the choice of the sampling interval of MPLS automatic bandwidth adjustment capability.

The fourth paper of this special issue, "Rate-distortion optimized frame dropping for multiuser streaming and conversational video" by W. Tu et al., considers rate-distortion optimized strategies for dropping frames from multiple conversational and streaming videos sharing limited network node resources. Experimental results show that a significant improvement in end-to-end performance is achieved compared to priority-based random early dropping schemes.

The fifth paper of this special issue, "A theoretical framework for quality-aware cross-layer optimized wireless multimedia communications" by S. Ci et al., presents a theoretical framework for integrated cross-layer control and optimization in wireless multimedia communications. The framework includes two essential parts: a delay-distortion-driven optimization framework and a new approximate dynamic programming technique based on significance measure and sensitivity analysis for high-dimensional nonlinear cross-layer optimization.

The sixth paper of this special issue, "Optimal multilayer adaptation of SVC video over heterogeneous environments" by T. Thang et al., proposes an optimized framework of controlling the SNR scalability across multiple spatial layers in scalable video coding format. The proposed framework has the flexibility in allocating the resource (i.e., bitrate) among spatial layers, where the overall quality is defined as a function of all spatial layers' qualities and can be modified on the fly.

ACKNOWLEDGMENTS

We thank all the authors for their contributions to the special issues and wish the readers a pleasant reading. We would also like to thank the reviewers for taking time to complete the reviews promptly.

Jianwei Huang
Zhu Li
Qian Zhang

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2008, Article ID 273187, 9 pages
doi:10.1155/2008/273187

Research Article

A Collaborative Wireless Access to On-Demand Services

Zohar Naor

Department of Mathematics Physics and Computer Science, Faculty of Science and Science Education, University of Haifa at Oranim Academic College of Education, 31905 Haifa, Israel

Correspondence should be addressed to zohar Naor, zohar@math.haifa.ac.il

Received 28 April 2007; Revised 1 September 2007; Accepted 12 November 2007

Recommended by Jianwei Huang

A collaborative access scheme that exploits the broadcast nature of the wireless communication in order to achieve multicast content delivery is presented in this paper. The key idea is that individual clients requesting for the same content can collaborate and share the same data channel. As opposed to broadcasting, this method enables the clients to determine online the delivered content, and thus supports on-demand services. On the other hand, a multicast content delivery is much more efficient than a unicast content distribution, which must use a dedicated data channel per each and every client. This method is particularly suitable for sessions having a long-time duration, for applications in which clients can subscribe to ahead of time, and for applications in which the clients receive the same information simultaneously. A multicast content distribution increases the network service throughput in terms of the expected number of clients served simultaneously, and therefore it offers a reduced waiting time for content delivery at highly loaded time periods. It is shown that the problem of maximizing the efficiency of distributing a content in a wireless network is NP-hard. An approximation algorithm is therefore used, that for any $0 < \varepsilon < 1$ finds an approximation solution with a relative accuracy ε. The proposed method does not require any hardware modification on the network equipment. Thus, it can be easily implemented.

1. INTRODUCTION

The concept of collaborative wireless networking is emerging as a promising technology to enhance system performance by sharing wireless resources among the demanding clients. For instance, sharing antennas to enable a virtual multiple-input multiple-output (MIMO) is an example for a collaborative scheme. The focus in this study is to use an application layer collaborative method in order to establish a virtual MIMO system. Video applications, such as near video-on-demand (NVOD) service, are characterized by sessions having a relatively long-time duration, and a relatively high arrival rate of clients requesting for the same data. Consequently, even a moderate demand for video applications may potentially block the wireless channel, unless a collaborative scheme is used, that enables bandwidth sharing among different clients. The traditional content distribution methods were originally designed for wired networks. As such, they do not consider the wireless channel conditions. For existing wireless networks, even a moderate demand for video streaming can easily exceed the system available bandwidth. For this reason, the approach adopted by manufactures of wireless communication equipment for advanced content distribution over wireless networks is to build dedicated broadcast networks for what is known as "mobile TV." These networks are expected to use a new infrastructure, based on either *DVB-H* [1] or *MediaFlo* [2] standards. Unfortunately, these schemes do not support on-demand services. Whereas unicast multimedia streaming is used for mobile TV today, broadcast extensions to mobile networks like 3GPP *MBMS* and 3GPP2 *BCMCS* are under standardization. The contribution of this paper is a formulation of the content distribution problem as an optimization problem that is proved to be NP-hard, and the development of an efficient, yet reliable scheme that enables an on-demand content distribution.

Traditionally, there are three representative approaches for content distribution: media servers replication, using existing proxies for media data buffering, and constructing peer-to-peer overlay networks, dedicated to content distribution. These schemes were designed mostly for wired

networks and do not consider the special characteristics of the wireless channel.

Building mirror sites is a very efficient solution. Unfortunately, since the mirror sites are core nodes, they are not aware of the conditions on the wireless channel. Moreover, multicast-based content delivery is very hard to be implemented by core nodes. For this reason, no large-scale commercial service supports multicast content distribution at the Internet backbone.

The proxy-based approach for content distribution is to use the memory of existing proxies for caching media data. The main drawback of this approach is that due to the huge memory size of a typical video file, the proxy memory cannot support a scalable service, and there is a need to use special machines, as it is done by cable TV companies. In order to overcome this problem, segment-based proxy caching strategies were proposed [3], to cache partial segments of the media objects instead of their entirety. Yet, the memory limitation still restricts the service scalability, unless a dedicated spacial machine is used. Most importantly, the proxy is not aware of the variable conditions of the wireless channel.

The idea to construct a peer-to-peer overlay network, dedicated to content distribution, is very popular across the Internet [4]. The key idea of this method is that each node in the overlay network forwards the data it is receiving, and serves other nodes as a server. The usage of an overlay peer-to-peer network that uses a proxy system supported by peer-to-peer networks was proposed in [5]. This approach is very effective when applied in wired networks. Its main drawback is that when nodes are leaving the group, the nodes that remain in the overlay network may suffer from service disconnection. Recovery algorithms [6] were proposed to recover from this situation. Unfortunately, the dynamics of group membership in a wireless network is much greater than that experienced in a wired network, due to the client mobility. Moreover, a wireless channel is severely vulnerable to phenomenons such as multipath fading, interference, base station reselection, and so forth. The outcome of these phenomenons is that a wireless channel may suffer from high and variable round trip time, rate fluctuations, link outage, and occasional burst losses. For these reasons, the approach of peer-to-peer overlay network used in the Internet is not recommended for a wireless network, due to the mobility of the participating peers and the instability of the peer-to-peer connection for a long-time period.

The studies cited above focus on the wired network, and do not consider the special characteristics of the wireless channel. Hence, there is a need to develop a strategy that can cope with the wireless channel conditions in a more efficient way, that exploits the broadcast nature of the wireless communication. It would be desirable to use this property to utilize the scarce wireless bandwidth more efficiently.

1.1. The contribution of this work

The main contribution of this work is the usage of the access point to the wireless network as a proxy server. Thus, independent clients requesting for the same content can collaborate, and share the same data channel. Traditionally, the access point to the wireless network (e.g., a base station in a cellular network) is used only for transmission. The suggested method requires a cross-layer architecture, aimed to reduce redundant traffic and thus to increase the network capacity, in terms of number of clients served simultaneously. Such an architecture can be applied on a metropolitan WiMAX network, in which the coverage area is sufficiently large such that many clients are expected to receive the same video stream simultaneously. The key idea is to exploit the broadcast nature of the wireless communication in order to create a mechanism that enables to transmit data requested by many users residing in the same area only once. This mechanism is achieved by an addressing method that labels each request for data, using the application type, the encoding method, and the data required as the keys. Consequently, independent requests for the same data initiated from the same area are accumulated in order to form a virtual multicast group. A Virtual Multicast Group (VMG) is an ad hoc multicast group, that contains all the users residing in the same area and waiting for the same data at the same encoding method. Since the task of forming VMGs is very hard to be implemented by core nodes, it is performed at the access portion, that is, by the wireless network. For this reason, this method can be integrated with previously published methods for content distribution [3–5].

Content distribution can be potentially performed either by the content provider, or by the clients themselves [4, 5], or by an intermediate node. The first two methods were originally designed for wired networks. As such, they do not consider the wireless channel conditions. They are vulnerable to phenomenons such as high and variable round trip time, rate fluctuations, occasional burst losses, and instability of the client-network physical connection for a long-time period. Hence, these methods are not optimized for a high quality of service content distribution over wireless networks. Most importantly, whenever the bandwidth required to satisfy the content demands directed to different content providers exceeds the available system bandwidth, only the wireless network can select which content is delivered first. The VMG method is based on an application-layer module, to be installed in an intermediate node, located at the gateway between the wireless network and the wired backbone. This method is especially suitable for applications that are either delay insensitive, or that the clients receive the information simultaneously. Examples for such applications are live video streaming, NVOD service, and download requests of popular video files. Due to the long-time duration of a typical video session, and its wide bandwidth, even a moderate demand for such applications can easily exceed the available bandwidth offered by a conventional wireless system. Hence, a bandwidth sharing method must be used under realistic load conditions. The expected waiting time for a service offered by the VMG method is in the worst case (under low load conditions) the same waiting time offered by the conventional methods that use a unicast content distribution. As the demand for content increases, the superiority of the VMG method increases exponentially, in comparison to the conventional unicast-based methods. Under realistic load conditions (i.e., similar to the demand experienced for NVOD

service), only a VMG method can cope with an intensive demand for the same content. Another important feature is that an implementation of the VMG scheme requires only minor software modifications, mainly on the network equipment. Only a small modification is required on the user equipment.

Recently it has come to our attention that the idea to use the access point as a proxy server has been independently proposed in [7]. However, the system considered in [7] is a WLAN, while this paper considers a wireless network such as a WiMAX network, which is used for content distribution. The basic idea proposed in [7] is to cache the recently transmitted downstream data in the AP for a possible future use. This data may be reused when a node attempts to *transmit* data to a peer node. In this paper, the collaboration is conducted when neighboring nodes attempt to *receive* the same information. Thus, the collaboration mechanism is totally different from the one presented in [7].

1.2. Paper organization

The structure of the rest of this paper is as follows. Model and problem formulations are given in Section 2. The concept of multicast content distribution is introduced and analyzed in Section 3. Performance analysis and simulation results are provided in Section 4. A summary and concluding remarks are given in Section 5.

2. MODEL AND PROBLEM FORMULATION

This work considers a metropolitan WiMAX access network, that provides on-demand multimedia services. The network is a broadband wireless access network for stationary, or relatively slow, clients sharing the same access point (AP). The information is delivered to the clients through a downlink data channel, shared among the clients residing within the AP coverage area. This channel is controlled by the system, that allocates the bandwidth to the demanding clients. The wireless bandwidth is assumed to be sufficient to support multiple requests simultaneously, and to use statistical multiplexing in order to transmit multiple variable bit rate transmissions over one broadband channel. The time slot allocated to each subscriber by the IEEE 802.16 protocol can be enlarged in order to cope with variable bit rate transmissions. The clients are assumed to be stationary, in the sense that during the data retrieval session the user location is fixed.

The amount of data, measured in bytes, required to retrieve a content depends on the content size in bytes, on the receiving device, and on the quality of the user-network connectivity. The user receiver determines the compression standards (e.g., *MPEG*-2), and the user-network connectivity affects the packet loss probability through the session. Each personal content demand is associated with a priority R, defined as the network revenue associated with this content multiplied by the user waiting time for a service, and a cost M', defined as the information amount in bytes that must be transmitted. M' depends on M, the memory size of the required content, the compression and encoding standard used, and on the packets loss during the session (i.e., on the user-network connectivity). The *normalized priority* ϕ is defined by

$$\phi = \frac{R}{M'} = \frac{tr}{M'}, \tag{1}$$

where r is the network revenue associated with this demand and t is its waiting time. Most service providers charge a fixed price per content type (e.g., the same price for a movie). Hence, ϕ depends on R and M', which depends on the content size M, the encoding method, and the user-network connectivity.

It is assumed that the personal content demands initiated by the clients are generated according to a Poisson process with a mean λ arrival per time unit. The time duration of a session is exponentially distributed with a mean $1/\mu$ service time. The *load factor* ρ is defined as

$$\rho = \frac{\lambda}{\mu}. \tag{2}$$

The *service throughput* is defined as the expected number of personal content demands satisfied by the network per time unit. The network goal is to maximize the service throughput, without starving demands for unpopular content. Note that new content demands arrive online. Since certain personal content demands may have a higher priority than other demands (e.g., due to a longer waiting time), the problem of maximizing the service throughput can be generalized as specified below. The *weighted service throughput* is defined as the sum of priorities over all personal content demands satisfied by the network, that is, given a set of U clients, and a set of B content demands $\mathcal{B} = \{b_1, \ldots, b_B\}$, where each content demand b_i has (a) a priority P_i that depends on the number of users associated with this demand, on their waiting time, and on the required data and service $P_i = \sum_i R_i$ and (b) a weight ψ_i, that reflects the required download rate, measured in bits per time unit. The maximum *weighted service throughput* problem is defined as follows: given a set of B content demands, the goal is to maximize the achievable *weighted service throughput* Θ, under the condition that the system bandwidth capacity is bounded from above by a predefined constant C. That is, the goal is to select a subset $B' \subseteq B$ of content demands, such that the sum

$$\Theta = \sum_{j \in B'} P_j \tag{3}$$

is maximal, under the condition that the sum ψ of the weights of the content demands satisfied by the network is bounded from above by the network bandwidth capacity C. That is,

$$\psi = \sum_{j \in B'} \psi_j \leq C, \tag{4}$$

where Θ is defined as the *weighted service throughput*. Equation (4) is the constraint that must be satisfied by the optimal assignment defined in (3). It is assumed that a request for a content is transmitted through an up-link signaling channel, and the acknowledge message is transmitted by the local server through a down-link control channel. Clearly, the

problem of maximizing the service throughput is a special case of the maximum weighted service throughput problem. From now on we therefore consider only the problem of maximum weighted service throughput.

Theorem 1. *The problem of maximum weighted service throughput is NP-hard.*

Proof. Given a group of n bandwidth demands, the goal is to maximize the weighted service throughput. We now prove that this problem is NP-hard. The reduction is from the knapsack problem which is known to be NP-hard [8]. The input to the knapsack problem is a set of n elements. Each element has a size s_i and a weight w_i, where s_i and w_i are positive integers. The goal is to maximize the sum of weights of elements, under the condition that the sum of sizes is bounded from above by C, where C is a pre-defined constant, known as the knapsack capacity. That is, to find a subset of the elements such that $\sum s_i \leq C$ and $\sum w_i$ is a maximum.

Given a knapsack problem with n elements and a knapsack with a capacity of C units, we consider the following *maximum weighted service throughput* problem: the input to the *maximum weighted service throughput* problem is a system consisting of n bandwidth demands and a bandwidth capacity C. For each element i given as an input to the Knapsack problem, with a size s_i and a weight w_i, we associate a bandwidth demand with a priority w_i and a required bandwidth s_i. Note that the term "weight" has a different meaning by each problem: the weight w_i in the original Knapsack problem is associated with the priority P_i of the equivalent bandwidth demand in the *maximum weighted service throughput* problem, while the size s_i in the original Knapsack problem is associated with the weight (i.e., the required bandwidth) of the equivalent ith bandwidth demand in the *maximum weighted service throughput* problem. The assignment strategy that maximizes the *weighted service throughput* is the optimal assignment for the original knapsack problem. Clearly, the opposite direction is also true—and any *maximum weighted service throughput* problem is essentially a 0/1 knapsack problem (i.e., each element can be either assigned or not assigned to the knapsack). Hence, given a knapsack problem, an optimal assignment can be always found by a linear reduction to the equivalent *maximum weighted service throughput* problem, and vice versa. □

3. MULTICAST CONTENT DISTRIBUTION

The key idea of the proposed method is to establish an efficient mechanism for a multicast transmission to multiple points residing in the same area. Since the problem of optimal content distribution is NP-hard, a fast approximation algorithm is used, which can achieve any desired approximation ratio. To achieve this goal, the *VMG* server handles a queue of content demands. A unique task identification (ID) is assigned to each content demand, depending on the required service type, the required data, and the expected bandwidth (different devices requesting the same content may deserve/capable for a different download rate). The VMG server handles a *task table*. Each task has a unique entrance in this table, to be determined by its ID. The list of all its associated clients is attached to each task. Upon receiving a new content demand, the VMG server computes its unique task ID and checks if this ID already exists in the *task table*. If it does not exist, the demand is forwarded *immediately* to the wired backbone for data retrieval. On the other hand, if this ID already exists, that is, a request to retrieve this content has been already sent (by another client), there is no need for another content retrieval. The new client just joins to the list of clients requesting for this data and waiting for a reply. When the data is available, the VMG server distributes the data efficiently to the clients, using a single data channel. This distribution mechanism is achieved simply by using the VMG (i.e., the task) ID as the addressee. Hence, the expected waiting time for data retrieval can be only reduced, in comparison to the conventional content distribution methods, while the bandwidth utilization can be only improved. Since the VMG server distributes the data, it must also handle the transcoding problem. This problem is handled by retrieving the same content only once, then converting the code to the required formats, using different channels for the different formats. Hence, while the same data is retrieved only once, the VMG server may assign multiple channels to the same content whenever it is required. In these cases, each channel is associated with a different VMG. That is, the term VMG considers only one (multicast) channel, shared among multiple clients. Due to the transcoding problem, the same content can be potentially transmitted through multiple channels.

The basic mechanism that enables an AP to transmit data directed to multiple points only once is based on the broadcast nature of the wireless channel, and on the bandwidth allocation strategy of the IEEE 802.16 protocol. The signaling data is transmitted through a shared downlink channel controlled by the system. In order to receive data, each subscriber must track its own unique ID, and to response only to the messages directed to him/her. The proposed distribution mechanism is that each client tracks its relevant VMG ID, in addition to its own personal ID. The AP allocates the same data channel to all the members of the same VMG. Consequently, privacy and security are provided since the signaling is conducted individually. The data is provided only to the VMG members, since the date channel is allocated only to them. On the other hand, this data channel is shared among all the VMG members. Thus, redundant traffic is avoided since the same information is transmitted only once.

The proposed method can potentially increase the utilization of both the wireless links, as well as the wired links associated with the wireless network. Since data requested by many users residing in the same area is transmitted only once, the bandwidth efficiency of the wireless links can be only increased. In addition, since many requests for the same data are accumulated by the VMG server and sent to the wired backbone as a single request, also the bandwidth efficiency of the wired portion of the network should increase. In many applications, such as NVOD and live video streaming, the data is received simultaneously by many users residing in the same area.

3.1. The definitions of priority and cost

Given the condition that n users request the same data denoted by k, the priority P^k of the VMG associated with k is given by

$$P^k = \sum_{i=1}^{n} R_i^k = \sum_{i=1}^{n} r_i^k t_i^k, \tag{5}$$

where R_i^k is the priority associated with the personal content demand of the ith user for the data k, r_i^k is the network revenue from this personal demand, and t_i^k is the waiting time of this personal demand. The *priority density* π_k of the content demand k, that reflects the network priority per bandwidth unit associated with this demand, is defined by

$$\pi_k = \frac{P^k}{\psi^k} = \frac{\sum_{i=1}^{n} R_i^k}{\psi^k}, \tag{6}$$

where ψ^k is the cost, that is, the required download rate of k, given in (10).

3.2. The content distribution algorithm

The content distribution algorithm is as follows.

(1) Assign a priority and weight, and compute, using (6), the *priority density* for each bandwidth demand.
(2) Sort the bandwidth demands in a nonincreasing order by their *priority density*. For demands having the same *priority density*, sort in a nondecreasing order by their expected time duration T_i (i.e., the shorter request is selected first, in order to reduce the expected waiting time for a service).
(3) Assign the bandwidth demands using the rule that the most valuable demand (i.e., the demand with the larger *priority density*) comes first. Whenever a bandwidth demand b_i is assigned, the system available bandwidth capacity C is reduced by the *weight (rate)* w_i for the next T_i time units, and the *service throughput* Θ, given in (3), is increased by P_i. In case of a variable rate it is assumed that $w_i(t)$ is given for $0 \le t < T_i$. *Note that preemption is not allowed*. This process continues until either there is no available system bandwidth, or all the bandwidth demands are assigned. If either all the bandwidth demands are assigned, or all the system bandwidth is used, do not continue to the next step; the assignment algorithm is successfully completed, and the assignment is optimal. Otherwise, continue to the next step.
(4) Apply a nearly optimal assignment algorithm, with a relative precision ϵ, where ϵ is a predefined constant.

The proposed approximation algorithm is based on the fast approximation algorithm for the Knapsack problem proposed by Ibarra and Kim [9]. The key idea is to partition the bandwidth demands into small and large items. The weights of the large items are scaled, and a dynamic programming is used to find an optimal solution to the problem with scaled weights and capacity. Afterwards, the small items are added by a greedy algorithm as follows: Let S' denotes the *service throughput* assigned by the third step of the assignment algorithm. We define a normalized factor $\delta = S'(\epsilon/3)^2$. The approximation algorithm creates a table whose length is $\lfloor (3/\epsilon)^2 \rfloor$, that contains elements that the profit of each one of them is less than or equal to $S'\epsilon/3$. A dynamic programming is used to maximize the *service throughput* without violating the bandwidth capacity C, by adding elements from the table to the list of bandwidth demands already assigned. A detailed description of the algorithm of Ibarra and Kim can be found in [9]. Many theoretical and practical papers have addressed the issue of an approximation solution to the 0/1 Knapsack problem [10, 11]. Some of them have a better worst case running time and space complexity. However, these algorithms outperforms the approximation algorithm of Ibarra and Kim [9] only for a very large input. The number n of different bandwidth demands arriving to the same VMG server during a relatively short time period (the expected waiting time of a bandwidth demand) is not expected to be that large to justify a relatively complex and nontrivial approximation algorithm, as these algorithms cited above. For this reason, the performance of the approximation algorithm used in this study is expected to be superior to alternative methods, due to its simplicity and relative accuracy. For most practical cases, the number of different bandwidth demands handled by the VMG server is expected to be sufficiently small, such that an optimal solution can be easily derived by the proposed assignment algorithm.

3.3. Implementation considerations

The concept of VMGs can be implemented by a software module, referred to as the *VMG server*. A VMG server is an application layer module, to be installed at the gateway between the wireless network and the wired backbone. In order to handle privacy and security issues, this gateway is defined as a proxy server. The VMG server provides services such as live video streaming, NVOD, download of video and music files, multicast, and a video conference call. There is no need for a significant modification of the user equipment, but the capability to track the task ID (i.e., the VMG ID), in addition to its own user ID. In addition, whenever a user initiates a bandwidth demand, he/she must update its location upon every movement to another AP. The operation of reselecting an AP must lead to leave-multicast operation at one VMG, and join-multicast operation at another VMG. If either VMGs has only one client, then the VMG server should either delete the VMG (if its only client leave the group), or create a new VMG (if a new client attempts to join an empty VMG). From the user point of view, no hardware modification is required. Multiple download requests for the same file or service should be sent from the VMG server to the wired IP network as a single request, created by a virtual user (i.e., the VMG server). The task ID is used as the ID of the virtual multicast group. Once the data is retrieved, the VMG server has to distribute this data to the list of clients waiting for a reply. Using the generic multicast mechanism already provided by wireless networks, the VMG server application maps the clients to the multicast groups.

3.4. Transmission errors recovery

A wireless channel is likely to suffer from phenomenons such as high and variable round trip time, rate fluctuations, link outage, and occasional burst losses. The main result of these phenomenons is a loss of packets. Consequently, a wireless channel is in general less reliable than a wired channel. For this reason, packets loss must be considered. This section considers the recovery mechanism which handles packets loss. Since it is assumed that users can move only between sessions, but not during a session, it is assumed that the packet loss probability of a user u_i during the session is a constant p_i $(0 < p_i < 1)$. It is assumed that p_i can be estimated for a good approximation (e.g., using the signal to noise ratio, SNR, and/or historical data). The expected number of transmissions required for a successful transmission is therefore given by

$$\sum_{k=1}^{l} p_i^{k-1}(1-p_i)k < \sum_{k=1}^{\infty} p_i^{k-1}(1-p_i)k = \frac{1}{1-p_i}, \quad (7)$$

where l is the maximum number of re-transmissions allowed by the access protocol. In IEEE 802.11 a typical value of l is 7. Hence, the actual amount of data required to receive D bytes is bounded from above by $D/(1-p_i)$. The bandwidth required to receive the same information during the same time interval is therefore

$$\psi_i = \frac{\psi_0}{1-p_i}, \quad (8)$$

where ψ_0 is the bandwidth required for $p_i = 0$. Hence, the cost of satisfying the personal content demand initiated by u_i depends also on the network connectivity of u_i. Given the receiving device of u_i and the required content, the download rate ψ_0 is determined at purchase time. The actual download rate ψ_i must take into consideration the expected packets loss. Given a VMG (i.e., the group of users attached to the same content demand and having the same ideal download rate ψ_0), the multicast download rate is determined as follows: The first client who initiates the content demand set the multicast download rate to be equal to its own download rate ψ_i given in (8), and the packet loss probability plp of the *VMG* to be equal to its own packet loss probability. Every additional client u_j with an estimated packet loss probability p_j increases the *VMG* plp by $p_j(1-\text{plp})$:

$$\text{plp} = \text{plp} + (p_j - p_j\text{plp}) = \text{plp} + p_j(1-\text{plp}). \quad (9)$$

Using the same analysis that lead to (8), the *VMG* required download rate is given by

$$\psi_{\text{vmg}} = \frac{\psi_0}{1-\text{plp}}. \quad (10)$$

3.5. Motivation

The motivation for using the VMG mechanism depends on the load on the shared data channel. Assuming a Poisson arrival rate with a mean λ arrival per time unit, and an exponentially distributed with a mean $1/\mu$ service (i.e., session) time, a conventional content distribution server behaves as an $M/M/1$ system. The expected waiting time for data retrieval is given by (see, e.g., [12])

$$W = \frac{\rho/\mu}{1-\rho}. \quad (11)$$

It follows from (11) that the waiting time grows exponentially with the *load factor* $\rho = \lambda/\mu$ [12]. As either the arrival rate or the service time increases, so does the waiting time. Video applications are characterized by a relatively long session time. For example, a typical time duration of live streaming of a movie is about 90 minutes. Moreover, the arrival rate of demands for popular data (e.g., popular video files, news broadcast) is also expected to be relatively high. Hence, even a moderate demand for video services can easily jam the wireless channel, unless some sharing method is used, that enables reusability of the wireless channel. As the arrival rate for a specific content is sufficiently high, whenever the time interval between two consecutive arrivals of requests to the same content is smaller than the waiting time of the first arrival, the second request for that content can just join to the first request and they can both share the same data channel, thus reducing their average waiting time. Given a rate of λ arrivals per time unit of demands for the same content, the expected time interval between two consecutive arrivals is $1/\lambda$. Upon a new arrival, the condition under which its expected waiting time for a service is longer than the expected waiting time for a new arrival of a request for the same content is given by

$$W > \frac{1}{\lambda}. \quad (12)$$

To simplify the analysis, it is assumed that only one stream is served. Substitute W from (11), then we get that the condition under which a new demand for the same content is expected to join the first demand while it is still waiting for a service is given by

$$W = \frac{\rho/\mu}{1-\rho} > \frac{1}{\lambda}. \quad (13)$$

Multiply both sides by λ and substitute $\rho = \lambda/\mu$, then we get that

$$\frac{\rho^2}{1-\rho} > 1, \quad (14)$$

which leads to

$$\rho^2 + \rho - 1 > 0. \quad (15)$$

The value of ρ which satisfies (15) and the condition $\rho \geq 0$ is

$$\rho > \frac{\sqrt{5}-1}{2} \approx 0.618. \quad (16)$$

Equation (16) implies that whenever the *load factor* satisfies the condition $\rho > 0.618$ a batch service outperforms an individual service. That is, whenever $\rho > 0.618$, the expected time duration between two consecutive arrivals is shorter than the

expected waiting time for a service. Thus, a new arrival is likely to join a previous arrival. Consequently, the expected waiting time for a service is reduced. It follows from (16) that the superiority of bandwidth sharing increases with the arrival rate for this content, and with the expected service time and waiting time. As the number of popular files/streams increases, the service rate increases to $E[s]\mu$, where $E[s]$ is the expected number of users sharing the same data channel. The expected waiting time for an FIFO service policy is therefore

$$W_F = \frac{\rho/E[s]\mu}{1-\rho/E[s]} \leq W, \tag{17}$$

where equality holds if, and only if, $E[s] = 1$. That is, the *load factor* ρ is sufficiently small such that no bandwidth sharing is used.

4. PERFORMANCE ANALYSIS AND SIMULATION RESULTS

In this section the performance of the proposed VMG method is compared to the performance of the conventional method currently used by existing wireless networks that uses a unicast content distribution. Note that it follows from (9) that the bandwidth required for multicast content distribution is always less than or equal to the bandwidth utilization of the conventional content distribution that uses unicast.

Theorem 2. *Given a constant number of content items (i.e., VMGs), the waiting time of any personal demand served by an FIFO scheduling is bounded from above by a predefined constant.*

Proof. Let n be the number of supported VMGs. Let T_{all} be the total service time of all the VMGs:

$$T_{\text{all}} = \sum_{i=1}^{n} t_i, \tag{18}$$

where t_i is the expected service time of the ith VMG. Then, under the worst case condition, a content demand arrived just after starting the download of its associated VMG, and cannot join to an ongoing session, has to wait to all the other $n-1$ VMGs to be served, as well as to its own VMG to be served. Hence, the waiting time is bounded from above by the time duration required to serve these n VMGs. This time period is bounded from above by T_{all}, and this bound is not tight whenever some of the VMGs can be served simultaneously. □

Let us consider a system in which content demands for a particular content are generated according to a Poisson process with a mean λ arrivals per time unit. Given that a new arrival has to wait Δt time units for a service, the probability of a new arrival during the waiting time of this demand, of another demand for this content is given by

$$P_{\text{batch}} = 1 - e^{-\lambda\Delta t}. \tag{19}$$

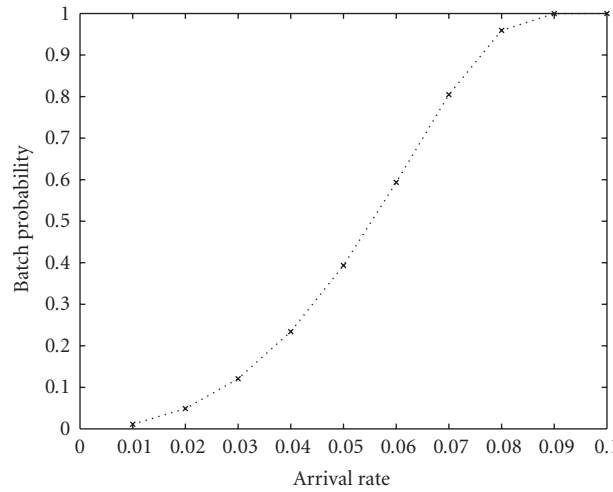


Figure 1: The batch probability of creating a VMG containing at least two collaborating clients, as a function of the expected arrival rate.

Using the individual service policy currently used, the expected waiting time is given in (11). Substitute the expected waiting time for Δt in (19), then we get the value of P_{batch} as a function of λ and μ. Figure 1 depicts the batch probability of at least two collaborating clients served simultaneously as a function of the arrival rate for the same specific content. Only this content is downloaded, and the download time is exponentially distributed with a mean of 10 time units. It is clearly demonstrated that even for a relatively low arrival rate of 0.1 arrival per time unit, the probability of accumulating at least two clients in the same VMG is practically 1. For instance, if the expected download time of a video file is 10 minutes, even for one arrival per 18 minutes the probability of having $n_i > 1$ is greater than 0.5. As the arrival rate approaches to one arrival per 10 minutes the expected waiting time of an $M/M/1$ system that uses a unicast content distribution actually explodes, and a batch service must be used. As the download time increases, so does ρ, the motivation for using multicast content distribution increases. For instance, if the expected time duration of a session is 30 minutes, a typical value for a popular TV show, or news broadcast, then even for one arrival per 54 minutes the probability of having more than one client at the same multicast group is greater than half.

4.1. Simulation results

Figure 2 depicts the expected waiting time for content delivery, for multicast content distribution and for unicast content distribution, as a function of the arrival rate to the system. The content demands are generated according to a Poisson process, and are equally distributed among 30 files. The download time of a file is exponentially distributed with a mean of 20 minutes. The system bandwidth capacity can support only 10 download requests at a time. As the arrival rate increases, the expected waiting time of a unicast content distribution increases exponentially, and the superiority

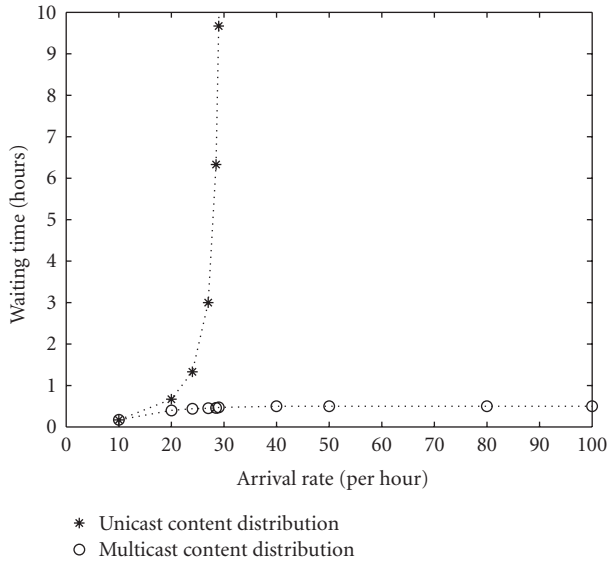


Figure 2: The expected waiting time offered by multicast content distribution, and by unicast content distribution methods, as a function of the arrival rate to the system, for a homogeneous load distribution.

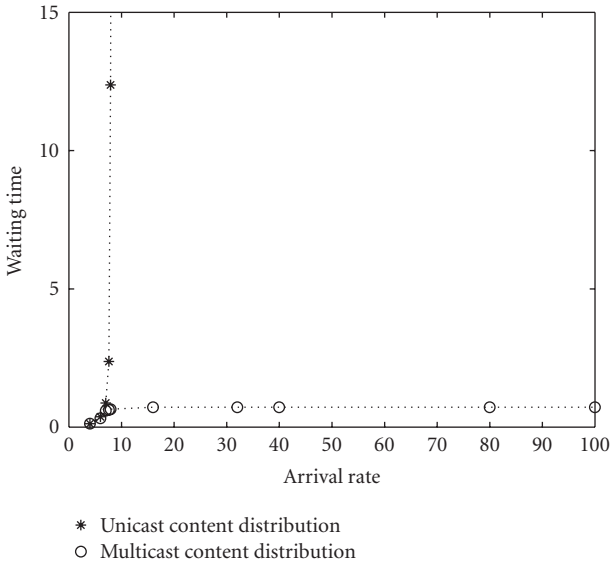


Figure 3: The waiting time in hours, as a function of the arrival rate to the system per hour, for a nonhomogeneous load distribution, in comparison to a unicast content distribution.

of a multicast content distribution over a unicast service is demonstrated very clearly.

Figure 2 considers content demands that are equally distributed among different video files. In reality, different video files are requested at different rates. The pattern of arrival rate to video on demand service and video rental stores [13] indicates that the majority (about 60%–80%) of the requests are for a few files. Many simulation experiments were conducted, for different load distribution patterns, and download time. Due to space limitation, two representative results are given

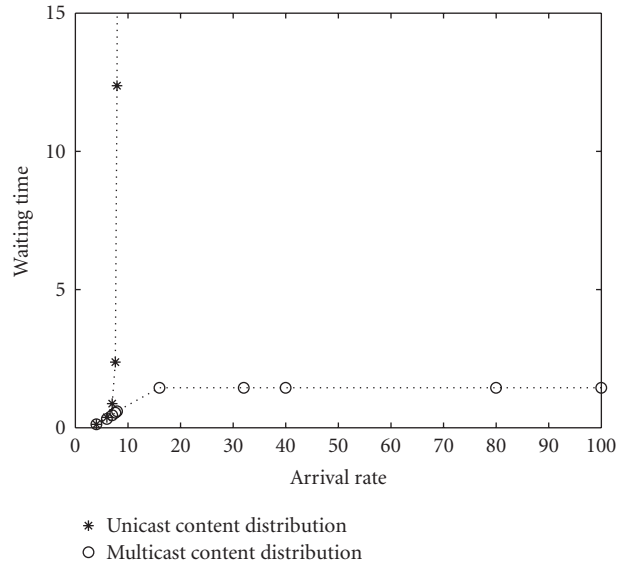


Figure 4: The waiting time in hours, as a function of the arrival rate per hour to the system.

below. Figure 3 considers a system that supports 16 different files. 68% of the new arrivals are equally distributed among 4 most popular files, other 20% of the arrivals are equally distributed among other 4 files, and the rest of 12% of the new arrivals are equally distributed among the 8 least popular files. The average download time of each file is 30 minutes, and an AP can support 4 different download requests simultaneously. New arrivals were generated according to a Poisson process, and the download time was exponentially distributed. The arrival rate ranges up to 100 arrivals per hour to an AP. Considering an AP that supports 1000 clients, this arrival rate represents a rating of 10% of the customers per hour, which is a relatively low arrival rate for cable TV and video-on-demand systems. Note that under these conditions, a unicast content distribution explodes as soon as the arrival rate to the AP approaches to 8 arrivals per hour.

Figure 4 considers a system that supports a group of 33 files. 3 popular files are requested by 20% of the clients per each file, while the other 40% of new arrivals are equally distributed among 30 less popular files. The system bandwidth capacity and the expected download time are both the same as in Figure 3. The superiority of multicast content distribution over a unicast content distribution is clearly demonstrated.

Due to the nature of the wireless fading channels, it may not be always possible to assign the same data channel to all the clients requesting for the same stream. Thus, it may not be always possible to assign all the clients requesting for the same content at the same time to a single data channel, since such an assignment may cause a significant variance in the signal-to-noise ratio experienced by different clients. However, since different clients can share the same data file, the memory utilization of the AP must be better. Thus, the expected number of clients sharing the same content at the same time can be used as a reliable measurement tool for the

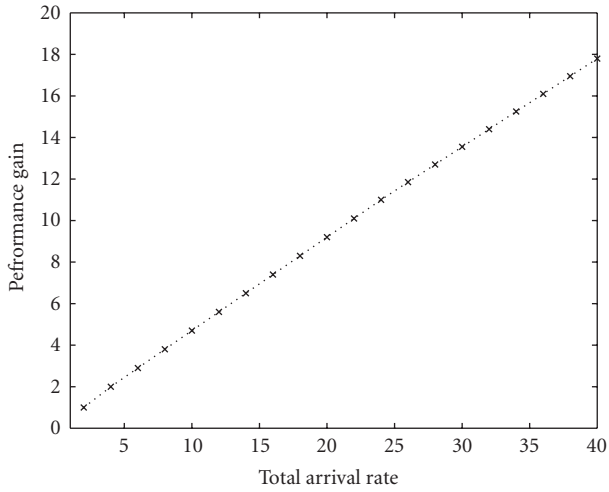


Figure 5: The performance gain, measured by the average number of clients sharing the same downloaded file, as a function of the total arrival rate per hour to a system which serves 100 clients.

performance gain achieved by the batch service. This performance gain can be reflected either by the utilization of the data channel, and consequently reducing the expected waiting time as predicted by (17), or by the AP memory utilization, or both. Figure 5 depicts the average number of clients sharing the same content $E[s]$ given in (17), as a function of the total arrival rate to the system, for an AP which serves 100 clients, and a realistic load distribution [13]. It is clearly shown that even for a moderate demand, the performance gain is significant. A group of 100 clients sharing the same AP can be expected for a WiMAX network which serves few buildings.

5. SUMMARY AND CONCLUDING REMARKS

The main advantage of multicast content distribution over the conventional unicast content distribution is its scalability. Due to the huge bandwidth consumed by video applications, and the relatively long-time duration of a typical session, even a moderate demand for video applications is sufficient to block a conventional wireless channel that uses a unicast content distribution. Hence, a collaborative scheme must be used, in order to enable bandwidth sharing. Under realistic load conditions, a multicast content distribution yields a significantly shorter waiting time than a unicast content distribution. Being implemented at the access portion of the network, the construction of the multicast groups is straightforward.

REFERENCES

[1] http://www.dvb.org.
[2] http://www.qualcomm.com/mediaflo.
[3] S. Chen, B. Shen, S. Wee, and X. Zhang, "Adaptive and lazy segmentation based proxy caching for streaming media delivery," in *Proceedings of the 13th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '03)*, pp. 22–31, Monterey, Calif, USA, June 2003.
[4] S. Ganguly, A. Saxena, S. Bhatnagar, R. Izmailov, and S. Banerjee, "Fast replication in content distribution overlays," in *Proceedings of the 24th IEEE Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '05)*, vol. 4, pp. 2246–2256, Princeton, NJ, USA, March 2005.
[5] L. Guo, S. Chen, S. Ren, X. Chen, and S. Jiang, "PROP: a scalable and reliable P2P assisted proxy streaming system," in *Proceedings of the 24th International Conference on Distributed Computing Systems Workshops (ICDCS '04)*, vol. 24, pp. 778–786, Tokyo, Japan, March 2004.
[6] M. Guo and M. H. Ammar, "Scalable live video streaming to cooperative clients using time shifting and video patching," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '04)*, vol. 3, pp. 1501–1511, Hong Kong, March 2004.
[7] E. Tan, L. Guo, S. Chen, and X. Zhang, "SCAP: smart caching in wireless access points to improve P2P streaming," in *Proceedings of 24th International Conference on Distributed Computing Systems Workshops (ICDCS '07)*, Toronto, Canada, June 2007.
[8] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, New York, NY, USA, 1979.
[9] O. H. Ibarra and C. E. Kim, "Fast approximation algorithm for the Knapsack and sum of subsets problems," *Journal of the ACM*, vol. 22, no. 4, pp. 463–468, 1975.
[10] H. Kellerer, R. Mansini, U. Pferschy, and M. G. Speranza, "An efficient fully polynomial approximation scheme for subset-sum problem," *Journal of Computer and Systems Science*, vol. 66, no. 2, pp. 349–370, 2003.
[11] M. J. Magazine and O. Oguz, "A fully polynomial approximation algorithm for the 0-1 knapsack problem," *European Journal of Operational Research*, vol. 8, no. 3, pp. 270–273, 1981.
[12] L. Kleinrock, *Queueing Systems*, vol. 1, 2, John Wiley & Sons, New York, NY, USA, 1976.
[13] C. Griwodz, M. Bär, and L. C. Wolf, "Long-term movie popularity models in video-on-demand systems or the life of an on-demand movie," in *Proceedings of the 5th ACM International Conference on Multimedia (MULTIMEDIA '97)*, pp. 349–357, Seattle, Wash, USA, November 1997.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2008, Article ID 265309, 9 pages
doi:10.1155/2008/265309

Research Article

A Stream Tapping Protocol Involving Clients in the Distribution of Videos on Demand

Santosh Kulkarni, Jehan-François Pâris, and Purvi Shah

Department of Computer Science, University of Houston, Houston, TX 77204-3010, USA

Correspondence should be addressed to Jehan-François Pâris, paris@cs.uh.edu

Received 1 May 2007; Revised 7 November 2007; Accepted 11 January 2008

Recommended by Qian Zhang

We present a stream tapping protocol that involves clients in the video distribution process. As in conventional stream tapping, our protocol allows new clients to tap the most recent broadcast of the video they are watching. While conventional stream tapping required the server to send to these clients the part of the video they missed, our protocol delegates this task to the clients that are already watching the video, thus greatly reducing the workload of the server. Unlike previous solutions involving clients in the video distribution process, our protocol works with clients that can only upload video data at a fraction of the video consumption rate and includes a mechanism to control its network bandwidth consumption.

1. INTRODUCTION

Distributing videos on demand is a costly proposition, mostly because of the high bandwidth requirements of the service. Assuming that the videos are in MPEG-2 format, each user request will require the delivery of approximately six megabits of data per second Hence, a video server allocating a separate stream of data to each request would need an aggregate bandwidth of six gigabits per second to accommodate one thousand overlapping requests.

This situation has led to numerous proposals aimed at reducing the bandwidth requirements of VOD services. These proposals can be broadly classified into two groups Proposals in the first group are said to be *proactive* because they distribute each video according to a fixed schedule that is not affected by the presence—or the absence—of requests for that video. They are also known as *broadcasting* protocols. Other solutions are purely *reactive*: they only transmit data in response to a specific customer request. Unlike proactive protocols, reactive protocols do not consume bandwidth in the absence of customer requests.

Nearly all these proposals assume a clear separation of functions between the server, which distributes the video and the customers, who watch it on their personal computers or on their television sets. As a result, they cannot take advantage of the upstream bandwidth of the clients to lower the server's workload.

The stream tapping protocol we present here is the first protocol that can harness the collective bandwidth of clients with limited individual upstream bandwidths As in conventional stream tapping, our protocol requires the server to start a new video broadcast whenever a client cannot get enough video data by "tapping" a previous broadcast of the same video. Unlike conventional stream tapping, our protocol uses the available upstream bandwidth of previous clients to reduce the amount of video data that the server will still have to send to the clients that "tap" a previous broadcast of the video As we will see, delegating these tasks to the clients results in a dramatic reduction of the server workload at medium to high request arrival rates.

The remainder of this paper is organized as follows Section 2 reviews previous work Section 3 introduces our stream tapping protocol Section 4 discusses its performance and shows how we can limit the network bandwidth consumption of the protocol at high arrival rates Section 5 presents a simple probabilistic model of our protocol and Section 6 discusses its applicability to actual networks Finally, Section 7 has our conclusions

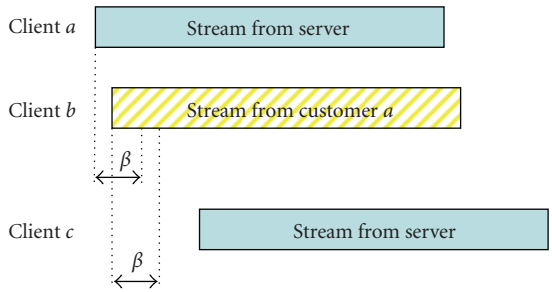


Figure 1: How chaining works.

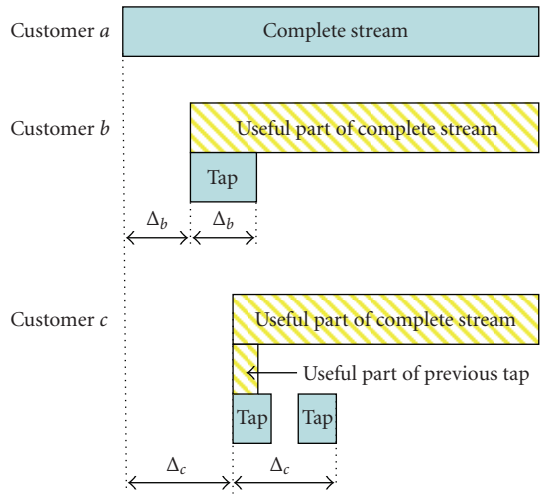


Figure 2: How stream tapping works.

2. PREVIOUS WORK

Two of the earliest reactive distribution protocols are batching and piggybacking. *Batching* [1] reduces the bandwidth requirements of individual user requests by multicasting one single data stream to all customers who request the same video at the same time. *Piggybacking* [2] adjusts the display rates of overlapping requests for the same video until their corresponding data streams can be merged into a single stream Consider, for instance, two requests for the same video separated by a time interval of three minutes. Increasing the display rate of the second stream by 10 percent will allow it to catch up with the first stream after 30 minutes.

Chaining [3] was the first video distribution protocol to take advantage of the upstream bandwidth of its clients It constructs chains of clients such that (a) the first client in the chain receives its data from the server, and (b) subsequent clients receive their data from their immediate predecessor As a result, video data are "pipelined" through the clients belonging to the same chain. Since chaining only requires clients to have very small data buffers, a new chain has to be restarted every time the time interval between two successive clients exceeds the capacity β of the buffer of the first client Figure 1 shows three sample client requests Since client a is the first customer, it will get all its data from the server As client b arrives less than β minutes after customer a, it can receive all its data from client a Finally, client c arrives more than β minutes after client a and must be serviced directly by the server. *Optimized chaining* [4] exploits the buffers of other clients in order to construct longer chains and reduce the server workload.

The cooperative video distribution protocol [5] extends the chaining protocol by taking advantage of the larger buffer sizes of modern clients Hence, it should be better named *extended chaining* If all clients have buffers large enough to store the entire video, the server will never have to transmit video data at more than the video consumption rate.

Stream tapping [6, 7], also known as *patching* [8], requires each client set-top box to have a buffer capable of storing at least 10 to 15 minutes of video data and to be able to receive data at at least twice the video consumption rate This buffer will allow the set-top box to "tap" into data streams that were originally created for previous clients, and then store these data until they are needed In the best case, clients obtain most of their data from an existing stream.

In particular, stream tapping defines two types of streams. *Complete streams* broadcast a video in its entirety. *Full tap streams* can be used if a complete stream for the same video started $\beta \leq b$ minutes in the past, where b is the size of the client buffer, measured in minutes of video data. In this case, the client begins receiving the complete stream right away, storing the data in its buffer Simultaneously, it receives a full tap stream and uses it to display the first Δ minutes of the video. After that, the client will consume directly from its buffer Stream tapping also defines *partial tap streams*, which can be used when $\Delta > \beta$. In this case, clients must go through cycles of filling up and then emptying their buffer, since the buffer is not large enough to account for the complete difference in video position.

To use tap streams, clients need only receive at most two streams at any one time. Clients that can receive data at three times the video consumption rate can use an option of the protocol called *extra tapping*. Extra tapping allows clients to tap data from any stream on the VOD server, and not just from complete streams. Figure 2 shows some samples of client requests As client a is the first client, it is serviced by a complete stream, whose duration is equal to the duration D of the video. Since client b arrives Δt minutes after client a, it can share $D - \Delta t$ minutes of the complete stream and only requires a full tap of duration Δt minutes Finally, customer c can use extra tapping to tap data from both the complete stream and the previous full tap, and so its service time is smaller than Δ_c.

Eager and Vernon's *dynamic skyscraper broadcasting* (DSB) [9] is another reactive protocol based on Hua and Sheu's *skyscraper broadcasting* protocol [10, 11] Like skyscraper broadcasting, it never requires the STB to receive more than two streams at the same time Their more recent *hierarchical multicast stream merging* (HMSM) protocol requires less server bandwidth than DSB to handle the same request arrival rate [12] Its bandwidth requirements are indeed very close to the upper bound of the minimum bandwidth for a reactive protocol that does not require the

STB to receive more than two streams at the same time, that is,

$$\eta_2 \ln\left(1 + \frac{N_i}{\eta_2}\right), \quad (1)$$

where $\eta_2 = (1 + \sqrt{5})/2$, and N_i is the request arrival rate.

Selective catching [13] combines both reactive and proactive approaches. It dedicates a certain number of channels for periodic broadcasts of videos, while using the other channels to allow incoming requests to catch up with the current broadcast cycle. As a result, its bandwidth requirements are $O(\log(\lambda_i L_i))$, where λ_i is the request arrival rate and L_i the duration of the video.

3. OUR PROTOCOL

Both chaining and the cooperative protocol require clients capable of sending video data at the video consumption rate As a result, they exclude clients that have limited upstream bandwidths, say, no more than one eighth to one fourth of their downstream bandwidths While these clients might be able to download video data at twice their video consumption rate, they might only be able to forward video data at one fourth to one half of that rate.

We wanted to develop a video distribution protocol that allowed clients to participate in the video distribution process even if they could only retransmit data at a fraction of the video consumption rate [14] We thus assumed the following:

(1) clients would be able to receive video data at twic their video consumption rate,

(2) clients would only be able to forward video data at a rate equal to a fraction α of the same video consumption rate,

(3) clients would not be able to multicast video data,

(4) clients would not have to forward video data after they have finished watching that video,

(5) clients should have enough buffer space to store the previously viewed portion of the video they are watching until they have finished watching it.

As we can see, our protocol makes few demands on the transmission capabilities of the client hardware In contrast, it requires client buffers capable of storing an entire video, that is, several gigabytes of compressed video data Two factors motivated this choice First, the diminishing cost of every kind of storage makes this requirement less onerous today than it would have been a few years ago Second, we expected many clients to keep the previously viewed portion of the video they are watching in their buffers in order to provide the equivalent of a VCR rewind feature.

Our protocol is a fairly straightforward implementation of stream tapping without extra tapping as extra tapping would have required clients to be able to receive videos at three times the video consumption rate It only differs from the original stream tapping protocol in the way it handles its tap streams While tap streams originally were the sole responsibility of the server, this task is now shared by the server and client b. Let us consider the scenario described in Figure 3 and focus on the request issued by client c. Let $\Delta t'$ denote the time interval between that request and a request issued by client b, and let T_c denote the time elapsed since the start of the last request that was serviced by a complete stream.

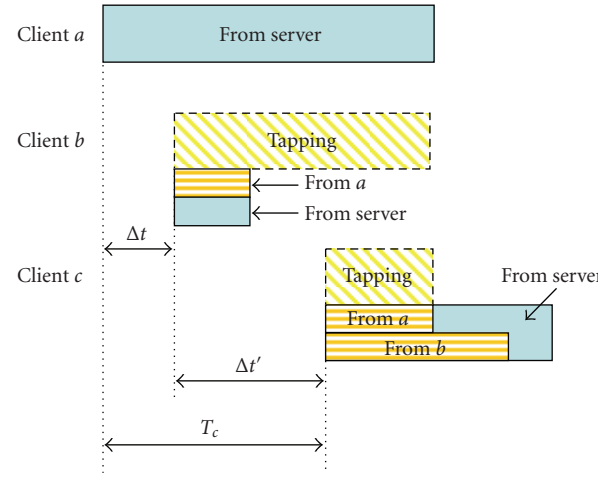


FIGURE 3: How the full tap streams are distributed among the server and the previous clients.

(1) If $T_c \geq D$, the two requests do not overlap and client c cannot tap any data from the last complete stream As in the original stream tapping protocol, the server will then start a new complete stream.

(2) If $T_c < D$, there is an overlap between the current request and the last complete stream As in the original stream tapping protocol, the server will then evaluate whether it would be more advantageous to keep tapping from the last complete stream or to start a new one If the server decides to keep tapping from the last complete stream, it will have to provide client c with a full tap stream of duration T_c Two alternatives must now be considered:

(a) if $T_c \leq D - \Delta t'$, client b will provide a fraction α of the full tap stream and the server the remaining $1 - \alpha$ fraction of the stream;

(b) if $T_c > D - \Delta t'$, client c will finish watching the video before being able to transmit all its share of the full tap stream, and will only be able to transmit a fraction $\alpha(D - \Delta t')/T_c$ of the full tap stream with the server transmitting the remainder of the stream.

If the video is long enough, the new request is likely to overlap with more than one previous request We propose to harness the available bandwidth of the clients that issued these requests in order to further reduce the workload of the server. The contributions of these clients will be subject to two restrictions. First, upstream bandwidth restrictions prevent any client to upload data for two different clients at the same time. Second, we will never require a client to transmit video data after the client has finished watching the video.

In our example, the request form client a is entirely serviced by a complete stream coming from the server The request from client b gets the last $D - \Delta t$ minutes of the video by tapping client a's complete stream and the first Δt minutes from a full tap stream of duration Δt A fraction α of this stream will be sent by customer a, and the remaining $1 - \alpha$ fraction will come from the server Assuming that the server decides not to start a new complete stream for customer c, that customer would get the following.

(1) The last $D-(\Delta t+\Delta t')$ minutes of the video by tapping client a's complete stream.

(2) A fraction α of the first $D - (\Delta t + \Delta t')$ minutes of the video from a tap stream sent by customer a; this tap stream will end when customer a will finish watching the video $D - (\Delta t + \Delta t')$ minutes after the arrival of customer c.

(3) A fraction α of the first $D - \Delta t'$ minutes of the video from a tap stream sent by customer b; this tap stream will end when customer b will finish watching the video $D - \Delta t'$ minutes after the arrival of customer c.

(4) The remaining portion of the first $\Delta t + \Delta t'$ minutes of the video from the server.

One last issue to consider is when to halt tapping from the current complete stream and start a new one.To achieve this goal, our protocol keeps track of the minimum average request service time of all requests sharing the same complete stream Before adding a new request to a group, it computes what would be the new average request service time of the group if the new request was added to the group Should this new average request service time be lesser than or equal to the minimum average request service time of the group, our protocol adds the new request to the group; otherwise, it starts a new group This criterion is similar but not identical to that used by Carter and Long [6, 7].

3.1. Handling client failures

To operate correctly, our protocol requires all clients to forward video data to the next customers for the same video Any client failure will deprive all subsequent customers from their video data.

There is a simple solution to the problem Let us return to the scenario of Figure 3, where client c receives most of its tap stream from clients a and b, while client b receives almost half of its tap stream from client a Any failure of either client a or client b would immediately affect the correct flow of data to client c A failure of client a will require the server to take over the role of client a and send the missing video data to clients b and c. A failure of client b would have less impact on the server workload as it would also free client a from its obligation to send client b a fraction of its tap stream, thus freeing enough upstream bandwidth to let client a take over the role of client b and send most of the missing video data to client c Making the protocol fault tolerant will thus require the server to keep track of which client is sending video data to each client.

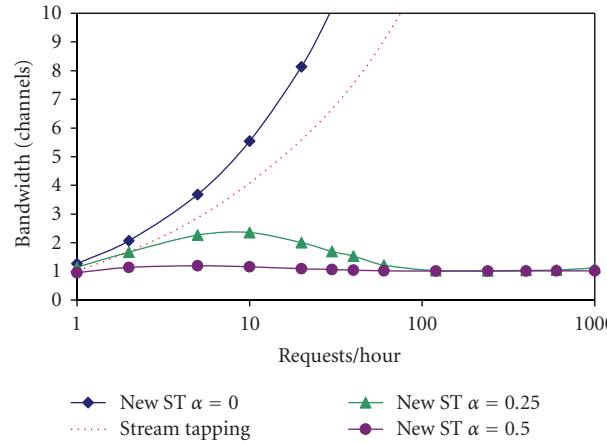


Figure 4: Server bandwidth requirements of the new stream tapping protocol.

4. PERFORMANCE EVALUATION

To evaluate the performance of our new protocol, we designed a simulator modeling the distribution of a two-hour video assuming that request arrivals could be modeled by a Poisson process [14]. It is based on similar simulators used for previous papers on stream tapping and was modified to model clients that could not forward video data at a rate equal to the video consumption rate [15, 16].

Figure 4 displays the server bandwidth requirements of our new stream tapping protocol for selected values of α and request arrival rates varying between one and one thousand requests per hour All bandwidths are expressed in multiples of the video consumption rates In addition, the dotted line represents the server bandwidth requirements of the original stream tapping protocol with extra tapping Let us observe that the comparison between the two protocols is not totally fair since extra tapping requires clients capable of receiving video data at three times the video consumption rate, while our protocol only requires clients capable of receiving video data at two times that rate.

As we can see, our new stream tapping protocol outperforms conventional stream tapping even when clients can only forward data at one fourth of the video consumption rate, that is, when $\alpha = 0.25$. These results are much better than those of an earlier version of the protocol that would not allow clients to receive video data from more than one client [16].

This excellent performance comes however at a stiff price As seen in Figure 5, the network bandwidth requirements of our stream tapping protocol increase much more rapidly than those of the original stream tapping protocol when the client request arrival rate exceeds ten requests per hour. This phenomenon can be explained in part by the fact that our protocol does not allow extra tapping A more important factor is the way the server decides when to start a new complete stream. Since the clients handle most of the tap streams, adding extra requests to any existing group has a negligible impact on the server workload As a result, the server never starts a new complete stream before the end

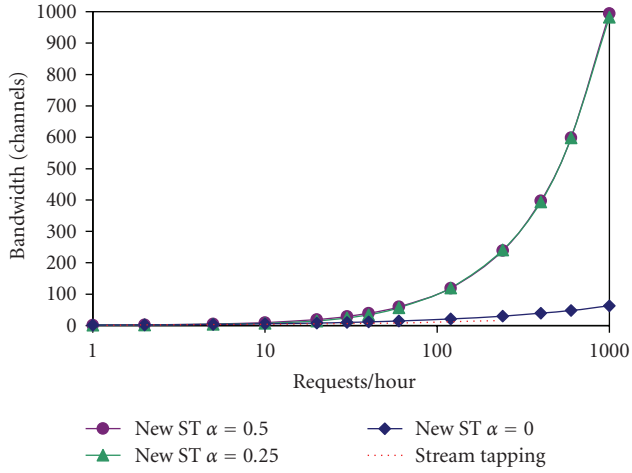


FIGURE 5: Network bandwidth requirements of the new stream tapping protocol.

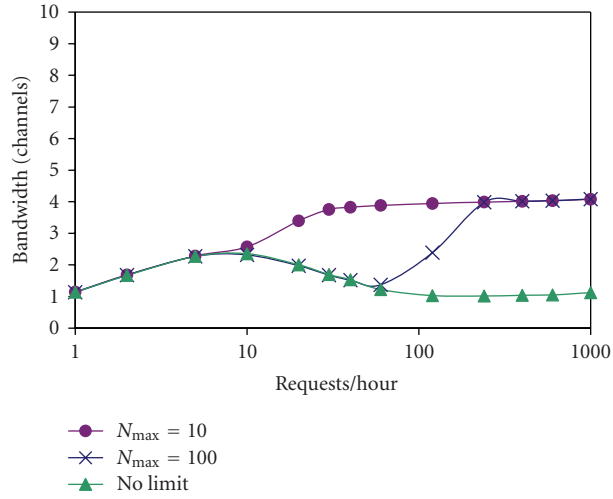


FIGURE 6: Server bandwidth requirements of the protocol for $\alpha = 0.25$, and $\beta_{max} = 0.25$.

of the previous one. Thus the average duration of a tap stream is equal to half the duration of the video and the average network bandwidth is roughly equal to one half the bandwidth required by a unicast scheme.

A simplistic solution to this problem would be to limit the size of the tap streams to a fraction β_{max} of the duration of the video. This would reduce the average duration of these streams and proportionally reduce the network bandwidth. This solution would however affect the performance of the protocol at low-arrival rates, where long tap streams are the norm Having investigated several other options, we found out that the best way to limit the growth of the network bandwidth was to limit the size of the tap streams at high arrival rates. At the same time, we did not want to complicate the design of the server by requiring it to maintain some moving average of the request arrival rates for each video.

We decided instead to use as threshold the number of clients sharing the same complete stream and force the server to start a new complete stream whenever (a) the size of the tap stream would otherwise exceed a fraction β_{max} o the duration of the video, and (b) more than N_{max} requests were already sharing the current complete stream.

Figures 6 and 7 display the impact of this modification to the server and network bandwidth of our protocol. We considered clients capable of uploading data at one-fourth the video consumption rate and set our β_{max} to 0.25. Each individual curve corresponds to a different value of N_{max}. We see that limiting the tap stream length to one quarter of the video duration reduces by a factor of four the network bandwidth of the protocol, while increasing the server bandwidth at the highest arrival rates by the same factor. Even under these conditions, the server bandwidth remains well below that of the original stream tapping protocol.

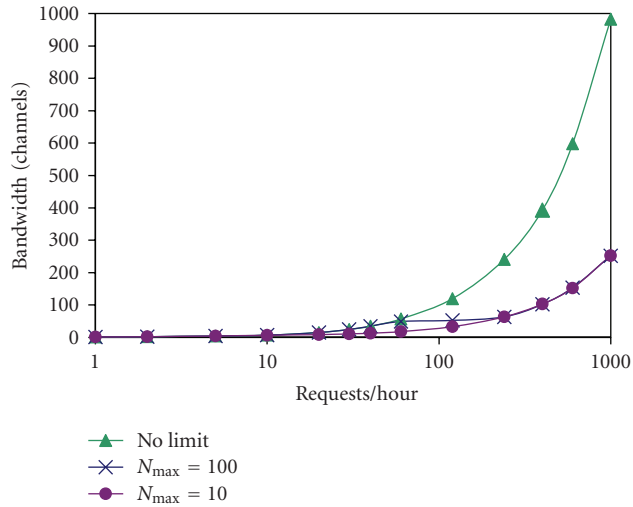


FIGURE 7: Network bandwidth requirements of the protocol for $\alpha = 0.25$, and $\beta_{max} = 0.25$.

5. AN ANALYTICAL MODEL

As in a previous paper [17], we consider stream tapping without extra tapping for a video of duration D that is being accessed at a rate λ. We assume that we will restart a new complete stream whenever the length of the next full tap exceeds βD, where $0 < \beta \leq 1$ is a parameter to be determined.

In other words, we will wait for an incoming request, start a complete stream of duration D, tap this stream over a time interval of duration βD, then restart the process.

During this time interval, the server will process an average of $\lambda\beta D$ requests in addition to the request that prompted the complete stream Hence the average number of requests sharing the same complete stream is

$$n_{avg} = 1 + \lambda\beta D. \quad (2)$$

Since the lengths of the $\lambda\beta D$ full tap streams in the group will be uniformly distributed over the interval $(0, \beta D]$, the average duration of each of these streams will be

equal to $\beta D/2$ The total duration of the streams required for processing these $1 + \lambda\beta D$ requests will thus be

$$W = D + (\lambda\beta D)\left(\frac{\beta D}{2}\right) = D + \frac{\lambda\beta^2 D^2}{2}. \quad (3)$$

We define the average service time T_{avg} for the requests in a group as the total duration W of the streams required for processing these requests divided by the number n_{avg} of requests in the group We will then have

$$T_{\text{avg}} = \frac{W}{n_{\text{avg}}} = \frac{D + \lambda\beta^2 D^2/2}{1 + \lambda\beta D}. \quad (4)$$

Our protocol assumes that previous clients will share with the server the task of sending the full tap streams Let $\gamma \leq 1$ denote the fraction of the tap streams that the clients will be able to send. This fraction γ will depend both on the factor α characterizing the upload bandwidth of the clients and the existence of previous clients whose requests overlap with the current requests. As the request arrival increases, the number of these clients will also increase, which will allow them to send a larger fraction γ of the tap streams.

For a given value of γ, the contribution of the server to the average request service time will be

$$T_{s,\text{avg}} = \frac{D + (1-\gamma)\lambda\beta^2 D^2/2}{1 + \lambda\beta D}, \quad (5)$$

and that of the previous clients will be

$$T_{c,\text{avg}} = \frac{\gamma\lambda\beta^2 D^2/2}{1 + \lambda\beta D}. \quad (6)$$

To find the value β_{opt} of the coefficient β that minimizes the server workload, we differentiate (5) with respect to β, obtaining

$$\frac{(1-\gamma)\lambda\beta D^2(1+\lambda\beta D) - \lambda D(D + (1-\gamma)\lambda\beta^2 D^2/2)}{(1+\lambda\beta D)^2}, \quad (7)$$

the only positive root of which

$$\beta_{\text{opt}} = \frac{\sqrt{2\lambda D(1-\gamma) + (1-\gamma)^2} - (1-\gamma)}{\lambda D(1-\gamma)}. \quad (8)$$

When γ equals zero, clients do not participate in the sending of full tap streams and (5) reverts to

$$\beta_{\text{opt}} = \frac{\sqrt{2\lambda D + 1} - 1}{\lambda D}. \quad (9)$$

When γ equals 1, clients handle all full tap streams, the server only handles the complete streams, and β_{opt} goes to infinity As a result, the server will never start a new complete stream before the end of the previous one, and the server bandwidth $B_{s,\text{ST}}$ will never exceed one channel.

To estimate the total bandwidth requirements of the clients when $\gamma = 1$, let us consider, let us consider an interval of duration $2D$ starting with the beginning of a new complete stream During this time interval, the system will receive

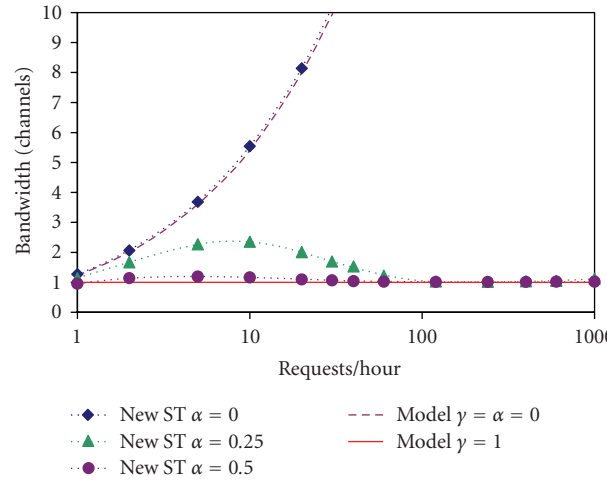


Figure 8: Server bandwidth requirements of the new stream tapping protocol.

an average of $2\lambda D$ requests in addition to the request that prompted the complete stream and the clients will send an average of $2\lambda D$ full tap streams. Since the lengths of these full tap streams will be uniformly distributed over the interval $(0, D)$, the average duration of each of these streams will be equal to $D/2$ The total duration of these streams will thus be

$$W_c = (2\lambda D)\left(\frac{D}{2}\right) = \lambda D^2. \quad (10)$$

The average bandwidth $B_{c,\text{ST}}$ used by the clients to send full taps streams is then given by dividing this expression by the duration of the considered interval,

$$B_{c,\text{ST}} = \frac{W_c}{2D} = \frac{\lambda D}{2}. \quad (11)$$

The total network bandwidth requirements B_{ST} of our protocol are obtained by adding the average bandwidth $B_{s,\text{ST}}$ used by the server to that expression As we can see in Figure 4, $B_{s,\text{ST}}$ is very close to on for large values of λ. We can thus approximate B_{ST} as

$$B_{\text{ST}} = B_{c,\text{ST}} + B_{s,\text{ST}} \approx \frac{\lambda D}{2} + 1. \quad (12)$$

Compare these requirements with those of a video distribution scheme that does not use any form of multicast Over an interval of duration T, it will handle an average of λT requests. Since each request will be serviced by a separate stream of duration D, the total duration of these streams will be λDT, and the average bandwidth requirements of the scheme would be λD, that is, almost twice the average bandwidth B_{ST} requirements of our cooperative stream tapping protocol.

Figures 8 and 9 compare our analytical results with our simulation results We considered the two limit cases when $\gamma = 0$ and $\gamma = 1$ As we can see, both models predict identical network bandwidths for all values of α and all arrival rates.

This is not the case for server bandwidths Both models are in substantial agreement when either $\alpha = 0$ or $\alpha \geq 0.5$ and disagree when $0 < \alpha < 0.5$ Let us consider these three cases.

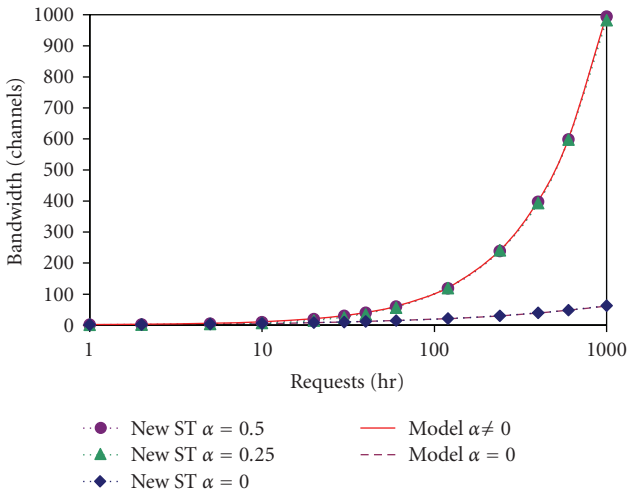


Figure 9: Network bandwidth requirements of the new stream tapping protocol.

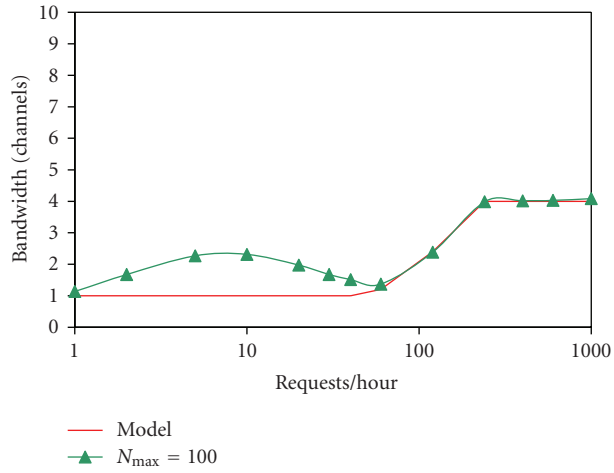


Figure 10: Server bandwidth requirements of the protocol for $\alpha = 0.25$, $\beta_{max} = 0.25$, and $N_{max} = 100$.

(1) $\alpha = 0$

When $\alpha = 0$, the clients cannot upload video data Hence, $\gamma = 0$ and both models predict identical network bandwidths.

(2) $\alpha \geq 0.5$

When $\alpha \geq 0.5$, the clients can upload video data at at least half the video consumption rate. As a result, they can handle most, if not all, of the tap streams, and their share γ of the stream tapping data is close to one. This is less true for request arrival rates below 20 requests per hour, where the server must still handle a small fraction of the tap streams.

(3) $0 < \alpha < 0.5$

When $0 < \alpha < 0.5$, the clients have very limited upload bandwidths They participate in the distribution of tap streams but cannot fully substitute for the server as long as the request arrival rate remains below 60 requests per hour As a result, $0 < \gamma < 1$ and the actual server bandwidth stands somewhere between the values predicted for $\gamma = 0$ and $\gamma = 1$.

Let us turn now our attention to the performance of the scheme reducing the network bandwidth consumption of our protocol at high arrival rates Recall that this scheme consisted of starting a new complete stream whenever (a) the size of the tap stream would otherwise exceed a fraction β_{max} of the duration of the video, and (b) more than N_{max} requests were already sharing the current complete stream.

On the average, this scheme will take effect whenever the request arrival rate λ will exceed a threshold,

$$\lambda^* = \frac{N_{max}}{\beta_{max} D}. \tag{13}$$

At higher arrival rates, the server will restart a new complete stream of duration each $\beta_{max} D$ time units Its bandwidth requirements will thus be equal to $1/\beta_{max}$ As a result, the average bandwidth requirements of the server will be

$$B_{s,ST} = \begin{cases} 1 & \text{for } \lambda < \lambda^*, \\ \dfrac{1}{\beta_{max}} & \text{for } \lambda \geq \lambda^*. \end{cases} \tag{14}$$

Let us now focus on the behavior of clients at arrival rates greater than or equal to λ_{max} Assuming again that the clients will handle all full tap streams, they will send an average of $\lambda \beta_{max} D$ full tap streams each $\beta_{max} D$ time units The average length of these tap streams will be equal to $\beta_{max} D/2$, and the average bandwidth $B_{c,ST}$ used by the client will be

$$B_{c,ST} = \begin{cases} \dfrac{\lambda D}{2} & \text{for } \lambda < \lambda^*, \\ \dfrac{\lambda \beta_{max} D}{2} & \text{for } \lambda \geq \lambda^*. \end{cases} \tag{15}$$

Figures 10 and 11 compare our analytical results with our simulation results for $\alpha = 0.25$, $\beta_{max} = 0.25$, and $N_{max} = 100$ As before, both models predict identical or near identical network bandwidths for all values of α and all arrival rates This is not the case for the server bandwidth Here again our analytical model underestimates the server bandwidth when the request arrival rate remains below 60 requests per hour because it incorrectly assumes that the clients can handle all tap streams.

We can thus say that the results of our probabilistic analysis confirm those obtained through our simulation study and conclude that our cooperative stream tapping protocol can effectively harness the collective upload bandwidth of its clients even when each individual client cannot upload video data at more than one quarter of the video consumption In addition, requiring the server to restart complete streams at fixed intervals provides an effective tool for limiting the network bandwidth consumption of the protocol.

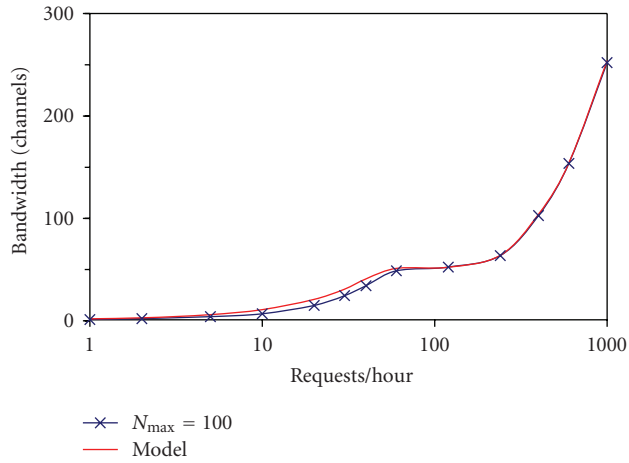


Figure 11: Network bandwidth requirements of the protocol for $\alpha = 0.25$, $\beta_{max} = 0.25$, and $N_{max} = 100$.

6. PROTOCOL APPLICABILITY

Many organizations such as universities and companies run their own multicast-capable networks Consider the example of a local telephone company that decides to increase its service offerings to include on-demand video entertainment Many of their existing clients are already subscribed to their high-speed internet access by means of DSL With our solution, the telephone company can make use of their multicast-enabled infrastructure to deliver videos on demand to a large number of clients at very low cost.

7. CONCLUSIONS

We have presented a stream tapping protocol that involves clients in the video distribution process Our protocol is tailored to multicast-capable environments where client machines are able to download video data at twice the video consumption rate but cannot necessarily forward video data at that rate. We observed that our technique achieved a dramatic reduction of the server workload at medium to high request arrival rates but also resulted in much higher network bandwidth consumptions. These increases can however be controlled by requiring the server to restart complete streams at some specific intervals Even then, the server bandwidth requirements of the protocol remain well below those of pure client-server solutions.

Our proposal differs from conventional peer-to-peer (P2P) solutions in two different ways First, it assumes the existence of one or more servers capable of multicasting video data at twice to four times the video consumption rate This allows the protocol to make much more efficient use of the network bandwidth as a single video stream can be broadcast to an arbitrary number of clients Second, it does not require peers to be able to upload video data at more than one quarter to one half of their video consumption rates, which would not be possible in a pure P2P solution such as [18] In pure P2P systems, the download rate is positively correlated to the upload rate Legout et al. [19] illustrated through experiments that the amount of data uploaded is very close to the data downloaded in P2P systems such as BitTorrent [20]. Thus, to sustain a video streaming service, a peer should have an upload rate at least equal the video consumption rate.

Stream tapping protocol involving clients is a very good solution, especially over local networks as it provides a scalable way to distribute on-demand high-bandwidth videos streams without overtaxing the network bandwidth.

REFERENCES

[1] A. Dan, P. Shahabuddin, D. Sitaram, and D. Towsley, Channel allocation under batching and VCR control in video-on-demand systems, *Journal of Parallel and Distributed Computing*, vol. 30, no. 2, pp. 168179, 1995.

[2] L. Golubchik, J. C. S. Lui, and R. R. Muntz, Adaptive piggybacking: a novel technique for data sharing in video-on-demand storage servers, *Multimedia Systems*, vol. 4, no. 3, pp. 140155, 1996.

[3] S. Sheu, K. A. Hua, and W. Tavanapong, Chaining: a generalized batching technique for video-on-demand systems, in *Proceedings of the IEEE International Conference on Multimedia Computing and Systems (ICMCS '97)*, pp. 110117, Ottawa, Ontario, Canada, June 1997.

[4] T.-C. Su, S.-Y. Huang, C.-L. Chan, and J.-S. Wang, Optimal chaining scheme for video-on-demand applications on collaborative networks, *IEEE Transactions on Multimedia*, vol. 7, no. 5, pp. 972980, 2005.

[5] J.-F. Pâris, A cooperative distribution protocol for video-on-demand, in *Proceedings of the 6th Mexican International Conference on Computer Science (ENC '05)*, pp. 240246, Puebla, Mexico, September 2005.

[6] S. W. Carter and D. D. E. Long, Improving video-on-demand server efficiency through stream tapping, in *Proceedings of the 6th International Conference on Computer Communications and Networks (ICCCN '97)*, pp. 200207, Las Vegas, Nev, USA, September 1997.

[7] S. W. Carter and D. D. E. Long, Improving bandwidth efficiency of video-on-demand servers, *Computer Networks*, vol. 31, no. 1-2, pp. 111123, 1999.

[8] K. A. Hua, Y. Cai, and S. Sheu, Patching: a multicast technique for true video-on-demand services, in *Proceedings of the 6th ACM International Conference on Multimedia*, pp. 191200, Bristol, UK, September 1998.

[9] D. L. Eager and M. K. Vernon, Dynamic skyscraper broadcast for video-on-demand, in *Proceedings of the 4th International Workshop on Advances in Multimedia Information Systems*, pp. 1832, Istanbul, Turkey, September 1998.

[10] K. A. Hua and S. Sheu, Skyscraper broadcasting: a new broadcasting scheme for metropolitan video-on-demand systems, in *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM '97)*, pp. 89100, Cannes, France, September 1997.

[11] K. A. Hua and S. Sheu, An efficient periodic broadcast technique for digital video libraries, *Multimedia Tools and Applications*, vol. 10, no. 2-3, pp. 157177, 2000.

[12] D. L. Eager, M. K. Vernon, and J. Zahorjan, Minimizing bandwidth requirements for on-demand data delivery, in *Proceedings of the 5th International Workshop on Advances in Multimedia Information Systems*, Indian Wells, Calif, USA, October 1999.

[13] L. Gao, Z.-L. Zhang, and D. Towsley, Catching and selective catching: efficient latency reduction techniques for delivering continuous multimedia streams, in *Proceedings of the 7th ACM International Conference on Multimedia*, pp. 203206, Orlando, Fla, USA, October 1999.

[14] S. Kulkarni and J.-F. Pâris, Involving clients in the distribution of videos on demand, in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME '06)*, pp. 16771680, Toronto, Ontario, Canada, July 2006.

[15] J.-F. Pâris, A stream tapping protocol with partial preloading, in *Proceedings of the 9th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS '01)*, pp. 423430, Cincinnati, Ohio, USA, August 2001.

[16] J.-F Pâris, Using available client bandwidth to reduce the distribution costs of video-on-demand services, in *Proceedings of the 7th Workshop on Distributed Data and Structures (WDAS '06)*, Santa Clara, Calif, USA, January 2006.

[17] J.-F. Pâris and D. D. E. Long, An analytic study of stream tapping protocols, in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME '06)*, pp. 12371240, Toronto, Ontario, Canada, July 2006.

[18] P. Shah and J.-F. Pâris, Peer-to-peer multimedia streaming using BitTorrent, in *Proceedings of the 26th IEEE International Performance, Computing, and Communications Conference (IPCCC '07)*, pp. 340347, New Orleans, La, USA, April 2007.

[19] A. Legout, G. Urvoy-Keller, and P. Michiardi, Understanding BitTorrent: an experimental perspective, *Tech. Rep. inria-00000156*, INRIA, Sophia Antipolis, France, 2005.

[20] B. Cohen, Incentive-build robustness in BitTorrent, in *Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems*, Berkeley, Calif, USA, June 2003.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2008, Article ID 624941, 15 pages
doi:10.1155/2008/624941

Research Article

Automatic Bandwidth Adjustment for Content Distribution in MPLS Networks

D. Moltchanov

Institute of Communication Engineering, Tampere University of Technology, P.O. Box 553, 33101 Tampere, Finland

Correspondence should be addressed to D. Moltchanov, moltchan@cs.tut.fi

Received 2 May 2007; Revised 19 September 2007; Accepted 15 January 2008

Recommended by Zhu Li

Aggregates of real-time traffic may experience changes in their statistical characteristics often manifesting non stationary behavior. In multi protocol label switching (MPLS) networks this type of the traffic is assigned constant amount of resources. This may result in ineffective usage of resources when the load is below than expected or inappropriate performance when the load is higher. In this paper we propose new algorithm for dynamic resource adaptation to temporarily changing traffic conditions. Assuming that network nodes may reallocate resources on-demand using automatic bandwidth adjustment capability of MPLS framework, the proposed algorithm, implemented at ingress MPLS nodes, dynamically decides which amount of resources is currently sufficient to handle arriving traffic with given performance metrics. This decision is then communicated to interior MPLS nodes along the label switched path. As a basic tool of the algorithm we use change-point statistical test that signals time instants at which statistical characteristics of traffic aggregates change. The major advantage of the proposed approach is that it is fully autonomous, that is, network nodes do not need any support from hosts in terms of resource reservation requests. The proposed algorithm is well suited for traffic patterns experiencing high variability, especially, for non stationary type of the traffic.

1. INTRODUCTION

Congestion is one of the major problems that degrades performance of networks. It occurs when network resources are insufficient for a current load or when load distribution is uncontrolled. The latter was the main problem of IP networks back in 1990s. Existing interior gateway protocols (IGPs) were topology-driven, took decision on routing packets based on network connectivity only and did not take into account resource requirements of traffic patterns. Traffic engineering (TE) capabilities of MPLS framework are intended to avoid the situation where congestion occurs as the result of inefficient resource allocation. TE attempts to provide best possible utilization of network resources avoiding noneven load distribution. Nowadays, MPLS is considered the vital part of QoS-aware networks.

Traffic patterns in packet-switched networks are characterized by deterministic trends similar to those found in classic telephone traffic. Particularly, it was demonstrated that there are clear daily variations in statistical parameters of traffic aggregates and these variations can be specific for different services [1]. They are often caused by preference of users to use a certain network service in a specific time of the day. Authors in [1] also noticed that there is a clear indication of the busy hour in link usage patterns. Irregular changes in statistical parameters of traffic aggregates were also reported in [2–4]. In [5], authors highlighted that operators' pricing scheme may also produce changes in traffic patterns resulting in additional source of nonstationarity. Besides commonly believed self-similar and long-range dependent properties (see [6–8] among others), nonstationarity is one of the major reasons for a traffic pattern to exhibit high variability.

In modern QoS-aware networks, resources for traffic aggregates are often allocated statically based on busy hour traffic expectations. In this case, high variability of a traffic pattern requires significant overprovisioning of network resources. Practically, high variability implies that there are large bursts in a traffic pattern. To serve such traffic with given performance metrics, sufficient amount of resources must be provided during this period. However, there are also

long time spans during which the local average of the traffic pattern stays well below the global average. This implies that for a traffic pattern exhibiting high variability static resource allocation leads to ineffective usage of network resources. We also note that the busy hour traffic volume may not be known in advance making the decision about the required resources completely unclear.

The network always has up-to-date view of traffic aggregates. The straightforward way to deal with overprovisioning or underprovisioning of resources is to allow network nodes to dynamically choose the amount of resources that is sufficient for current arrival statistics of a traffic aggregate. The rest of resources can be temporarily assigned to other traffic aggregates. To deal with this problem, MPLS networks provide automatic bandwidth adjustment capability. It allows an MPLS label switched path (LSP) to automatically adjust bandwidth allocation based on its traffic load measured over the sampling interval of a certain length. Unfortunately, automatic bandwidth adjustment procedure does not take into account that statistical characteristics of traffic patterns may vary even during a single sampling interval. The length of the sampling interval is another important issue that affects performance of the resource allocation and requires additional administrative efforts. Dealing with time-varying traffic patterns, fully automatic bandwidth adjustment scheme that requires no additional administrative effort would be of a great benefit for network operators.

In this paper, we propose resource description and allocation scheme that dynamically reallocate resources to traffic aggregates in MPLS networks. Assuming that network nodes may reallocate resources on-demand using automatic bandwidth adjustment capability of MPLS framework, the proposed algorithm decides which amount of resources is currently sufficient to handle arriving traffic with given performance guarantees. We describe the structure of the proposed system and demonstrate its applicability using real traffic traces. As a basic tool of the algorithm, we use change-point statistical test. It allows to automatically determine whether statistical characteristics of a traffic pattern change and, if so, estimate new parameters of the traffic pattern. The major advantage of the proposed approach is that it is fully autonomous, that is, the network does not need any support in terms of explicit resource reservation requests. The proposed algorithm is especially wellsuited for nonstationary type of the traffic whose statistical parameters change in time. Presented numerical results reveal that the proposed approach may provide significant resource savings.

The rest of the paper is organized as follows. In Section 2, we briefly review automatic bandwidth allocation capability of MPLS networks. In Section 3, we consider dynamic nature of aggregated video traffic. Model for covariance-stationary behavior of aggregated video traffic is also proposed there. Change-point statistical test for detecting changes in arrival statistics is introduced in Section 4. The dynamic resource allocation system is proposed in Section 5. Numerical examples are given in Section 6. Conclusions are drawn in the last section.

2. MPLS NETWORKS

2.1. Traffic engineering in MPLS

Modern QoS-aware networks such as DiffServ, MPLS, or joint DiffServ/MPLS are specifically designed to be flexible enough to reallocate network resources in the best possible way, such that the required performance is provided using minimum amount of resources. Although MPLS is not considered a QoS framework for IP networks, it provides a number of advantageous features to network operators. According to MPLS, data are transmitted along the so-called label switched paths (LSPs) that can be explicitly chosen by network operators [9]. For this purpose, MPLS assigns short labels to network packets that describe how to forward them through the network. This feature enables effective traffic engineering (TE) capabilities [10] compared to classic approach that adopts usage of interior gateway protocol (IGP) metrics. Label switched paths (LSPs) are computed in traffic engineering databases (TED, [10]) in ingress routers that are filled by IGP advertisements. For this reason, IGP protocols were extended to support advanced metrics such as available bandwidth on a link [11, 12]. LSPs are then established using either resource reservation protocol modified to support traffic engineering (RSVP-TE, [13]) or label distribution protocol with constrained routing (CR-LDP, [14]).

Traffic engineering capabilities of MPLS allow to control the path that data packets follow in the network avoiding standard routing strategy of IGP protocols. They are intended to deal with the situation where congestion occurs as the result of inefficient resource allocation. TE tries to provide best possible utilization of network resources avoiding noneven load distribution. This is achieved by real-time monitoring of the traffic load on each network element and further tuning of traffic management attributes, routing parameters, and resources constraints.

2.2. Automatic bandwidth adjustment

Traffic aggregates in DiffServ, MPLS, and joint MPLS/DiffServ networks are still described statically using the token bucket mechanism. Automatic bandwidth adjustment capability of MPLS networks allows an LSP to automatically adjust its bandwidth allocation based on the measured traffic load. It works as follows. During the sampling interval of a certain length, the average bandwidth is monitored. At the end of the interval, an attempt is made to signal a new LSP with the bandwidth allocation set to the average value for the preceding sampling interval. If a new path is established, the original path is removed and the LSP is switched to the new path. If a new path is not established, the LSP continues to use its current path until the end of the next sampling interval, when another attempt is made.

While the approach described above is considered to be automatic in nature, it does not take into account possible time-varying behavior of traffic patterns. When the sampling interval is too large, there can be multiple changes in the mean value of traffic observations that may severely

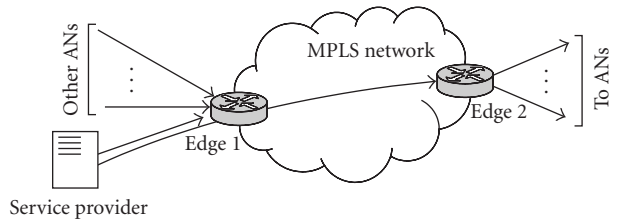


Figure 1: Reference configuration of the service.

bias the statistics. On the other hand, too short sampling interval may burden the network with excessive signaling information. The approach that triggers LSP reestablishment when statistical characteristics of a traffic patterns do change could provide trade-off between the amount of signaling information and accuracy of bandwidth allocation.

2.3. Service configuration

Reference configuration of the service we consider is presented in Figure 1, where AN stands for access network. We assume that there is a network operator providing network resources for a service provider. Service provider provides a QoS-constrained video distribution service to a number of users across the network. We assume that there is a specifically established LSP for video traffic aggregate and routers along the path of the traffic aggregate implement MPLS framework with dynamic bandwidth adjustment capability. The network operator is required to provide guarantees to this traffic aggregate in terms of mean losses and delays.

3. TIME-VARYING NATURE OF AGGREGATED VIDEO TRAFFIC

To represent traffic arrival process, we use video traffic traces available from the University of Berlin [15]. We consider video traffic using the granularity of a second. Although we use H.263 VBR traces, we have checked that conclusions stated in this paper remain valid for MPEG-4 VBR traces from the same traffic archive and MPEG-1 VBR sequences from University of Wurzburg [16].

3.1. Traffic aggregate: varying number of sources

We generated traffic traces simulating behavior of the video distribution system as explained below. We assume that at a certain instant of time t_0 there are no active sessions and this is the time instant when the system enters its operational state. After t_0, session requests start to arrive to the system. Each arrival requests an arbitrary video sequence. We used uniform distribution over all available traces to determine the requested sequence. Interarrival times of sessions are geometrically distributed with a certain probability P. We assumed that session arrival process is time-varying meaning that P varies in time. This is a natural assumption for any service where the session arrival rate is different for different time of the day. We simulated 50 session arrivals. First 15 and last 15 arrivals were drawn from geometrical distribution with $P = .001$, 20 arrivals in the middle occur according to geometrical distribution with $P = .005$. Note that these parameters allow to mimic behavior of the system prior, during and after the busy hour.

Time-series of generated traces are shown in Figure 2. Observing these traces, it is natural to expect that statistics of the aggregated traffic are time-varying. Indeed, when the rate of session arrivals varies in time, it is unrealistic to assume that aggregated traffic may converge to covariance stationary process except for some segments during which the number of active sessions remains constant. For other choices of P and different number of generated sessions, traces look qualitatively similar meaning that their statistical parameters do vary in time. Note that statistical characteristics of traffic aggregates may also vary due to other phenomena including long-range dependent, self-similar or nonstationary properties of individual streams. In this paper, the actual cause of variability is of no interest as long as change detection mechanism is able to detect points at which changes occur and subsequently estimate current traffic statistics.

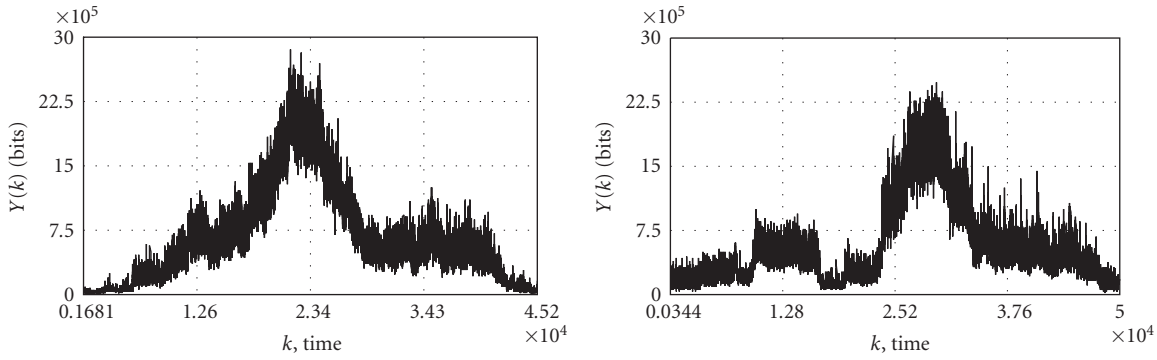
An important property of aggregated traffic is that changes in the mean value lead to changes in the variance and vice versa. Figure 3 highlights this behavior. As one can observe, this dependency is almost linear allowing to track changes in one parameter only, either mean or variance.

3.2. Traffic aggregate: fixed number of sources

Consider statistical characteristics of the aggregated traffic when the number of multiplexed sources is kept constant. To obtain aggregated traffic patterns, we arbitrarily choose 9 traces out of the archive and multiplex them synchronizing starting times of all traces. We limited all traces to 1200 seconds.

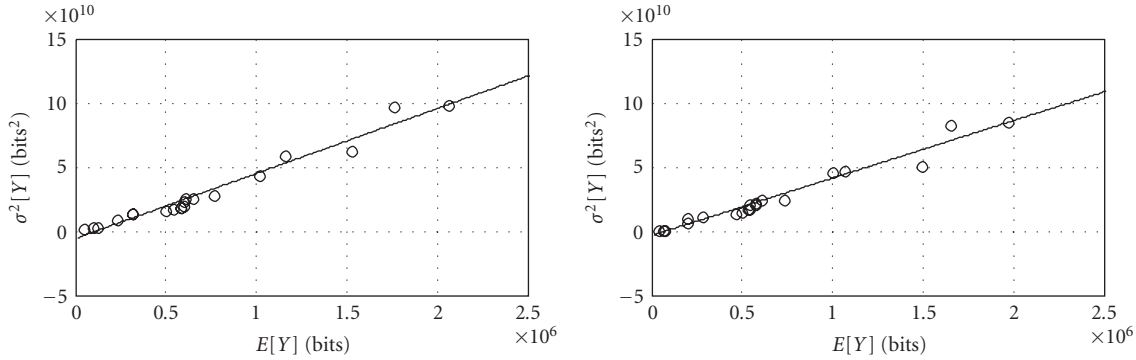
Time-series of six aggregated traces are shown in Figure 4. Observing these traces, one may suggest that statistics of the aggregated traffic may not significantly vary in time. Indeed, when the number of video traffic sources is fixed and relatively large, it is straightforward to assume that the aggregated traffic may converge to covariance stationary process with normal distribution [5].

To support our hypothesis about covariance stationarity, we carried out a number of tests. First, we divided each trace into 6 nonoverlapping segments each of which contains exactly 200 observations. Mean values of these segments are shown in Figure 5. One may see that the mean value of single traces does not change in time significantly. For all traces, the ranges of mean values are less than 20% of global means of respective traces. These variations can be attributed to positive autocorrelation found in these traces. Finally, we tested our traces for homogeneity. To do so, each trace was divided into two samples having the same number of observations. In our case, this corresponds to 600 observations. Then, we applied χ^2 test for homogeneity of two samples. This test allows to check hypothesis H_0 that two parts have the same distribution against alternative hypothesis, H_1, that they have different distributions. Performed tests revealed that with



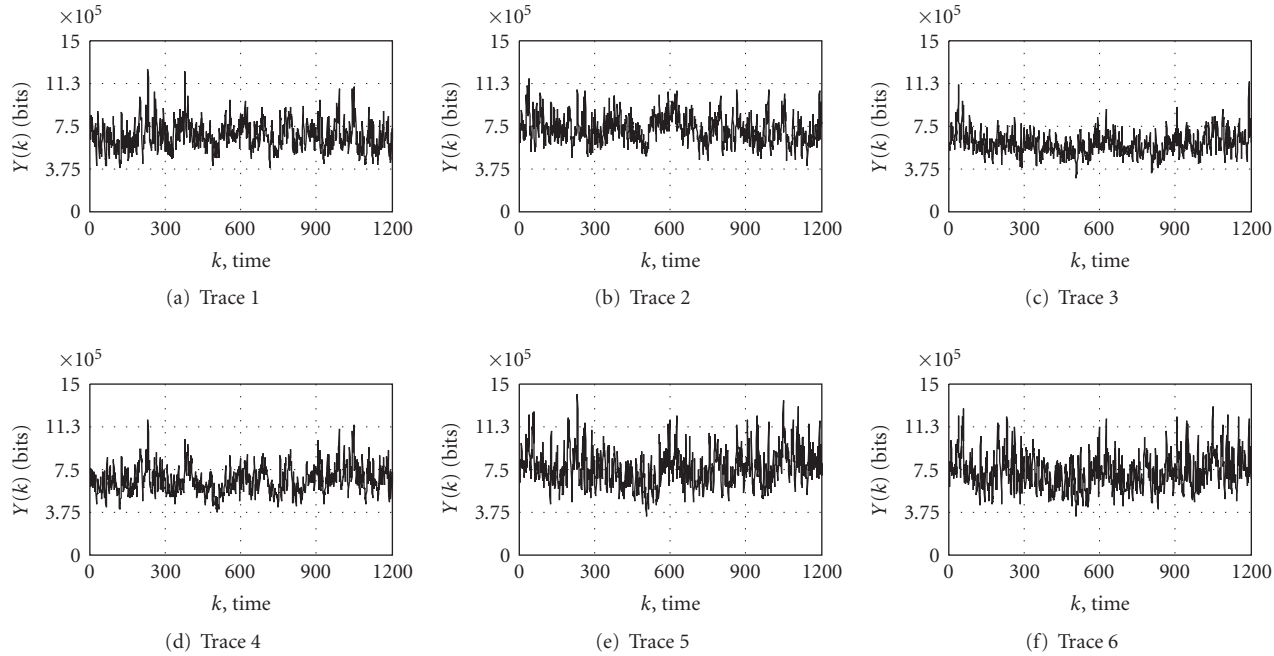
(a) Trace 1 (b) Trace 2

Figure 2: Aggregated traffic from varying number of sources.



(a) Trace 1 (b) Trace 2

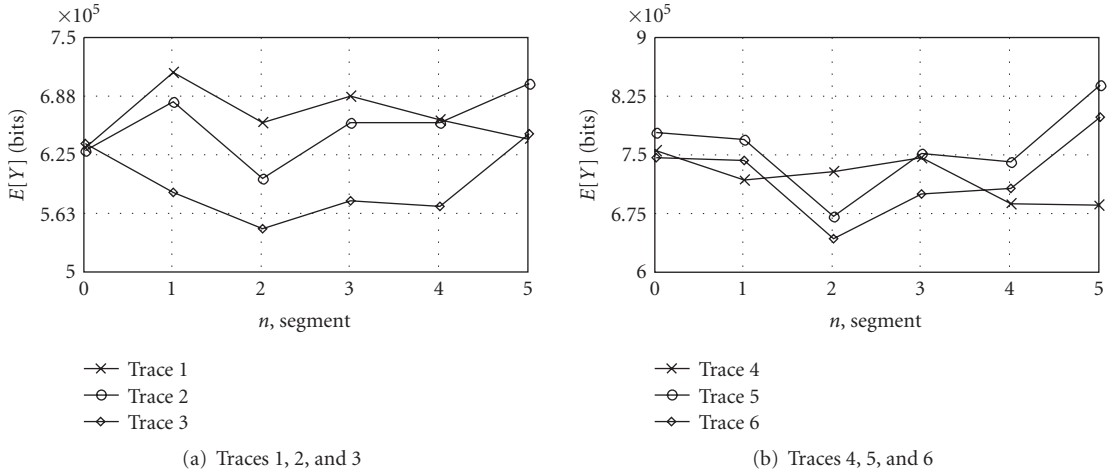
Figure 3: Dependence between mean and variance of aggregated traffic.



(a) Trace 1 (b) Trace 2 (c) Trace 3

(d) Trace 4 (e) Trace 5 (f) Trace 6

Figure 4: Aggregated traffic traces from a fixed number of sources.



(a) Traces 1, 2, and 3

(b) Traces 4, 5, and 6

Figure 5: Mean values of segments.

level of significance $\alpha = 0.1$ two parts of the same trace have the same distribution.

Histograms of relative frequencies ofconsidered traces and their approximations by normaldistributions are shown in Figure 6, where $f_{i,Y}(\Delta)$, $i = 1, 2, \ldots$ is the relative frequency corresponding to is bin, $\Delta = (\max_{\forall i} Y(i) - \min_{\forall i} Y(i))/m$ is the width of the histogram bin. These approximations suggest that aggregated traffic from fixed number of sources is normally distributed. The χ^2 goodness-of-fit test performed with the level of significance $\alpha = 0.1$ confirmed this hypothesis.

Empirical normalized autocorrelation functions (NACFs) for all traces are shown in Figure 7. One may note that the memory of the process is short and limited to few lags. We approximate this behavior using a single geometrical term, for example, $y(i) = K_Y(1)^i$, $i = 0, 1, \ldots$. This approximation exactly captures lag-1 values of NACFs of empirical processes and does not significantly overestimate or underestimate autocorrelation coefficients for larger lags.

Presented results allow us to assume that when the number of multiplexed sources remains constant the aggregated video traffic composes realization of covariance stationary stochastic process. We note that our tests do not allow us to be statistically strict regarding covariance stationarity. Unfortunately, there are no universal and effective methods to statistically test whether given observations are stationary or not. However, Carried out tests confirm that the marginal distribution is normal and depends on the number of multiplexed traces only NACF tends to zero as time progresses confirming that the process is ergodic. All these conditions are necessary for an underlying process to be covariance stationary. In Section 6, we carry out another test that also confirms our hypothesis.

3.3. Model for aggregated traffic

To model aggregated traffic from fixed number of sources, we use covariance stationary autoregressive process of order one, AR(1). This process has normal marginal distribution and geometrically decaying ACF allowing us to suggest that it may produce fair approximation of empirical data [17]. These two properties of traffic patterns were found to produce the major impact on their service performance in a network (see [18, 19] among others).

A process is said to be autoregressive of order one if it is given by

$$X(n) = \phi_0 + \phi_1 X(n-1) + \epsilon(n), \quad n = 0, 1, \ldots, \tag{1}$$

where ϕ_0 and ϕ_1 are some constants, $\{\epsilon(n), n = 0, 1, \ldots\}$ are independently and identically distributed random variables having the same normal distribution with zero mean and variance $\sigma^2[\epsilon]$.

For (1) to be weakly stationary, it is sufficient to have $\phi_1 \neq 1$. In this case,

$$E[X(n)] = \mu_X, \qquad \mathrm{Cov}\big(X_0(n), X_i(n+i)\big) = \gamma_X(i), \tag{2}$$

where μ_X, $\gamma_X(i)$, $i = 0, 1, \ldots$ are some constants.

Mean, variance and covariance of AR(1) are related to ϕ_0, ϕ_1, and $\sigma^2[\epsilon]$ as

$$\mu_X = \frac{\phi_0}{1-\phi_1}, \quad \sigma^2[X] = \frac{\sigma^2[\epsilon]}{1-\phi_1^2}, \quad \gamma_X(i) = \phi_1^i \gamma_X(0). \tag{3}$$

Parameters of AR(1) models are related to statistical data as

$$\phi_1 = K_Y(1), \quad \phi_0 = \mu_Y(1-\phi_1), \quad \sigma^2[\epsilon] = \sigma^2[Y](1-\phi_1^2), \tag{4}$$

where $K_Y(1)$, μ_Y, and $\sigma^2[Y]$ are estimates of lag-1 autocorrelation coefficient, mean, and variance, respectively.

4. CHANGE-POINT STATISTICAL TESTS

4.1. Basic principles

To differentiate between fluctuations of the aggregated traffic around a constant mean and those variations caused by

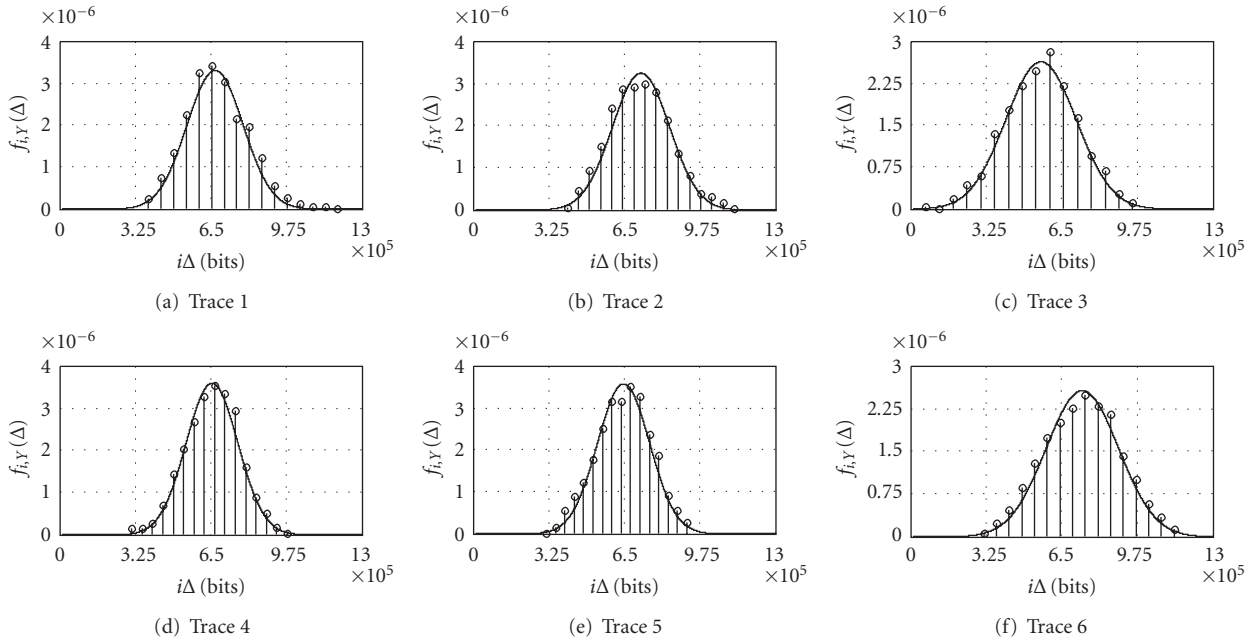


Figure 6: Histograms and their approximations by normal distribution.

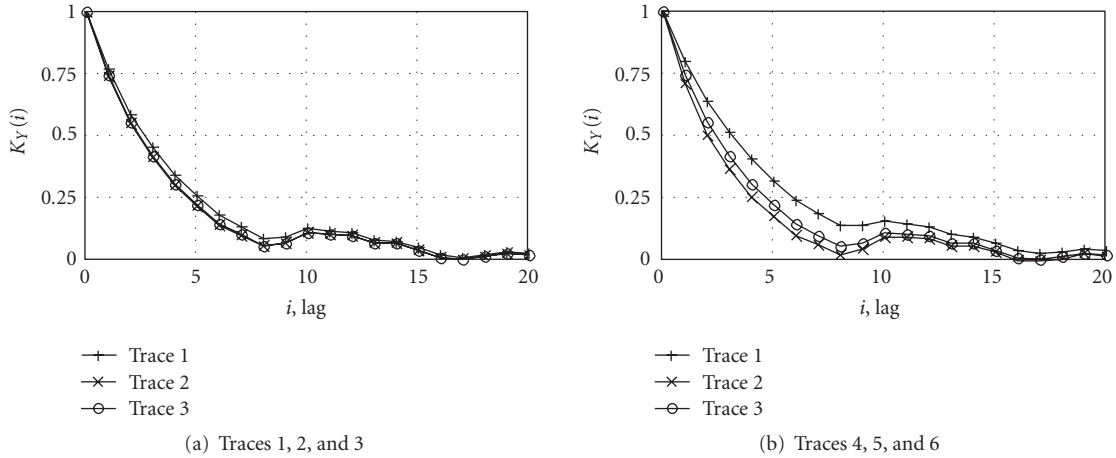


Figure 7: Normalized autocorrelation functions of traces.

changes in the mean value, we propose to use change-point statistical tests. There are a number of change-point detection algorithms developed to date. The common approach to deal with this task is to use control charts including Shewhart, cumulative sum (CUSUM), or exponentially-weighted moving average (EWMA) charts. The idea of control charts is to classify all causes of deviation from a target value into two groups. These are common causes and special causes. Deviation due to common causes is the effect of inherent causes affecting a given process. Special causes are not the part of the process, occur accidentally, and affect the process significantly. Control charts signal the point at which special causes occur using two control limits. If values of a certain function of initial observations are between them, process is in control. If some value falls outside, the process is out of control.

For detecting changes in aggregated traffic, we assume the following. We consider that common causes of deviation are those resulting in inherent stochastic nature of the multiplexed traffic from fixed number of video sources. Special causes are all those causing changes in parameters of in control processes. For example, among those are arrivals of new sessions adding new traffic and completions of ongoing ones subtracting some traffic. The control procedure is as follows. Initially, a control chart is parameterized using estimates of parameters of the aggregated traffic process. When change in a parameter occurs, a new process is considered as in control and the control chart is reparameterized according to statistics of this process.

Change-point tests often require observations to be independent. We have seen that aggregated video traffic is characterized by positive correlation. Autocorrelation makes

control charts less sensitive to changes in the mean value. For detecting changes in the mean value of autocorrelated processes, two approaches have been proposed. According to the first approach, control limits of charts are modified to take into account autocorrelation properties. The idea of the second approach is to fit observations using a certain time-series model and subsequently test residuals. If the model fits empirical data well, residuals are uncorrelated and control charts for independent observations can be used. Performance of change-point statistical tests for autocorrelated data has been compared in [20, 21]. It was shown that modified control charts on initial observations perform better when autocorrelation is positive. Due to this reason, we use the first approach.

4.2. EWMA control charts

Let $\{Y(n), n = 0, 1, \ldots\}$ be a sequence of initial observations. The value of EWMA statistic at the time n, denoted by $L_Y(n)$, is given by

$$L_Y(n) = \gamma Y(n) + (1 - \gamma)L_Y(n - 1), \tag{5}$$

where parameter $\gamma \in (0, 1)$ is constant.

The reason to use EWMA statistical test is that it takes central part among other control charts. Although, according to (5), the most recent value always receives more weight in computation of $L_Y(n)$, the choice of γ determines the effect of previous observations of the process on the current value of EWMA statistics. Indeed, when $\gamma \to 1$ all weight is placed on the current observation, $L_Y(n) \to Y(n)$, and EWMA statistics degenerate to initial observations. As a result, EWMA control chart behaves like Shewhart one [22]. On the other hand, when $\gamma \to 0$ the current observation gets only a little weight, but most weight is assigned to previous observations. In this case, EWMA control chart behaves similar to CUSUM one [23]. Summarizing, EWMA charts provide more flexibility at the expense of additional complexity in determining one more parameter γ. Due to inherent flexibility, we use EWMA control charts.

Assume that initial observations $\{Y(n), n = 0, 1, \ldots, N\}$ are taken from covariance stationary process with mean $E[Y]$ and variance $\sigma^2[Y]$ and can be well represented by AR(1) process. If $L_Y(0) = E[Y]$, it is easy to see that $E[L_Y] = E[Y] = \mu_Y$ when $n \to \infty$. The approximation for variance of $\{L_Y(n), n = 0, 1, \ldots\}$ is given by [20]

$$\sigma^2[L_Y] = \sigma^2[Y]\left(\frac{\gamma}{2 - \gamma}\right)\left(\frac{1 + \phi_1(1 - \gamma)}{1 - \phi_1(1 - \gamma)}\right), \tag{6}$$

where ϕ_1 is the parameter of AR(1) process.

The control limits $E[L_Y] \pm C_Y(n)$ are given by

$$E[L_Y] \pm C_Y(n) = E[L_Y] \pm k\sigma[Y]\sqrt{\left(\frac{\gamma}{2 - \gamma}\right)\left(\frac{1 + \phi_1(1 - \gamma)}{1 - \phi_1(1 - \gamma)}\right)}, \tag{7}$$

where k is a design parameter whose values are tabulated.

4.3. Parametrization of EWMA charts

To parameterize the EWMA control chart, a number of parameters have to be provided. Firstly, parameter γ determining the decline of weights of past observations should be set. The values of k and γ determine the wideness of control belts for a given process with certain $\sigma^2[Y]$ and μ_Y. These four parameters affect behavior of the so-called average run length (ARL) value that is used to determine efficiency of a certain change detection procedure. ARL is defined as the average number of observation up to the first out of control signal. There are a number of methods to compute ARL value for given γ and k. Tabulated values of in control ARL can be found in [20, 21]. Finally, μ_Y and $\sigma^2[Y]$ are not usually known in practice and must be estimated from empirical data. Therefore, estimates of μ_Y and $\sigma^2[Y]$ should be used in (7). As a result, for real-time implementation of EWMA test, there should always be a certain warmup period involving estimation of the mean. Finally, the first value of EWMA statistics is usually set to the global mean of observations or, if unknown, to the estimate of mean.

5. RESOURCE ALLOCATION SYSTEM

5.1. Functionality of the system

The proposed resource description and allocation system should be implemented in ingress MPLS routers. We assume that ingress routers conform to MPLS specifications. The only important difference compared to MPLS functionality is the resource controller. The purpose of this controller is to monitor arriving traffic of different LSPs for possible changes in its statistical characteristics and manage the resource allocation for a given behavior aggregate. Another responsibility of the controller is to estimate the amount of resources required to serve incoming traffic with given performance metrics. It is important that this controller does not change the traffic pattern allowing it to proceed unaltered to the output port. We also note that no changes to interior nodes are required and they must strictly follow MPLS specifications.

The structure ingress node is shown in Figure 8. It is intended to operate as follows. Traffic policing unit is statically parameterized such that the highest possible load a content provider intends to pay for is allowed to enter the network. Those packets that conform to this specification proceed to the resource controller. This element monitors conforming traffic for possible changes in its statistical characteristics and estimates the amount of resources required to serve it with given performance metrics. Performance metrics must be configured manually. When change in traffic statistics occurs, resource controller estimates new resource allocation and advertises it to the resource allocation mechanism at the output port and to the MPLS TED associated with ingress node. RSVP-TE is then used to update resource allocation at all nodes along the path of LSP.

The structure of the resource controller is shown in Figure 9. Two major components of this system are the

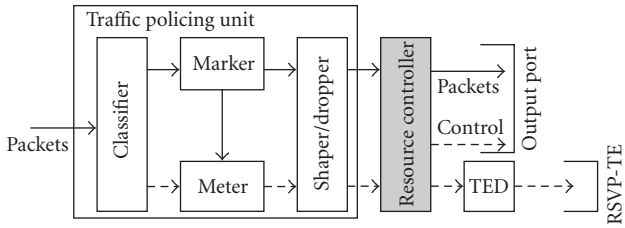


Figure 8: The structure of the ingress MPLS node.

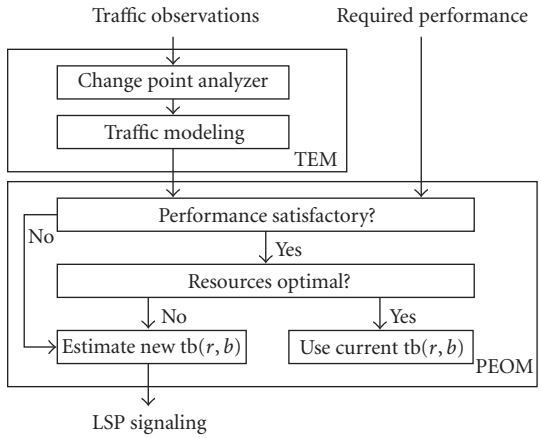


Figure 9: The structure of EWMA resource controller.

real-time traffic estimation module (TEM) and the performance evaluation and optimization module (PEOM). The system operates as follows. The TEM is responsible for detecting changes in arriving traffic statistics and dynamic estimation of the traffic state in terms of AR(1) model. To achieve that, traffic statistics must be observed in real-time and fed to the input of the change-point analyzer. Change-point analyzer tests incoming observations for changes in the mean value using EWMA statistical test. It is important to note that the change-point analyzer must signal the point when a change in arriving traffic statistics is detected. Otherwise, no actions must be taken by the TEM except for traffic monitoring. If a change is detected, traffic statistics are computed during the warmup period. Then a new model of the traffic is parameterized in the modeling block and fed to the input of PEOM.

According to the design of PEOM, traffic model is fed to the input of the decision module. Taking the required performance at the IP layer as another input, this module decides whether the current performance is satisfactory. If the performance is satisfactory, the system must further decides whether the amount of resources for a current state of the traffic is optimal. In order to take these two decisions, the module containing the performance evaluation framework is activated and supplied with the current traffic model. This module may implement the performance evaluation framework or just contain a set of precomputed performance curves corresponding to a wide range of traffic statistics. Due to the real-time nature of the performance control system, the latter approach is preferable.

Theoretically, when the input traffic statistics change, performance provided to aggregated traffic or optimal resource allocation providing a certain performance level or both change. However, in practical applications we always have a finite granularity of performance parameters and, therefore, we have to check whether a new traffic model does not satisfy performance requirements of aggregated traffic or results in not optimal resource allocations. If the performance is predicted to be unsatisfactory, the current traffic model is used to decide which resource allocation parameters provide both the required performance level and optimal usage of resources. Finally, if the resource allocation is not optimal but the performance is satisfactory, new optimal resource allocation must be computed. Obtained resource allocation parameters are then used till the next change in traffic statistics.

5.2. Estimating resource allocation

As a descriptor of the amount of resources required by an LSP, token bucket is used. The actual amount of resources in terms of the buffer space and outgoing link rate share is inferred from token bucket parameters at each node. According to the token bucket mechanism, the amount of traffic allowed in the time interval $[0, t)$ is upper bounded by $A(t) = rt + b$, $t \geq 0$, where token rate r is related to the outgoing link rate share, and bucket size b is related to the buffer space. To parameterize a token bucket, we have to find a Fair (r, b) satisfying performance criteria. Usually, there are infinite number of pairs (r, b) satisfying the required loss performance. However, there is always upper bound on b that also satisfies delay requirements. There are a number of approaches to estimate token bucket parameters using statistics of arrival process. Approaches range from approximations providing token bucket parameters for worst case traffic scenarios to exact ones involving solution of equivalent queueing systems [24, 25]. We are interested in simple approach providing a feasible option for online implementation.

To estimate parameters of the token bucket, we use overflow theory. Let $\gamma(t)$ be the cumulative arrival process and let $A(s, t)$ be the amount of work arriving in the interval $(s, t]$. The probability that the queue length exceeds b is then

$$Pr\{Q > b\} = Pr\left\{\sup_{t\geq 0}(\gamma(t) - rt) > b\right\}, \tag{8}$$

where r is the link rate share assigned to a traffic aggregate.

Using lower bound approximation, we get

$$Pr\left\{\sup_{t\geq 0}(\gamma(t) - Ct) > b\right\} \geq \sup_{t\geq 0} Pr\{\gamma(t) > b + Ct\}. \tag{9}$$

Denoting $F_t(x) = Pr\{\gamma(t) \leq x\}$, we get

$$Pr\{Q > b\} \geq \sup_{t\geq 0} Pr\{\gamma(t) > b + Ct\} = \sup_{t\geq 0}(1 - F_t(b + Ct)). \tag{10}$$

Note that the bucket size, b, should be set such that the delay is equal to or less than the maximum allowable

delay in a network element. Thus the task in (10) reduces to finding a suitable link rate share that should be assigned to a traffic aggregate. When $F_t(x)$ is normal, the following approximation can be used [26]:

$$r = E[X] + \alpha\sigma[X], \quad \alpha = \sqrt{-2\ln\epsilon - \ln 2\pi}, \qquad (11)$$

where ϵ is the buffer overflow probability.

5.3. The immediate update mechanism

According to our algorithm, new resource reservation should only be advertised to network elements when the resource controller already estimated a new allocation. This information is available at the end of the warmup period. For some traffic patterns (e.g., VoIP aggregates), this period can be long enough to receive inadequate treatment in a network. To ensure that the proposed algorithm continuously provides the desired level of performance, we propose to use following adaptive mechanism. When increase in the mean value is detected, a new resource allocation (r_+, b_+) is immediately estimated and advertised to the output buffer and network elements along the path of the LSP. Since at the beginning of the warmup period we do not know traffic statistics yet, the choice of (r_+, b_+) is somewhat arbitrary. One possible way is to estimate new resource allocation using mean $(E[Y] + C_Y)$ and variance, $\sigma^2[Y]$, where $E[Y]$, C_Y, and $\sigma^2[Y]$ are the mean, control limit, and variance of previous in control segment, respectively. When the warmup period expires and statistics of new in control process are available, new resource allocation is computed and advertised.

The same algorithm can be used when the mean value decreases. Another approach is to advertise new resource allocation at the end of the warmup period only. Indeed, since the mean value decreases we still ensure that appropriate performance is continuously provided. Note that the immediate update mechanism results in additional signaling load consisting of RSVP-TE reservation updates and subsequent IGP-TE resource allocation advertisements [27].

6. NUMERICAL EXAMPLES

6.1. Aggregated traffic: fixed number of sources

In Section 2, we assumed that aggregated traffic from fixed number of video traffic sources composes realization of the covariance stationary process. Let us now provide one more reason for this conclusion applying EWMA change-point test to first two traces demonstrated in Figure 4. We already found that this traffic is normally distributed and there is significant lag-1 autocorrelation. Due to these properties, control limits are computed according to (7). The warmup period used to compute control limits was set to 50 observations. EWMA statistics for different values of γ and k are shown in Figure 10. Note that $k = 3$, $\gamma = 0.01$ correspond to 137.91 in control ARL for the first trace and 175.24 in control ARL for the second trace. Parameters $k = 3$ and $\gamma = 0.001$ correspond to 956.68 in control ARL for the first trace and 1286.97 in control ARL for the second one. One can observe that no change in the mean value of traffic observations is detected even though the ARL values for $k = 3$ and $\gamma = 0.01$ are relatively small. These results suggest that the mean value of the aggregated traffic from fixed number of sources does not vary in time as required by covariance stationarity. Similar results have been obtained for other traces.

Note that for both traces and all values of γ, the parameter k was set to 3. In general, to match a given ARL value, different values of γ require different values of k. It was also found that usage of tabulated values may lead to many out of control signals even when there are no new session arrivals or departures. One of the reasons is that the monitored process may not be exactly covariance stationary and may occasionally contain some extreme observations (outliers). These observations may be of local significance only and may not affect the future evolution of the monitored process as well as the service process of arriving traffic. Since the EWMA change detection is assumed to operate in real time, each out of control signal starts a new warmup period. During this period, new control limits are estimated and the process cannot be monitored. From this point of view, we should detect only those changes that occur for sure.

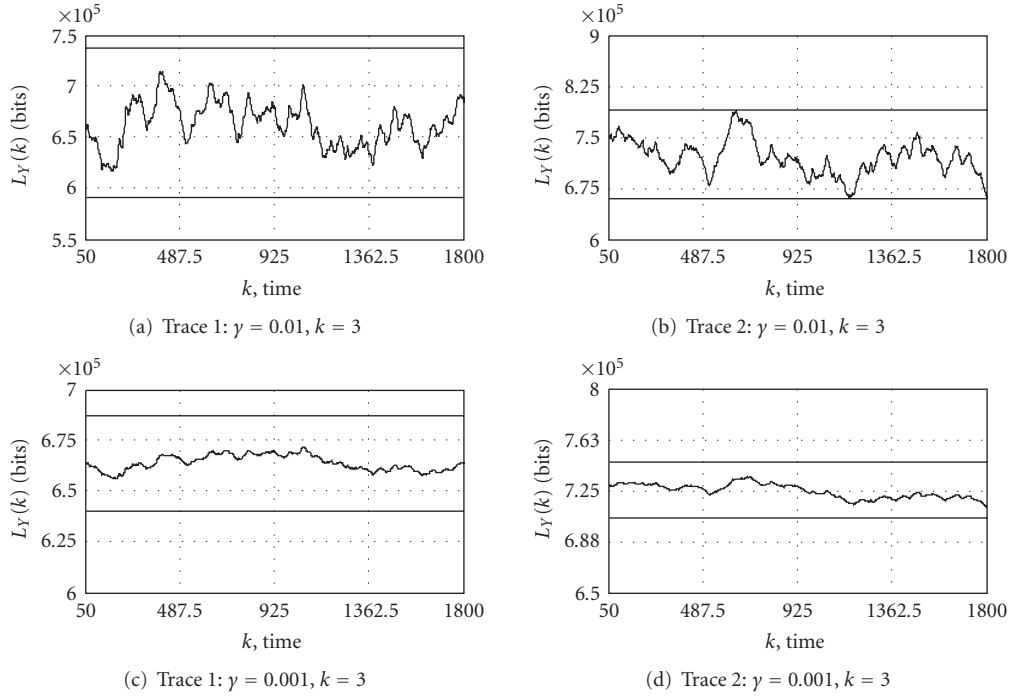
Results presented in this subsection provide additional necessary condition for covariance stationary of the multiplexed traffic from fixed number of video sources. Our observations stay in control in EWMA chart constructed for AR(1) process suggesting that they can be realization of this process. Thus EWMA chart can be seen as a tool for testing stationarity of observations.

6.2. Aggregated traffic: varying number of sources

Consider now aggregated traffic from varying number of sources shown in Figure 2. EWMA statistics computed for these traces are shown in Figures 11(a) and 11(b) ($k = 3$, $\gamma = 0.001$). Figures 11(c) and 11(d) demonstrate the same statistics for first several session requests, where solid horizontal lines represent control limits and boxes represent session arrivals. The first box indicates the time instant when second session request arrives to the system.

Consider EWMA statistics in detail. We started the control chart when the first session arrives. It occurred at 1681 second for trace 1 and at 334 second for trace 2. The warmup period used to compute statistics of observations and control limits of charts was set to 50 observations. One may observe from Figures 11(c) and 11(d) that both processes stay in control when the second sessions arrive and remain in control until new traffic adds up to EWMA statistics and changes are detected. Note that there are gaps before EWMA statistics exceed the upper control limit for both traces. This is due to the memory of EWMA statistics that allows to avoid false signals but worsens reactive properties of the chart. In general, when the value of γ gets smaller, this interval becomes larger while the probability of false change detection decreases.

EWMA statistics computed for our traces with $k = 3$ and $\gamma = 0.001$ are shown in Figures 12(a) and 12(b). The same



(a) Trace 1: $\gamma = 0.01$, $k = 3$

(b) Trace 2: $\gamma = 0.01$, $k = 3$

(c) Trace 1: $\gamma = 0.001$, $k = 3$

(d) Trace 2: $\gamma = 0.001$, $k = 3$

Figure 10: EWMA statistics for aggregated traffic: fixed number of sources.

statistics for first several session arrivals are shown in Figures 12(c) and 12(d). Note that changes in mean values of the aggregated traffic were successfully detected for these values of γ and k.

6.3. Dynamic resource allocation

The proposed dynamic resource allocation system tries to provide bandwidth savings compared to busy hour reservation and conventional MPLS automatic bandwidth adjustment while keeping performance metrics of interest at the desired level. Consider how much gain we get applying the proposed algorithm.

Firstly, we compare performance of the proposed algorithm to that of the static busy hour resource reservation. The performance metric of interest is the amount of bandwidth required to transmit our traffic aggregates with a given overflow probability. Values of bandwidth allocation for our algorithm were computed using (11) such that the overflow probability is $\epsilon = 0.01$. Busy hour bandwidth allocation was chosen as the maximum bandwidth allocation computed by our algorithm. Note that it is different from the actual busy hour and represents the amount of bandwidth required to serve the traffic during the highest in control period of EWMA control chart. Parameter k of control charts was chosen such that ARL was always kept at 500. We also used the immediate update mechanism.

Figure 13 demonstrates bandwidth allocation according to considered algorithms. Estimated values of bandwidth allocation are shown in Table 1, where average values are shown. Note that our algorithm always provides significant performance gain compared to busy hour resource reservation. For trace 1 our algorithm allows to save 71% and 70% of bandwidth when EWMA change-point detection with $\gamma = 0.01$ and $\gamma = 0.001$ is used, respectively. For trace 2, these numbers are 70% and 69%, respectively.

Next we compare performance of the proposed algorithm to that of MPLS automatic bandwidth adjustment. Performance metrics of interest are the amount of LSP reestablishments and the amount of required bandwidth according to these two approaches. Values of bandwidth allocation were computed using (11) such that the overflow probability is $\epsilon = 0.01$. To highlight influence of k and γ on the performance of the proposed algorithm, parameter k was set such that ARL values for $\gamma = 0.01$ are always three times higher than those with $\gamma = 0.001$. For $\gamma = 0.01$, k was always chosen such that ARL is 300.

Figure 14 illustrates bandwidth allocation required by these two approaches. As one can notice, MPLS automatic bandwidth allocation always leads to higher required bandwidth for a given traffic aggregate. This is due to specific behavior of arrival statistics during sampling intervals over which the bandwidth allocation is estimated. This includes occasional outliers, drastic and gradual changes in arrival patterns. All these features lead to biased estimates of mean and variance that are further used to compute bandwidth allocation. Note that they also affect performance of the proposed algorithm.

It should also be stressed that the performance of MPLS bandwidth adjustment algorithm is highly sensitive to the choice of the sampling interval. When the sampling interval is too large, changes in arrival statistics are more likely to occur within a single interval. Large sampling intervals also lead to delays in LSP reestablishments with

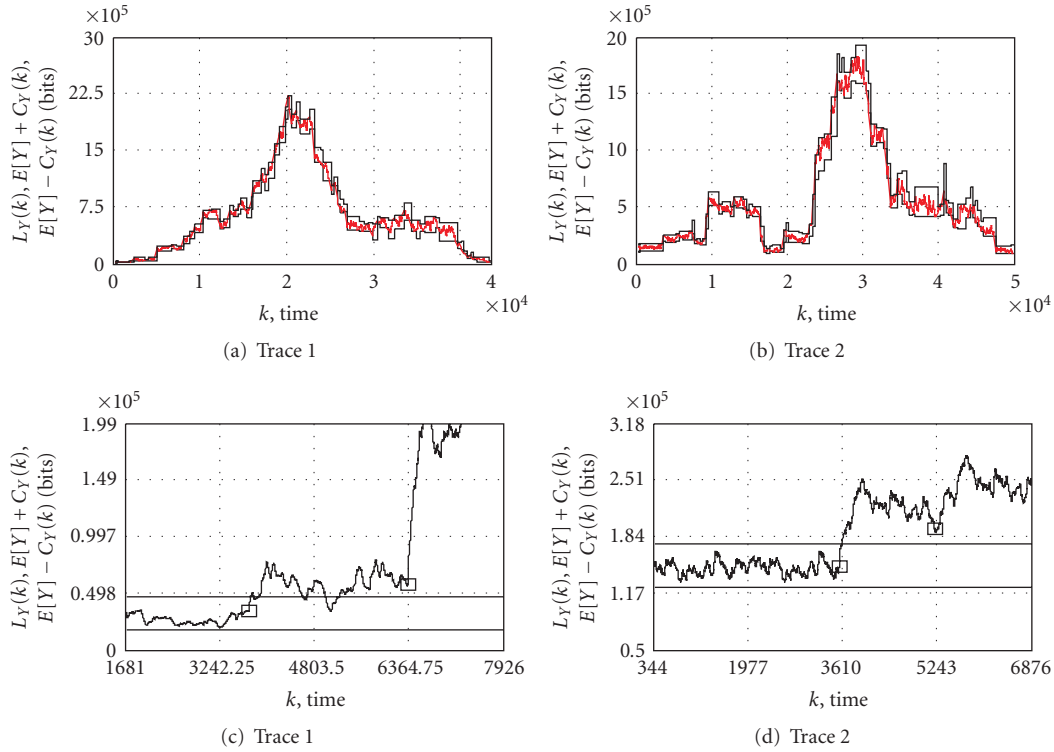


Figure 11: EWMA statistics: varying number of sources ($\gamma = 0.01$).

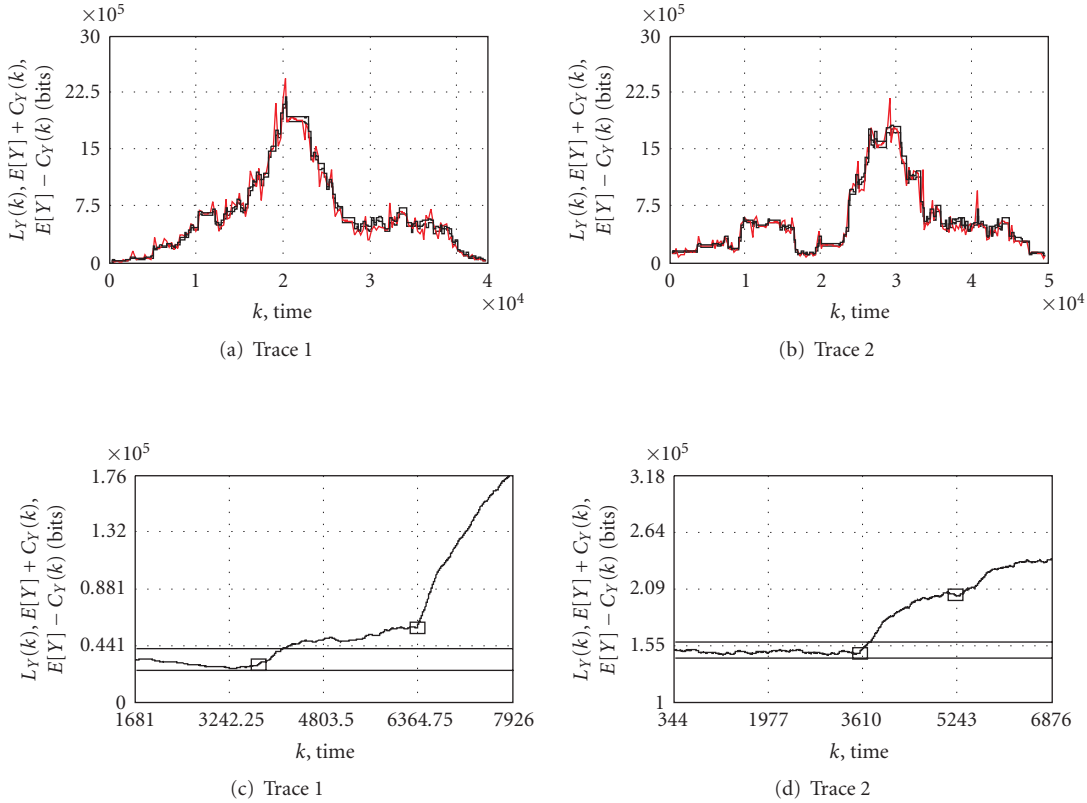


Figure 12: EWMA statistics: varying number of sources ($\gamma = 0.001$).

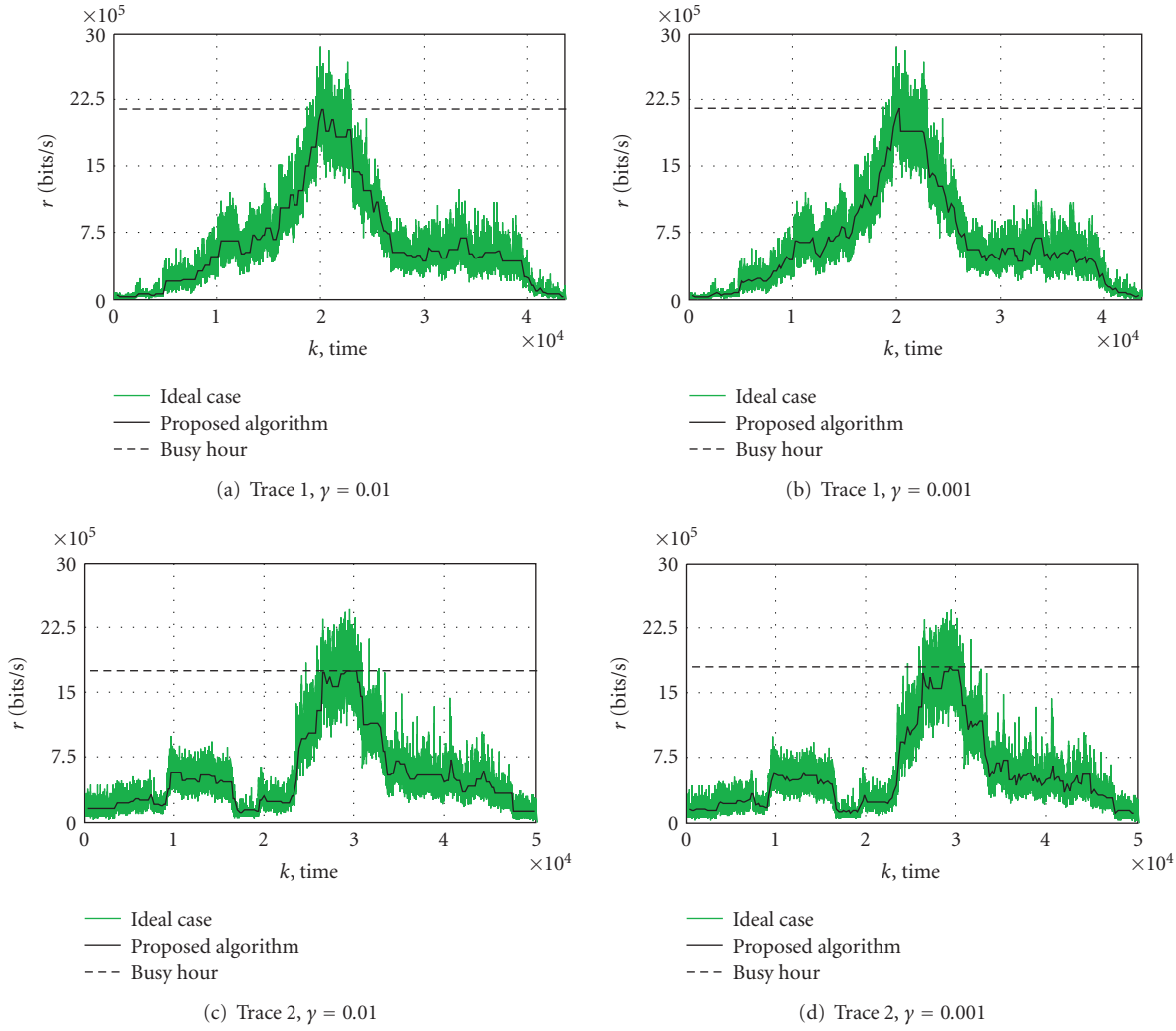


Figure 13: Bandwidth allocations for different approaches.

Table 1: Comparison with static busy hour bandwidth allocation.

Trace	Trace 1, bits/s.		Trace 2, bits/s.	
γ	$\gamma = 0.01$	$\gamma = 0.001$	$\gamma = 0.01$	$\gamma = 0.001$
Busy hour algorithm	2.14E6	2.16E6	1.76E6	1.80E6
Proposed approach	6.15E5	6.51E5	5.37E5	5.59E5

estimated bandwidth allocation. Due to presence of changes in arrival statistics, estimated bandwidth allocation may not be optimal at the end of intervals. On the other hand, when the sampling interval is too short, too many LSP reestablishments may burden the network with excessive signaling load.

Table 2 compares the amount of signaling load and average bandwidth required to transfer considered traffic aggregates for different lengths of the sampling interval and different values of γ. The average amount of bandwidth required to serve traffic aggregates according to our algorithm is considered as 100%. Parameters $\phi_{0.01}$ and $\phi_{0.001}$ denote the percentage of average bandwidth required by MPLS automatic bandwidth adjustment compared to our algorithm. First of all, observing Tables 1 and 2 we note that MPLS automatic bandwidth adjustment allows to significantly decrease the amount of bandwidth required to serve considered traffic aggregates compared to static busy hour bandwidth allocation. Although the performance gain is significant, it has observable limit. Indeed, when we decreased the length of the sampling interval 10 times (from 500 to 50 seconds) the average bandwidth required to serve traffic aggregates decreased by only 0.04% for trace 1 and 0.05% for trace 2. This is very small performance gain, especially, considering that we are close to theoretical and practical limits. Indeed, further decrease in the length of the

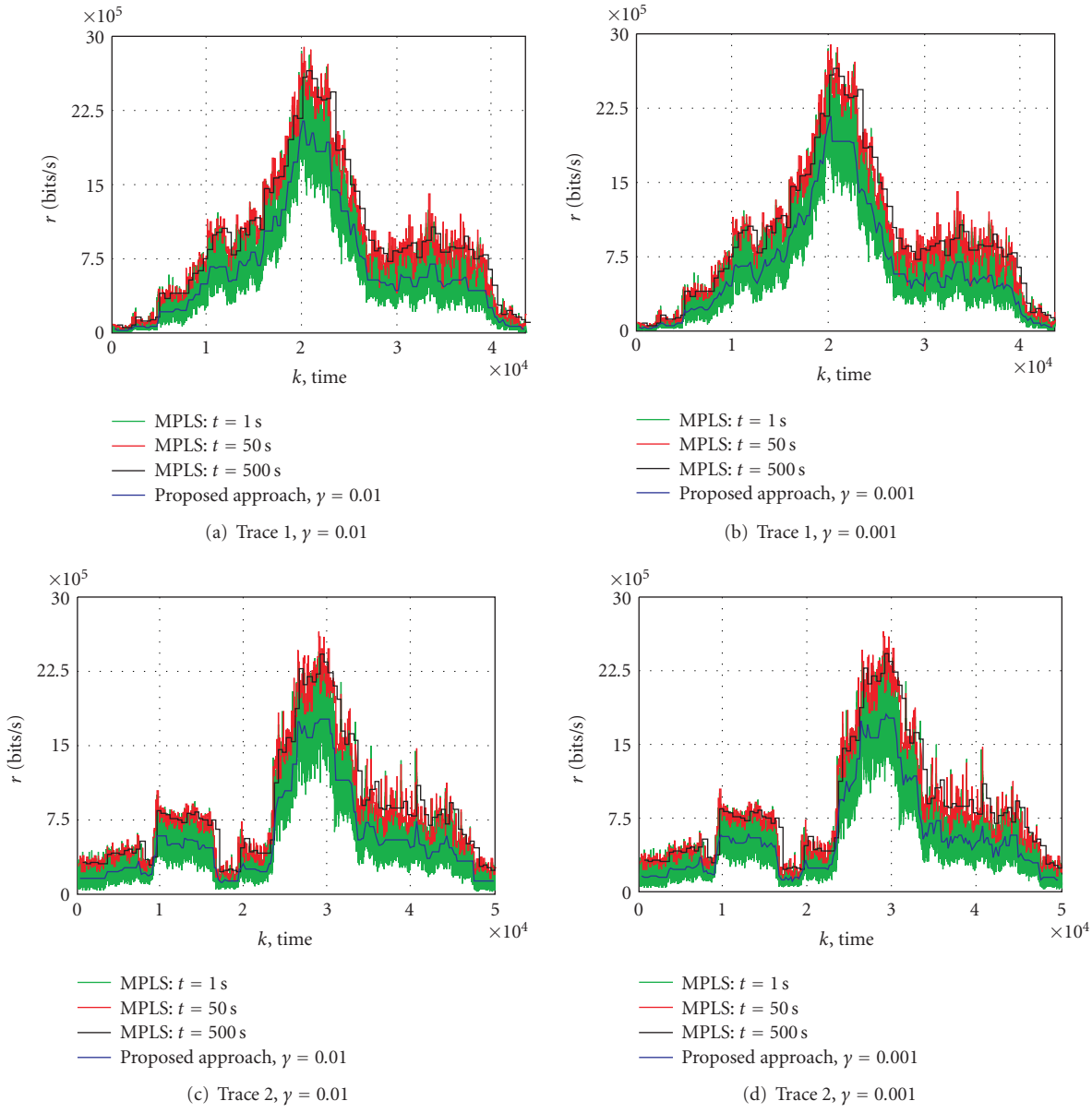


Figure 14: Bandwidth allocations according to different approaches.

sampling interval will lead to biased estimators of mean and variance and may overload the network with signaling load. On the other hand, as we observe, the proposed algorithm performs significantly better. For both values of γ it requires significantly less signaling load while still provides the same performance level in terms of losses. Noticeably that for $\gamma = 0.01$ our approach requires even less signaling than MPLS bandwidth adjustment with sampling interval set to 500 seconds.

Finally, influence of parameters k and γ on the performance of the proposed algorithm is also seen from Table 2. Indeed, three times lower ARL that was used with $\gamma = 0.001$ led to almost three times higher number of change detections. Although in our particular case it did not lead to extremely biased bandwidth allocation, it is still noticeable. This effect is due to more frequent change detections in arriving traffic patterns leading to periods of uncertainty during which statistical parameters are estimated. During these periods we had to use the immediate update mechanism that may overestimate or underestimate the amount of required bandwidth.

It should be noted that, theoretically, the proposed algorithm cannot provide deterministic guarantees even if we use the immediate update mechanism and the target overflow probability is set to 0. The main reason is uncertainty introduced by warmup periods during which statistical characteristics of traffic aggregates are estimated. However, in all our experiments the cumulative length of warmup periods was significantly shorter compared to the overall length of in control periods.

Table 2: Comparison with MPLS automatic bandwidth adjustment.

Algorithm	Updates	r, bits/s.	$\phi_{0.01}$, %	$\phi_{0.001}$, %
		Trace 1		
MPLS, $t = 50$ s.	870	9.21E5	150	141
MPLS, $t = 100$ s.	435	9.24E5	150	142
MPLS, $t = 200$ s.	145	9.35E5	152	143
MPLS, $t = 500$ s.	87	9.57E5	155	147
Proposal, $\gamma = 0.01$	66	6.15E5	100	100
Proposal, $\gamma = 0.001$	199	6.51E5	100	100
		Trace 2		
MPLS, $t = 50$ s.	1001	8.13E5	151	145
MPLS, $t = 100$ s.	500	8.21E5	153	147
MPLS, $t = 200$ s.	166	8.31E5	155	149
MPLS, $t = 500$ s.	100	8.56E5	159	153
Proposal, $\gamma = 0.01$	75	5.37E5	100	100
Proposal, $\gamma = 0.001$	180	5.59E5	100	100

7. CONCLUSIONS

We proposed the dynamic resource description and reservation algorithm for QoS-aware networks. The core of the algorithm is statistical change detection procedure. We demonstrated that if the local mean of the traffic pattern changes in time, the required amount of resources needed to serve this traffic with given performance metrics can be appropriately adapted using already available features of MPLS framework. The only required change is EWMA change-detection algorithm that should be implemented in ingress MPLS routers. The proposed algorithm is well suited for non stationary type of the traffic whose statistical parameters change in time. Numerical examples demonstrated that when traffic patterns exhibit nonstationary behavior the proposed approach allows to save significant amount of resources compared to busy hour resource allocation and conventional MPLS automatic bandwidth adjustment.

We also note that content distribution services considered in this paper are just example of applications that may have time-varying traffic characteristics. In principle, the proposed approach can be used for any type of the aggregated traffic that experiences high variability due to time-varying statistical characteristics.

REFERENCES

[1] K. Thompson, G. J. Miller, and R. Wilder, "Wide-area internet traffic patterns and characteristics," *IEEE Network*, vol. 11, no. 6, pp. 10–23, 1997.

[2] S. Bates and S. McLaughlin, "Is VBR video non-stationary or self-similar? Implications for ATM traffic characterisation," in *Proceedings of the 10th European Simulation Multiconference (ESM '96)*, pp. 27–34, Budapest, Hungary, June 1996.

[3] T. Karagiannis, M. Molle, M. Faloutsos, and A. Broido, "A nonstationary poisson view of Internet traffic," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '04)*, vol. 3, pp. 1558–1569, Hong Kong, March 2004.

[4] J. Cao, W. Cleveland, D. Lin, and D. Sun, "On the non-stationarity of internet traffic," in *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '01)*, pp. 102–112, Cambridge, Mass, USA, June 2001.

[5] J. Kilpi and I. Norros, "Testing the Gaussian approximation of aggregate traffic," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment (IMW '02)*, Marseille, France, November 2002, http://www.imconf.net/imw-2002/proceedings.html.

[6] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson, "Self-similarity through high-variability: statistical analysis of Ethernet LAN traffic at the source level," *IEEE/ACM Transactions on Networking*, vol. 5, no. 1, pp. 71–86, 1997.

[7] A. Feldmann, A. Gilbert, P. Huang, and W. Willinger, "Dynamics of IP traffic: a study of the role of variability and the impact of control," in *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM '99)*, pp. 301–313, Cambridge, Mass, USA, August-September 1999.

[8] W. Willinger, D. Alderson, and L. Li, "A pragmatic approach to dealing with high-variability in network measurements," in *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement (IMC '04)*, pp. 88–100, Taormina, Sicily, Italy, October 2004.

[9] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," RFC 3031, IETF, 2001.

[10] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus, "Requirements for traffic engineering over MPLS," RFC 2702, IETF, 1999.

[11] D. Katz, K. Kompella, and D. Yeung, "Traffic engineering (TE) extensions to OSPF version 2," RFC 3630, IETF, 2003.

[12] H. Smit and T. Li, "IS-IS extensions for traffic engineering (TE)," RFC 3784, IETF, 2004.

[13] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow, "RSVP-TE: extensions to RSVP for LSP tunnels," RFC 3209, IETF, 2001.

[14] L. Andersson, P. Doolan, N. Feldman, A. Fredette, and B. Thomas, "LDP Specification," RFC 3036, IETF, 2001.

[15] "MPEG-4 and H.263 video traces for network performance evaluation," Technical University of Berlin, http://www.tkn.tu-berlin.de/research/trace/trace.html, 2006.

[16] "MPEG traffic archieve," University of Wuerzburg, http://www3.informatik.uni-wuerzburg.de/MPEG/traces/, 2003.

[17] G. Box and G. Jenkins, *Time Series Analysis: Forecasting and Control*, Holden Day, San Francisco, Calif, USA, 2nd edition, 1976.

[18] S.-Q. Li and C.-L. Hwang, "Queue response to input correlation functions: discrete spectral analysis," *IEEE/ACM Transactions on Networking*, vol. 1, no. 5, pp. 522–533, 1993.

[19] B. Hajek and L. He, "On variations of queue response for inputs with the same mean and autocorrelation function," *IEEE/ACM Transactions on Networking*, vol. 6, no. 5, pp. 588–598, 1998.

[20] J. Wieringa, "Control charts for monitoring the mean of AR(1) data," Department of Econometrics, Faculty of Economic Sciences, University of Groningen, http://www.ub.rug.nl/eldoc/som/a/98a09/98a09.pdf, 2005.

[21] J. Wieringa, "Statistical process control for serially correlated data," PhD Thesis, Department of Econometrics, Faculty of Economic Sciences, University of Groningen, Groningen, The Netherlands, 2005, http://dissertations.ub.rug.nl/faculties/eco/1999/j.e.wieringa/.

[22] W. A. Shewhart, *Statistical Method from the Viewpoint of Quality Control*, Graduate School, Department of Agriculture, Washington, DC, USA, 1939.

[23] E. S. Page, "A test for a change in a parameter occurring at an unknown point," *Biometrika*, vol. 42, no. 3-4, pp. 523–527, 1955.

[24] Y. Koucheryavy, D. Moltchanov, and J. Harju, "A top-down approach to VoD traffic transmission over DiffServ domain using the AF PHB class," in *Proceedings of IEEE International Conference on Communications (ICC '03)*, vol. 1, pp. 243–249, Anchorage, Alaska, USA, May 2003.

[25] R. G. Garroppo, S. Giordano, and M. Pagano, "Estimation of token bucket parameters for aggregated VoIP sources," *International Journal of Communication Systems*, vol. 15, no. 10, pp. 851–866, 2002.

[26] R. Guerin, H. Ahmadi, and M. Naghshineh, "Equivalent capacity and its application to bandwidth allocation in high-speed networks," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 7, pp. 968–981, 1991.

[27] F. Le Faucheur, "Protocol extensions for support of Diffserv-aware MPLS traffic engineering," RFC 4124, IETF, 2005.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2008, Article ID 628970, 13 pages
doi:10.1155/2008/628970

Research Article

Rate-Distortion Optimized Frame Dropping for Multiuser Streaming and Conversational Videos

Wei Tu,[1] Jacob Chakareski,[2] and Eckehard Steinbach[1]

[1] *Media Technology Group, Institute of Communication Networks, Munich University of Technology, 80333 Munich, Germany*
[2] *Vidyo Inc., Hackensack, NJ 07601, USA*

Correspondence should be addressed to Wei Tu, wei.tu@tum.de

Received 11 May 2007; Accepted 14 September 2007

Recommended by Zhu Li

We consider rate-distortion optimized strategies for dropping frames from multiple conversational and streaming videos sharing limited network node resources. The dropping strategies are based on side information that is extracted during encoding and is sent along the regular bitstream. The additional transmission overhead and the computational complexity of the proposed frame dropping schemes are analyzed. Our experimental results show that a significant improvement in end-to-end performance is achieved compared to priority-based random early dropping.

1. INTRODUCTION

In today's Internet, video packets are typically transmitted using best-effort service. The packet forwarding service at network nodes is significantly degraded if the network is congested. In this paper, we consider the scenario where K streaming videos and N conversational videos pass through a network node (e.g., a multimedia gateway) with limited forwarding resources, as illustrated in Figure 1. The packets can be temporarily cached in the node's buffer, but if the overload persists, the buffer will overflow and some packets will be lost. Our goal is to improve the overall quality of the streams for the given forwarding resource R_{out} of the node.

For video applications, transcoding [1–3] or pruning of the video stream can be performed to adapt the source rate to the available transmission rate. Transcoding is computationally expensive and not suitable for a node that has to rapidly forward packets of many different users. Furthermore, video source pruning by random frame dropping may have a dramatic influence on the reconstructed video quality. In [4–6], static priority labels for I-, P-, and B-frames are used to perform priority-based random dropping (PRD) for streaming video. In particular, video frames are dropped according to their priority labels. Random selection is performed among frames with the same priority label. Priority-based random early dropping (PRED) [7] improves PRD by early dropping of lower priority frames at certain predefined buffer fullness levels. Nonetheless, static priority labels cannot accurately describe the importance of video frames. For example, the first P-frame in a group of pictures (GOP) is in most cases much more important than the last P-frame, although they belong to the same priority class. In [8, 9], the decodability of the video frames is used to make dropping decisions.

Rate-distortion (RD) optimization has been widely employed to deal with the varying importance of video frames. In [10], for instance, it is used to achieve RD-optimized frame scheduling for a single video stream. RD-optimization for bit allocation between source coding and channel coding is used, for example, in [11]. RD-optimization is also the state-of-the-art for coding mode selection in video compression [12, 13]. An RD-based robust delivery scheme is proposed in [14, 15]. However, all these works focus on the video encoding at the sender to choose the best encoding and sending strategy according to the constrained transmission rate and the expected packet loss rate. When RD-optimization is done at intermediate nodes in the network, side information has to be transmitted along with the video stream [16–18]. In particular, [16] considers RD-optimization in a broadcast networking scenario, and performs optimization only for a single stream, while [17] employs RD side information for transcoding at network nodes. Only streaming video is considered in [18], while in this paper we also examine the

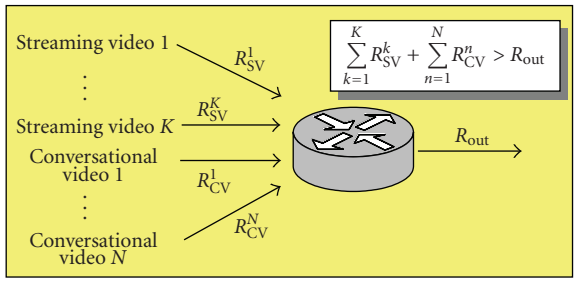


Figure 1: A network node with K incoming streaming videos and N conversational videos sharing the same outgoing link.

additional case of frame dropping for both streaming and conversational videos, simultaneously.

In the present paper, we propose RD-optimized video frame dropping strategies for streaming or conversational video on overloaded network nodes. For rate shaping streaming videos, we use a *distortion matrix* and a *rate vector* [19] as side information. We denote our approach as the cost function-based approach, which minimizes a Lagrangian cost function in order to find the optimum dropping pattern. The cost function employs the following quantities to determine the optimal pattern: the *rate vector* and the *distortion matrix* of all incoming streams, as well as the current fullness of the outgoing buffer. The Lagrangian multiplier in the cost function is selected as a function of the buffer fullness and is used to adjust the aggressiveness of the dropping process. The *distortion matrix* can be extracted only for GOP structured video. For conversational video with IPPP... structure, only *hint tracks* [20, 21] can be calculated and therefore, the utility-based frame dropping strategy is used for the conversational videos. Frame dropping decisions are made only when the buffer does not have enough space to hold them.

In addition, we also examine the scenario when both streaming and conversational videos are passing through a network node. In this case, we propose to use separate classification buffers combined with a scheduler for dynamic resource assignment to the two buffers, which is located between the two classification buffers and the outgoing link buffer at the node. For simplicity, in our framework we replace continuous time with the discrete frame slots of the video sequences, which means that dropping decisions will only be made at multiples of one frame duration. In case the streams have different frame rates, the dropping decision can be made synchronized to the stream with the highest frame rate. Another approach would be to collect a small number of frames from all incoming streams and then perform a dropping decision. This approach, however, would introduce additional delay.

The main contributions of this work include the following:

(1) *joint rate shaping for both streaming and conversational videos*,

(2) *extensive simulation results* which provide a comprehensive performance comparison of different frame dropping schemes as well as a reference for parameter selection,

(3) *analysis of computational and storage cost* which can be used as a reference to select different dropping schemes for a given scenario.

The rest of the paper is organized as follows. Section 2 describes the side information used for streaming video and the corresponding frame dropping strategies. Next, the side information and dropping strategy for conversational video are introduced in Section 3. An integrated RD-optimized framework for both streaming and conversational video applications is presented in Section 4. Furthermore, in Section 5 we analyze the memory requirements and the computational complexity of the techniques considered in this paper. Section 6 presents the simulation results that demonstrate the improvements achieved by our proposed RD-optimized frame dropping strategies. Conclusions are drawn in Section 7.

2. FRAME DROPPING STRATEGIES FOR STREAMING VIDEO

In this section, we first introduce the priority-based frame dropping schemes, which are used for comparison in this paper. Then, the definition and the procedure to construct the side information (*distortion matrix* and *rate vector*) for our approach are presented. Based on this side information, we propose an RD-optimized frame dropping strategy based on the current buffer fullness of the network node.

2.1. Priority-based random early dropping (PRED)

PRD makes dropping decisions based on fixed priority labels assigned to the video frames. With current conventional coding scheme, frames can be prioritized according to their frame type: I-, P-, or B-frame. When frames from multiple video streams simultaneously arrive at a network node, it is the I-frames among them that have the highest priority to be placed into the node's buffer. It may happen though that some of these I-frames cannot be placed into the buffer and therefore they are dropped even before the buffer is totally full. In this case, P-frames are tried next to be placed into the buffer, followed then by the remaining (if any) B-frames. This strategy leads to the most efficient usage of the node's buffer. However, the loss of I-frames and P-frames has a dramatic influence on the reconstruction video quality.

PRED sets thresholds for dropping according to the number of priorities available for the video streams. Here, we only have three priority levels {I, P, B}, so only two dropping thresholds are needed. But if we have different priority levels for all P-frames according to their positions in the GOP, we can set $n = N_G/(N_B + 1)$ thresholds, where N_G is the length of the GOP and N_B denotes the number of B frames between two I/P-frames. As shown in Figure 2, when the buffer fullness reaches T_1, the least important B-frames are all dropped. The last P-frame in the GOP is dropped when T_2 is reached. I-frames have the highest priority level and should not be early dropped, so all P-frames are dropped when the buffer

fullness reaches the highest threshold (T_n). Early dropping of less important frames reduces the likelihood of having to drop more important frames at a later time.

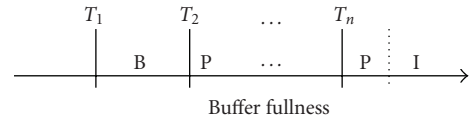


Figure 2: Example settings of dropping thresholds for PRED.

2.2. Distortion matrix (DM) and rate vector (RV)

The *distortion matrix* proposed in [19] allows us to calculate the distortion caused by dropping frames in a GOP structured video stream. When calculating the reconstruction distortion, it is assumed that a simple "copy and freeze previous frame" error concealment scheme is employed by the decoder. In particular, a missing frame and all of its descendants[1] are replaced, at reconstruction, by the decoder with the temporally nearest previous frame that has been decoded. Note that this is done regardless of the presence status, at the decoder, of the descendant frames. Hence the name of the concealment scheme.

Our approach to frame dropping follows this logic. When a network node drops an arriving video frame, it subsequently drops all its dependent frames that arrive at the node afterwards. Therefore, a video frame drop pattern comprises in our case an incoming frame that is dropped at present together with its descendant frames that will be dropped afterwards.[2] The increase in reconstruction distortion affecting a video stream caused by a frame drop pattern is the sum of the individual increments in reconstruction distortion for the concealed video frames. That is because the frames that have been decoded do not contribute to the increase in reconstruction distortion. The *distortion matrix* for a GOP with $IB_1B_2P_1B_3B_4P_2B_5B_6$ structure is given in (1), where $D_{F_{\text{loss}}}^{F_{\text{rep}}}$ represents the increased distortion in MSE that is observed when replacing frame F_{loss} by F_{rep} as part of the concealment strategy. The column left to the matrix shows the replacement frame F_{rep} for every row of the matrix. For instance, $D_{B_1}^I$ represents the additional reconstruction distortion if the first B-frame of the GOP is lost and is therefore replaced by the I-frame of that GOP. R is a frame from the previous GOP that is used as a replacement for all the frames in the current GOP if the I-frame of the current GOP is lost. As a worst case assumption, we use the I-frame of the previous GOP as the replacement frame in this case:

$$\begin{array}{c} R: \\ I: \\ P_1: \\ P_2: \\ B_1: \\ B_3: \\ B_5: \end{array} \begin{bmatrix} D_I^R & D_{B_1}^R & D_{B_2}^R & D_{P_1}^R & D_{B_3}^R & D_{B_4}^R & D_{P_2}^R & D_{B_5}^R & D_{B_6}^R \\ / & D_{B_1}^I & D_{B_2}^I & D_{P_1}^I & D_{B_3}^I & D_{B_4}^I & D_{P_2}^I & D_{B_5}^I & D_{B_6}^I \\ / & / & / & / & D_{B_3}^{P_1} & D_{B_4}^{P_1} & D_{P_2}^{P_1} & D_{B_5}^{P_1} & D_{B_6}^{P_1} \\ / & / & / & / & / & / & / & D_{B_5}^{P2} & D_{B_6}^{P2} \\ / & / & D_{B_2}^{B_1} & / & / & / & / & / & / \\ / & / & / & / & / & D_{B_4}^{B_3} & / & / & / \\ / & / & / & / & / & / & / & / & D_{B_6}^{B_5} \end{bmatrix}. \tag{1}$$

The entries of the *rate vector* correspond to the sizes of the video frames expressed in bytes. Then, at a network node, the size of an incoming frame and the sizes of its descendant frames are summed up to determine the rate saving achieved by dropping these frames.

2.3. Cost function-based video frame dropping

In Section 2.1, we reviewed the idea of PRED and discussed the benefit obtained by "early" dropping. In this section, a cost function-based approach is proposed, which takes advantage of the RD side information to enable more flexible frame dropping decisions, while still using the buffer fullness info for early dropping.

If the buffer is empty or is lightly loaded, no frames should be dropped. However, when the buffer fills up, frames that have the least impact on the perceived quality at the receiver should be dropped first. The decision which frames to drop is jointly made in our approach for all video streams. Given the RD side information introduced in Section 2.2, the active network node can perform RD-optimized frame dropping. For this, the node checks the current buffer fullness and minimizes the Lagrangian cost function

$$J_p(n) = \sum_{k=1}^{K} \Delta D_p^k(n) - \lambda(n) \sum_{k=1}^{K} \Delta R_p^k(n) \tag{2}$$

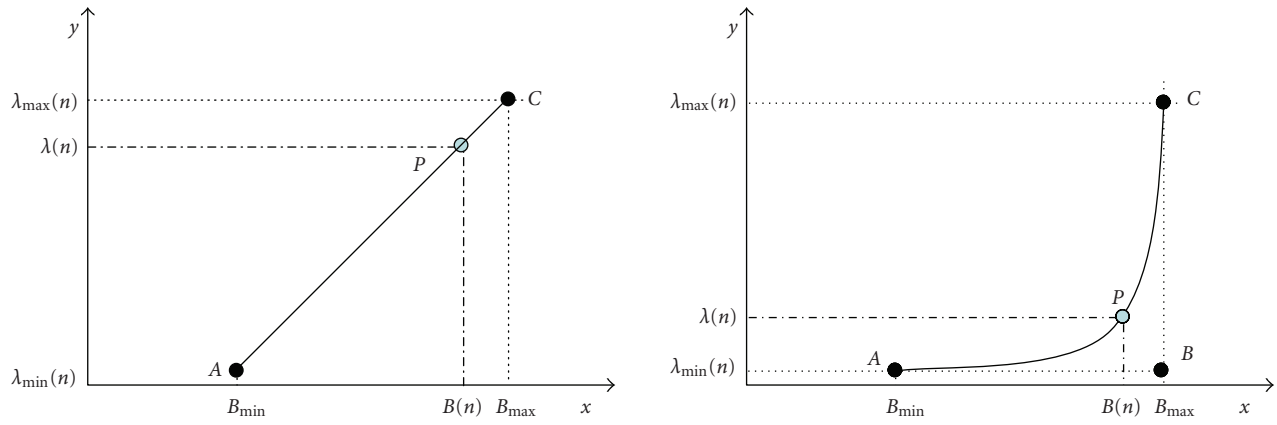
in order to determine the optimal drop pattern. In (2), n is the current (discrete) time instant (slot), $\Delta D_p^k(n)$ is the additional distortion introduced in video k for a given drop pattern p, and $\Delta R_p^k(n)$ is the corresponding rate saving in bytes.

When the *distortion matrix* and *rate vector* described in Section 2.2 are used, a dropping decision should comply with the following rules. If the current frame that arrives at the active node is an I-frame, we can either drop this frame or send it to the outgoing link buffer. If we drop it, this means that all the following P- and B-frames in the same GOP cannot be decoded and have to be dropped also. This dropping strategy leads to a significant increase in distortion for this GOP but at the same time allows us to reduce the sending rate to 0 for this GOP. If we do not drop the I-frame at this moment, we can still decide to drop the subsequent P-frames and B-frames. This will lead to reduced distortion but also the rate saving will be smaller. We could also drop the B-frames only if we decide not to drop the P-frames. Again, the additional distortion will be reduced but also the rate saving will be even smaller.

Therefore, if the current incoming frame is an I-frame, there are in total 4 dropping choices $\{I, P, B, N\}$, where I denotes dropping the whole GOP, P stands for dropping the subsequent P- and B-frames in the GOP, while B signifies dropping the all B-frames in the current GOP only and N stands for "drop nothing." If the current frame is a P-frame,

[1] These are the frames in the encoding chain that depend on the missing frame in order to be decoded, that is, decompressed.

[2] In case they do arrive at the node.



(a) Linear interpolation

(b) Quadratic interpolation

FIGURE 3: Interpolation of $\lambda(n)$ between $\lambda_{min}(n)$ and $\lambda_{max}(n)$ as a function of the current buffer.

the choices are reduced to $\{P, B, N\}$. If the current frame is a B-frame, the choices are also $\{B, P, N\}$, where B denotes the case of dropping all the remaining B-frames in the GOP, including the current one, and P stands for dropping the subsequent (relative to the current B-frame) P- and B-frames.

Now, if we denote the number of possible drop patterns at time n for video k as $A^k(n)$, then for K videos we obtain the dropping set $P(n)$ including $\prod_{i=1}^{K} A^i(n)$ different drop patterns. One of the drop patterns will minimize (2). This pattern represents the optimal dropping strategy at time n. In order to perform this minimization, we have to determine a reasonable value for the Lagrangian multiplier $\lambda(n)$ in (2). In this work, we determine $\lambda(n)$ as a function of the buffer fullness $B(n)$. If the buffer is empty, we certainly do not want to drop any video frames. This has to be reflected by an appropriate choice of $\lambda(n)$. On the other hand, if the buffer is full, $\lambda(n)$ should be selected such that all incoming frames are dropped as queueing them in the outlink buffer would fail anyway. In order to determine appropriate values for $\lambda(n)$ for any buffer level, we define a minimum buffer fullness B_{min}, below which no dropping should happen and a maximum buffer fullness B_{max} above which all incoming frames are dropped. The two buffer fullness levels B_{min}, B_{max} and the corresponding dropping strategies lead to two extreme values for the Lagrange multipliers $\lambda_{min}(n)$ and $\lambda_{max}(n)$. The values for $\lambda(n)$ between B_{min} and B_{max} can be interpolated. We consider two different interpolation schemes for $\lambda(n)$ in this work.

Figure 3(a) illustrates a linear interpolation of $\lambda(n)$ between $\lambda_{min}(n)$ and $\lambda_{max}(n)$ as a function of the current buffer fullness $B(n)$. Hence, we write

$$\lambda(n) = \frac{B_{max} - B(n)}{B_{max} - B_{min}} \cdot \lambda_{min}(n) + \frac{B(n) - B_{min}}{B_{max} - B_{min}} \cdot \lambda_{max}(n). \tag{3}$$

Linear interpolation is the simplest way to interpolate $\lambda(n)$. An interpolation function that leads to more aggressive dropping if the buffer fullness approaches B_{max} can be realized by quadratic interpolation of $\lambda(n)$, as shown in Figure 3(b). With three control points A, B, and C, we can define a quadratic Bézier curve for $\lambda(n)$ with

$$\begin{aligned} A &= (A_x, A_y) = (B_{min}, \lambda_{min}(n)), \\ B &= (B_x, B_y) = (B_{max}, \lambda_{min}(n)), \\ C &= (C_x, C_y) = (B_{max}, \lambda_{max}(n)), \end{aligned} \tag{4}$$

$$P_x = (1-t)^2 \cdot A_x + 2t \cdot (1-t) \cdot B_x + t^2 \cdot C_x, \tag{5}$$

$$P_y = (1-t)^2 \cdot A_y + 2t \cdot (1-t) \cdot B_y + t^2 \cdot C_y. \tag{6}$$

The interpolated point $P = (P_x, P_y)$ moves on this curve from A to C by varying the parameter t from 0 to 1. For a given $B(n)$, we determine t and then $\lambda(n) = P_y$ from (6).

In order to determine $\lambda_{min}(n)$, we evaluate (2) for every drop pattern and select $\lambda_{min}(n)$ such that the minimum of (2) is obtained for the drop pattern where nothing is dropped in all K video streams. This means that

$$\begin{aligned} J_{p_n}(n) &= \sum_{k=1}^{K} \Delta D_{p_n}^k(n) - \lambda_{min}(n) \sum_{k=1}^{K} \Delta R_{p_n}^k(n) \\ &\leq \sum_{k=1}^{K} \Delta D_{p}^k(n) - \lambda_{min}(n) \sum_{k=1}^{K} \Delta R_{p}^k(n), \\ &\qquad \text{for } p \in P(n),\ p \neq p_n, \end{aligned} \tag{7}$$

where p_n represents the pattern when no frame dropping occurs in any of the video streams. As $J_{p_n}(n)$ equals zero, this leads to

$$\lambda_{min}(n) \leq \frac{\sum_{k=1}^{K} \Delta D_{p}^k(n)}{\sum_{k=1}^{K} \Delta R_{p}^k(n)}, \quad \text{for } p \in P(n),\ p \neq p_n, \tag{8}$$

and we pick $\lambda_{min}(n)$ to be as big as possible while still satisfying all the inequalities in (8). The value for $\lambda_{max}(n)$ is derived in a similar fashion. For this, the minimization of (2) should

now lead to the decision of dropping as many frames as possible (drop pattern p_a), which leads to

$$\begin{aligned} J_{p_a}(n) &= \sum_{k=1}^{K} \Delta D_{p_a}^{k}(n) - \lambda_{\max}(n) \sum_{k=1}^{K} \Delta R_{p_a}^{k}(n) \\ &\le \sum_{k=1}^{K} \Delta D_{p}^{k}(a) - \lambda_{\max}(n) \sum_{k=1}^{K} \Delta R_{p}^{k}(n), \\ &\qquad \text{for } p \in P(n),\ p \neq p_a. \end{aligned} \tag{9}$$

This results in

$$\lambda_{\max}(n) \ge \frac{\sum_{k=1}^{K} \Delta D_{p_a}^{k}(n) - \Delta D_{p}^{k}(n)}{\sum_{k=1}^{K} \Delta R_{p_a}^{k}(n) - \Delta R_{p}^{k}(n)}, \quad \text{for } p \in P(n),\ p \neq p_a \tag{10}$$

and we pick $\lambda_{\max}(n)$ to be as small as possible while still satisfying all inequalities in (10).

3. FRAME DROPPING STRATEGIES FOR CONVERSATIONAL VIDEO

Compared with streaming video, conversational video is typically encoded in an IPPPPP... form. B-frames are normally not used because of the additional delay that would be introduced. Therefore, no "early" or even priority-based dropping as mentioned in Section 2.1 can be employed for conversational video. Video frames of multiple users are put into the buffer in a round robin (RR) way and dropped if the buffer cannot hold them. As conversational video does not have a GOP structure, the *distortion matrix* also cannot be used here to perform dropping decisions. Hence, we here propose to use the *hint tracks* [20, 21] as the side information and perform a utility-based frame dropping for conversational video to selectively drop the least important frames.

3.1. Side information for conversational video

Rate-distortion *hint tracks* are measured by feeding a specific loss pattern to the decoder and summing up the resulting increase in MSE over all affected frames of the video sequence. Without periodic I-frames in conversational video, there is no resynchronization between the encoder and decoder. Therefore, in order to increase the error resilience of the video stream to packet losses during transmission, slices (or rows) of macroblocks in video frames are intraupdated periodically, usually in a round-robin fashion. This is the so-called partial intraupdate. Figure 4 illustrates the error propagation when frame n is lost under the assumption that there is no remaining error propagating from earlier frames. The total distortion in this case is the sum of the distortions of all the following frames until the end of the video stream. However, with partial intraupdate, we can assume that the error propagation by the loss of frame n can be totally stopped after an equivalent intraupdate period of M frames.[3] Therefore, only the individual distortions up to frame $M + n$ need to be considered. Please note, here we calculate the *hint tracks* under the assumption that the losses of each frame are independent, which is the so-called zeroth-order distortion chain model DC^0 in [20]. This side information gives accurate distortion estimation when there is only one frame loss in the M consecutive frames. We can of course construct higher-order *hint tracks*, which can be extracted by feeding some loss patterns with more losses. However, high-order *hint tracks* have very high costs in terms of computational complexity as well as a huge storage requirement.

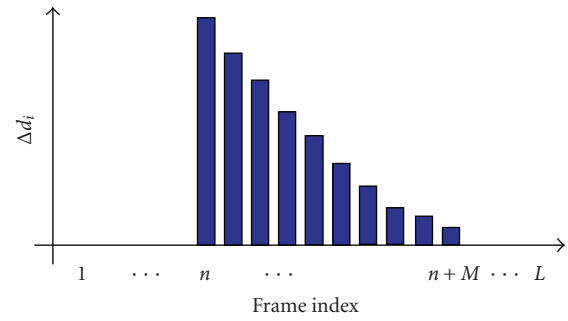


FIGURE 4: Error propagation for a single frame loss.

Since the future frame information for conversational video is unknown, it is impossible to premeasure the *hint track* (DC^0) value associated with a given loss pattern. Therefore, the model proposed in [22] is used to predict/estimate the distortion values $\Delta d(n + i)$ associated with future frames $n + i$ in the case of loss/drop of frame n. In particular,

$$\Delta d(n+i) = \begin{cases} \Delta d(n) \cdot r^{i} \cdot \left(1 - \dfrac{i}{M}\right), & \text{for } 0 \le i \le M, \\ 0, & \text{otherwise}, \end{cases} \tag{11}$$

where M is the equivalent intraupdate period, as explained above, and i indicates the distance between the future (concealed) frame and the lost frame n. In (11), $\Delta d(n)$ is the MSE information sent along with the video stream, representing the distortion of the current frame n in the case when this is the only lost frame and it is concealed by copying the previous frame. The attenuation factor r^i ($r < 1$) accounts for the effect of spatial filtering and the term $1 - i/M$ accounts for the intraupdate. Finally, the overall additional distortion $\Delta D(n)$ affecting the video sequence due to the loss of frame n, including error propagation into future frames, is then calculated as

$$\Delta D(n) = \sum_{i=0}^{M} \Delta d(n+i). \tag{12}$$

As a "copy and keep on decoding" error concealment scheme is employed in this case, the rate saving information

[3] This is the number of frames needed to intrarefresh all the macroblock locations in a video frame using this approach.

$\Delta R(n)$ associated with a given drop pattern is simply the sum of the size of the dropped frames. Therefore, only the sizes of the individual frames in bytes or bits need to be sent together with the *hint tracks* distortion information.

3.2. Utility-based frame dropping

Unlike streaming video, the importance of future frames in conversational video is unavailable. Therefore, it does not make sense to make dropping decisions for conversational videos until the buffer is unable to hold the new incoming frames. In particular, all new incoming frames are placed at the tail of the buffer queue if there is enough space left. Otherwise, we compare the importance of these frames and make a dropping decision.

In the *hint track* framework [18], for DC^0, the distortion ($\Delta D(n)$) and rate ($\Delta R(n)$) information associated with a video frame n comprise, respectively, the additional distortion affecting the reconstructed video sequence and the corresponding data rate reduction, when a single video frame n from the compressed video stream is dropped. The distortion-per-bit utility for a frame is then calculated as the ratio $\Delta D(n)/\Delta R(n)$ [20]. In our approach, we sort the current incoming (nth) frames from all K videos in decreasing order of their distortion per-bit utility. Then, from the head of the sorted list, frames are placed into the node's outgoing link buffer. If the frame at the current top of the list does not fit into the buffer, we turn to the next frame in the sorted list until no additional frame can be placed into the buffer. Please note, because of the tight delay constraint, optimization is done only among newly incoming video frames that correspond to a single time instant (one frame slot).

4. RD-OPTIMIZED DROPPING FOR STREAMING AND CONVERSATIONAL VIDEOS

In Sections 2 and 3, we have discussed the side information and dropping strategies for streaming and conversational videos. In this section, we consider the case when both types of video pass through the network node simultaneously and share one outgoing link.

4.1. Proposed framework

As shown in Figure 5, the RD-optimizer performs two independent dropping decisions for streaming video and conversational video, as proposed in [23]. The surviving (not being dropped) frames are stored in two independent classification buffers. The buffer for conversational video is relatively small in order to limit the forwarding delay experienced by these frames as this type of video application requires low latency. On the other hand, the classification buffer for streaming video is larger due to the more relaxed requirement on the delivery delay in this case. A scheduler is located behind the two buffers, which dynamically assigns the shared resource (forwarding data rate) to the two buffers by fetching video packets from them and putting them into the shared outgoing link buffer.

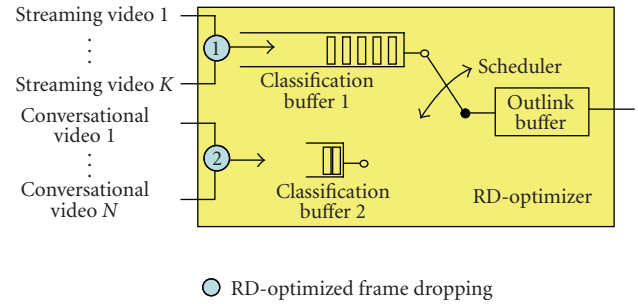


Figure 5: Structure of the RD-optimizer for frame dropping of streaming and conversational videos.

For streaming video, we opt to employ the cost function-based dropping strategies introduced in Section 2.3. The distortion-per-bit utility introduced in Section 3.2 is employed for the conversational video. For streaming video, information about future frames is taken into account. When the dropping decision is made for conversational videos, we can only compare the importance of the current frame with previous frames. As the selected frame is first put into the classification buffer, which we assume can be accessed by the RD-optimizer, frame replacement for this buffer is enabled. When new frames arrive at the node and the classification buffer is full, frames in the buffer with lower utility than the new incoming frame will be marked as dropping candidates. If the buffer space released by dropping these frames is enough to put in a new frame, they are physically dropped from the temporal buffer. On the other hand, if the released space is not enough to hold the new frame, it means the new frame is either too big or is not important enough for the reconstruction quality of the corresponding stream. Then this new frame is dropped and the marked frames in the buffer are recovered. Please note that this approach is equivalent to that taken in [20] for creating priorities among frames in a transmission window at a streaming server.

4.2. Scheduling strategies

Two separate classification buffers are employed to limit the additional delay experienced by the conversational video streams, as explained earlier. Compressed video has a variable bit-rate, and hence fixed resource assignment in terms of forwarding data rate sometimes wastes resources and leads to unnecessary frame dropping. With a dynamic resource assignment in place, the multiplexing of the multiple streams decreases the variation of the bit-rate and provides for more efficient resource utilization. Here, we propose two schemes for dynamic assignment of the data rate on the outgoing link.

4.2.1. Short-term mean-rate-based scheduling

Compressed video streams are typically VBR (variable bit rate), so when the outgoing link provides a transmission rate equal to the mean data rate of the incoming video stream, most likely some packets will be dropped if there is only a very small buffer at the node. But if we can perform the

assignment adaptively following the variability of the stream's bit-rate, the node's forwarding resources can be more efficiently used. Without the knowledge of the sizes of future frames for conversational video, we can only make an estimate of the future bit rate, given the knowledge of the incoming data rate history. Here, we present a straightforward way to account for this. We take F past frames from each stream as an estimation window. The current resource assignment is then calculated as follows:

$$r_{\mathrm{SV}}^{i} = \sum_{k=1}^{K} \sum_{j=i-F-1}^{i-1} R_j^k, \tag{13}$$

$$r_{\mathrm{CV}}^{i} = \sum_{n=1}^{N} \sum_{j=i-F-1}^{i-1} R_j^n, \tag{14}$$

$$S_{\mathrm{SV}}^{i} = R_{\mathrm{out}} \cdot \frac{r_{\mathrm{SV}}^{i}}{r_{\mathrm{CV}}^{i} + r_{\mathrm{SV}}^{i}}, \tag{15}$$

$$S_{\mathrm{CV}}^{i} = R_{\mathrm{out}} - S_{\mathrm{SV}}^{i}. \tag{16}$$

In the equations above, r_{SV}^{i} and r_{CV}^{i} are the sum of bytes from the previous F frames of K streaming videos and N conversational videos, respectively. S_{SV}^{i} and S_{CV}^{i} represent the assigned transmission rate to the two buffers. R_{out} is the total transmission rate on the outgoing link and it is assumed to be constant during the whole transmission. With the same formulae (15) and (16), dynamic resource assignment for variable data rate on the outgoing link rate can also be accommodated by considering R_{out} to be a function of time.

4.2.2. Buffer fullness-based scheduling

Buffer fullness-based scheduling is an efficient way for the scheduler to avoid buffer overflow. When a buffer is heavily loaded, it means its incoming rate of traffic is bigger than the assigned service rate and therefore new incoming frames are likely to be dropped. In this case, a large portion of the outlink rate should be assigned to this buffer. On the other hand, when one of the two buffers is lightly loaded, it can still hold some new incoming frames. Hence, more transmission slots should be assigned to the other buffer then. Furthermore, when the two buffers have roughly the same fullness, it is not efficient to assign the same amount of resource to each of them, as their corresponding incoming rates may differ significantly. This is because the two buffers serve two different types of applications: streaming video and conversational video that usually have different data rates. Hence, we assign a weight to each buffer according to their incoming rates, and distribute the forwarding resource among them based on these weights.

The mean rates calculated with (13) and (14) represent the most recent (short-term) rates feeding the two buffers. Since they vary rapidly over time, employing them to determine the buffer weights may actually be inappropriate in this case. In particular, they may overly influence the resource allocation among the two buffers, thereby rendering their instantaneous fullness less important. Therefore, in order to avoid this effect we employ (17) and (18) instead which supply more stable cumulative mean rates.

The transmission rate assigned to the streaming videos at frame i can then be calculated with (19), and the remaining transmission capacity is assigned to the conversational videos,

$$r_{\mathrm{SV}}^{i} = \frac{1}{K \cdot (i-1)} \cdot \sum_{k=1}^{K} \sum_{j=1}^{i-1} R_j^k, \tag{17}$$

$$r_{\mathrm{CV}}^{i} = \frac{1}{N \cdot (i-1)} \cdot \sum_{n=1}^{N} \sum_{j=1}^{i-1} R_j^n, \tag{18}$$

$$S_{\mathrm{SV}}^{i} = R_{\mathrm{out}} \cdot \frac{r_{\mathrm{SV}}^{i} \cdot B_{\mathrm{SV}}^{i}}{r_{\mathrm{SV}}^{i} \cdot B_{\mathrm{SV}}^{i} + r_{\mathrm{CV}}^{i} \cdot B_{\mathrm{CV}}^{i}}. \tag{19}$$

Here r_{SV}^{i} and r_{CV}^{i} are, respectively, the mean incoming rates of the streaming videos and the conversational videos from the beginning until frame $i-1$. B_{SV}^{i} and B_{CV}^{i} denote, respectively, the fullness in percentage of the two buffers at the time instance when the ith frames of every stream arrive at the node.

5. COMPLEXITY ANALYSIS

In this section, we discuss the computational complexity and the storage requirements of the two RD-optimized frame dropping strategies proposed in this paper for streaming and conversational videos, respectively.

5.1. Memory cost

PRD/PRED are based on the static priority labels assigned to every frame, which are included in the bitstream, so there is no additional storage cost for PRD/PRED.

As shown in [19], the *distortion matrix* has $(1/2)N_G * (3 + N_G/(N_B + 1))$ entries for a GOP consisting of N_G frames with N_B B-frames between two P- or I-frames. However, less entries need to be stored in reality as in the cost function in (2), we only consider the overall (cumulative) additional distortion caused by selecting a dropping choice for a current frame. In particular, as explained in Section 2.3, there are at most four possible dropping decisions that can be made for each frame. Therefore, no more than four distortion values need to be associated to one video frame. Furthermore, given that the additional distortion is zero when nothing is dropped, there are only two remaining choices for which distortion values need to be stored in the case of P- and B-frames, and three such values in the case of I-frames. Hence, the distortion matrix can be compacted into $2 * N_G + 1$ entries for each GOP.

When the *hint track* framework based on the DC^0 distortion chain model is employed, frame drop patterns are constructed by considering every video frame independently [21]. Therefore, only L entries for a video stream with L frames need to be stored and sent as side information in the case of *hint track* DC^0. However, when higher-order distortion chain models are used in the *hint track* framework, the

Table 1: Construction of the test sequences.

Test sequence	Carphone	Claire	Foreman	Grandma	Miss America	Mother Daughter	Salesman	Suzie
# of frames	380	270	400	300	150	320	220	150
SV_1_20	5	1,4	3	—	2	6	—	—
SV_2_22	—	5,6	—	4	3	1	2	—
SV_3_24	—	4	2	6	3	1,5	—	—
SV_4_26	—	3	—	4	1	2,6	5	—
CV_1	1	3	4	—	—	5	—	2
CV_2	3	—	1	2,6	5	—	—	4
CV_3	—	2	—	3,6	—	1,5	4	—
CV_4	6	3	—	—	2	4	1,5	—

memory requirements are more demanding. In particular, the number of distortion values that need to be stored increases polynomially with the order of the distortion chain. For example, $L*(L-1)/2$ entries need to be stored in the case of *hint track* DC^1.

The rate information that needs to be stored is the same in both approaches and comprises the sizes of the video frames, as explained in Sections 2.2 and 3.1. Hence, there are L rate entries for L frames. Furthermore, for a given drop pattern, the associated rate reduction represents the sum of the sizes of the dropped frames in the case of the *hint track* framework, while for the *distortion matrix* approach, this quantity includes in addition the sizes of all dependent frames.

5.2. Computational complexity

The cost function-based frame dropping strategy for streaming video offers up to four possible dropping choices for every frame, which leads to an upper bound of 4^K drop patterns for K incoming streaming videos. As we need to calculate the distortion and rate saving for every drop pattern to select the optimal one, the computational complexity is very high in this case. However, in the cost function in (2) only one $\lambda(n)$ is used at every frame slot. For this reason, minimizing $J(n)$ is the same as minimizing $J_p^k(n)$ separately for each stream. Hence, we can rewrite the cost function as

$$\operatorname{argmin} J(n) = \sum_{k=1}^{K} \operatorname{argmin} J_p^k(n), \quad \text{for } p \in P(n), \text{ for } p \in P^k(n) \tag{20}$$

so that the maximum number of possible drop patterns is reduced to $4*K$. Including the computation of $\lambda(n)$, the total calculation complexity is $O(8*K*L)$ for K videos, each of length L frames. Please note that this is the worst case that in practice is actually unattainable. That is because frame dropping decisions are only made when the buffer fullness reaches a predefined threshold. Furthermore, dropping decisions affecting future frames reduce the number of prospective drop patterns when the optimization is performed again, at the next frame slot.

With the utility-based approach for conversational video, the individual frames are considered independently for the DC^0 model. Therefore, there are only two possible dropping choices for every frame, to drop or not to drop and the resulting overall computational complexity is $O(N*\log(N)*L)$, where $N*\log(N)$ is the cost for sorting the importance at every frame slot. In particular, with the classification buffer in the hybrid scenario, assume that W frames are in the temporal buffer and need to be sorted according to their distortion-per-bit utility, the resulting computational complexity is $O((W+N)*\log(W+N)*L)$ in this case.

6. SIMULATION RESULTS

In this section, we examine the performance of several frame dropping strategies for streaming and conversational videos. First, we show the improvement achieved by the proposed RD-optimized frame dropping strategies introduced in Sections 2 and 3. Then, the performance of the frame dropping optimizer from Section 4 that considers both streaming and conversational videos is evaluated.

The videos employed in our simulation experiments are encoded with the H.264 MPEG-4/AVC codec [24] with a frame rate of 25 Hz. Long test sequences are generated by concatenating several short test sequences. For streaming video, each short sequence is appended at the tail of the resulting long sequence in integer multiples of the associated GOP length. This means that a number of frames at the end of a short sequence may be left out if its length is not an integer multiple of the GOP size. In Table 1, the entries in the first row under the names of the short sequences represent their corresponding lengths in number of frames. For example, the sequence *Carphone* is 380 frames long. Furthermore, the entries in each of the following rows, when moving towards the bottom of Table 1, represent the relative order of concatenation of the short sequences, for each of the resulting long sequences. For example, the long sequence SV_1_20 represents a concatenation of the short sequences: *Claire*, *Miss America*, *Foreman*, *Claire*, *Carphone*, *Mother*&*Daughter*, in this order. The test sequences are named SV_X_YY for streaming video, where YY stands for the length of the GOP and X is the index of the video. The number of B-frames between two P- or I-frames is set to be 1 in our experiments. For conversational

Table 2: Encoding characteristics of the test sequences.

Name	SV_1	SV_2	SV_3	SV_4	Sum/Avg
Rate (kbps)	92.44	67.99	69.55	50.60	280.58
Y-PSNR (dB)	38.63	38.32	38.10	38.24	38.33
Name	CV_1	CV_2	CV_3	CV_4	—
Rate (kbps)	122.07	119.06	67.81	116.42	425.36
Y-PSNR (dB)	37.57	37.26	37.36	37.69	37.47

videos, the name is CV_X. The encoding structure for conversational videos is IPPP... with an intraupdate interval of $M = 18$.

Table 2 summarizes the encoding (rate and quality) characteristics of the eight test sequences employed in our experiments. Furthermore, the entries in the last column in Table 2 represent, respectively, the sum of the mean rates and the average PSNR values for each of the two categories: streaming video and conversational video. As shown in Section 5.1, one GOP streaming video with N_G frames needs $2*N_G+1$ and N_G entries for the distortion and the rate information, respectively. With the assumption that each entry needs two bytes, each frame in SV_1 needs on average 6.1 bytes, which results in 0.152 kbps overhead traffic. Compared to the bitrate of the video stream at 92.44 kbps, this less than 0.2% overhead can be ignored. For the conversational video, the number of distortion entries is even smaller and compared to the bitrate of the video stream the overhead for the side information is insignificant.

In order to avoid the prospective loss of the very first I-frame for every test sequence, we assume that these frames have been forwarded by the network node and that all dropping decisions are made after the arrival of the second frame of each stream. For this reason, we set 7.5 KB out of 16 KB (total buffer size) as the initial buffer load in the case of streaming video when the frame dropping process starts. For conversational video, because of the strict delay constraint, the buffer size is set to be 5 KB. Again, the influence of the first I-frames is ignored and the initial buffer load is set to be 0 byte. The relation between the buffer size and the corresponding frame dropping performance and the decisions have been investigated in [23].

In our simulations, we measure the performance of a frame dropping strategy through the luminance (Y) PSNR values of the reconstructed video frames averaged over all videos. This quantity is computed as

$$\overline{\text{PSNR}} = \frac{1}{K}\sum_{k=1}^{K}\left(\frac{1}{L}\sum_{i=1}^{L}10\log_{10}\left(\frac{255^2}{\text{MSE}^k(i)}\right)\right), \quad (21)$$

where K is the number of videos, each of length L frames, and $\text{MSE}^k(i)$ is the MSE distortion for frame i of video k.

6.1. Threshold settings for PRED

The implementation of PRED is straightforward and the only important point here is to select the proper thresholds for the random dropping of B-frames (T_1) and P-frames (T_2). In our experiments, the four streaming videos introduced in the previous section are employed as test videos. Note that no conversational videos are employed in the experiments, as no static priority labels can be established for them ahead of time. The operation of PRED on such content reduces to random dropping without priorities. In our experiments here, we go through all the possible values for T_1 from 30% to 100% of the buffer fullness and T_2 is always bigger than or equal to T_1.

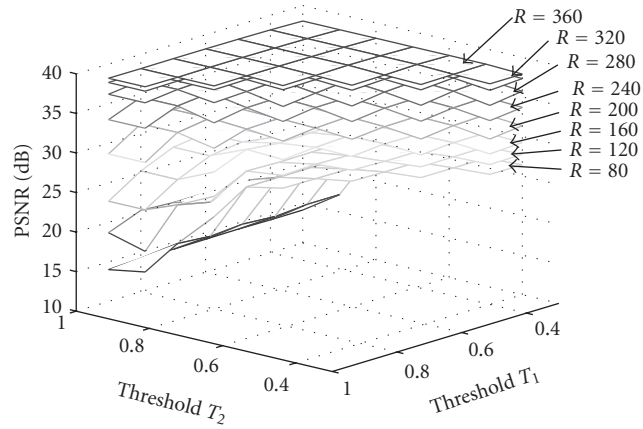


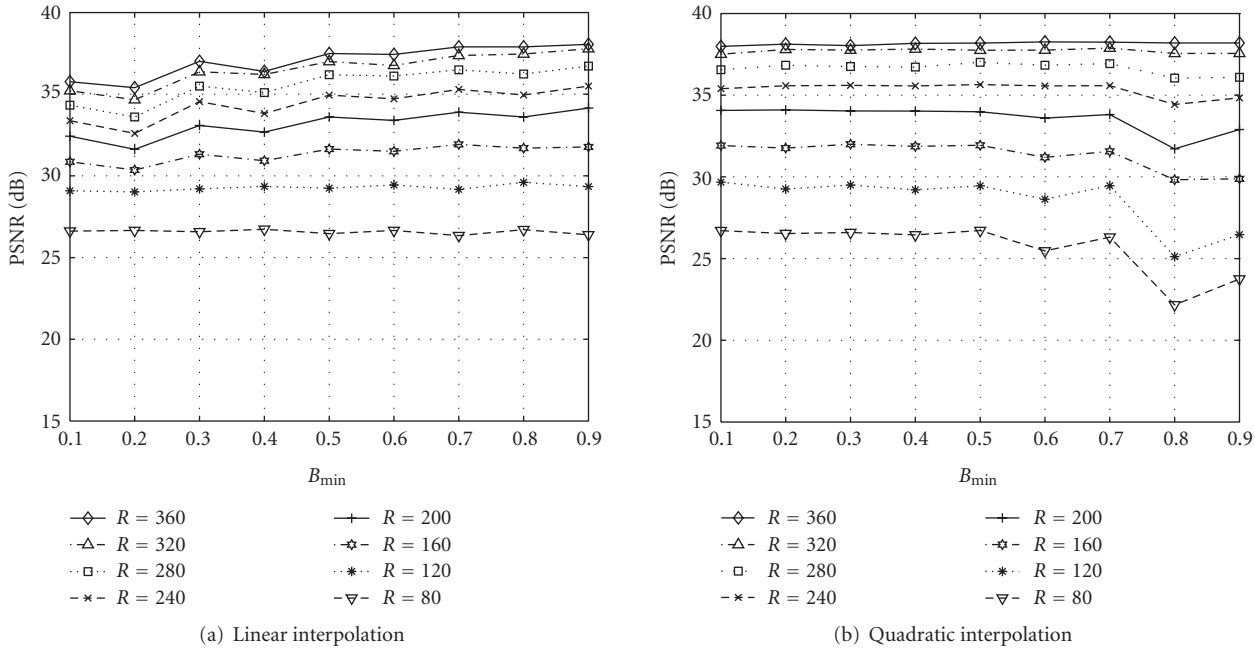
Figure 6: Setting of PRED thresholds, R represents the outlink rate in kbps.

In Figure 6, we show the average reconstruction quality (Y-PSNR) of the four streaming videos as a function of T_1 and T_2 at different outlink transmission rates, which are represented with different surfaces in the figure. In principle, the higher the transmission rate, the higher the reconstruction video quality. However, we can see that at low rates, the performance surface is not flat and a big performance drop can be observed when large values for T_1 and T_2 are selected. The performance at higher rates is more stable, as the observed reduction in video quality due to an improper selection of thresholds does not exceed 1∼1.5 dB here. The upper and lower performance bounds of PRED are shown in Table 3, which are the highest and lowest points on each surface in Figure 6. The normalized rate is the percentage of available transmission rate versus the mean rate of all users. We can see from the table that a large performance gap exists between the two bounds for the case when the transmission rate on the forwarding link is much smaller than the mean aggregate source rate of the videos. However, it is not easy in practice always to select the optimal thresholds such that the upper performance bound is achieved.

As described in Section 2.1, we can have different dropping thresholds for P-frames depending on their position in a GOP. For example, if we set the start point for dropping frames to be at 50% of the buffer size, and then each successive dropping threshold to be associated with a further increment of 5%, we achieve the upper performance bound for the case when only two thresholds are used. This means that more accurate frame dropping decisions can be made, when finer priority steps in terms of frame dropping are employed.

Table 3: Performance bounds of PRED.

Rate (kbps)	80	120	160	200	240	280	320	360
Normalized rate	0.285	0.428	0.570	0.713	0.855	0.998	1.140	1.283
$PSNR_{max}$ (dB)	26.43	27.75	29.37	31.91	35.18	36.83	37.82	38.23
$PSNR_{min}$ (dB)	14.03	16.67	21.68	28.17	32.77	34.73	36.50	37.47



(a) Linear interpolation

(b) Quadratic interpolation

Figure 7: Performance of DM-based frame dropping, R represents the outlink rate in kbps.

6.2. Cost function-based RD-optimized frame dropping

In the following, we examine the influence of λ on the performance of cost function-based frame dropping for the two interpolation methods introduced in Section 2.3. In Figures 7(a) and 7(b), we show the results for the cost function-based frame dropping strategy when using linear and quadratic interpolations for the multiplier λ, respectively. In all simulations, we fix B_{max} to be 100% of the buffer size.

Quadratic interpolation exhibits a degraded quality when very high values for B_{min} are selected at very low outlink rates, as shown in Figure 7(b). This is because quadratic interpolation leads to aggressive frame dropping decisions when the buffer fullness approaches B_{max} and is far away from B_{min}. Setting B_{min} to be bigger than 0.8 results in late dropping of less important frames and which in turn causes unnecessary loss of some frames with high importance. The curves are smooth and flat when B_{min} is smaller, as the dropping decision is very moderate when the buffer is lightly loaded. When linear interpolation is used, small values for B_{min} at high outlink rates lead to unnecessary dropping of some frames with low importance. To summarize, selecting B_{min} larger than 0.5 is fine for linear interpolation and for quadratic interpolation, B_{min} should be selected smaller than 0.6. By selecting B_{min} between 0.5 and 0.6, we obtain good results for both schemes.

6.3. Performance comparison among all frame dropping schemes for streaming video

In this section, we compare the performance of the frame dropping schemes for streaming video examined in this paper, as a function of the forwarding data rate on the outgoing link. In Figure 8, PRD denotes the priority-based random frame dropping. PRED here fixes the thresholds T_1 and T_2 to be 70% and 90%, respectively, of the buffer fullness, while the PRED_UB curve in Figure 8 corresponds to the upper bound from Table 3. Our proposed RD-optimized cost function-based dropping strategy that uses the *distortion matrix* as the side information is shown as the CF_DM curve in the figure, where B_{min} is selected to be 0.6.

PRD performs the worst at all the rates, as can be seen from Figure 8. PRED also shows a poor performance at low link rates, while PRED_UB performs much better by the proper selection of dropping thresholds. CF_DM

outperforms all other schemes as a result of its accurate distortion estimation and dynamic adjustment of the dropping aggressiveness according to the buffer fullness level.

6.4. Utility-based frame dropping for conversational video

We compare our utility-based frame dropping for conversational video with the pure random dropping in a round robin fashion. When a video packet arrives, if the outgoing link buffer can still hold it, the packet is put into the buffer, otherwise, this packet is simply dropped. For the utility-based approach, when N new incoming frames arrive at the node and the buffer cannot hold all of them, they are sorted according to their utility and put into the buffer one after another until the buffer is full.

Table 4 shows the averaged PSNR values of the luminance (Y) component for the four test conversational videos at different outgoing link rates. The mean score of the four videos (boldfaced numbers) presents the overall reconstruction quality. The utility-based frame dropping outperforms the random frame dropping in the range of middle-to-high rates, because at very low rates, consecutively dropping of a large number of frames leads to an inaccurate estimation of distortion. However, if we look at the performance of individual users, it is more fair by using the utility-based approach (maximum difference from 0.9 dB~5.5 dB) compared to the pure random dropping approach (maximum difference from 3.8 dB~10 dB). Therefore, in addition to the overall quality, our approach also shows a good characteristic with respect to the fairness among users.

6.5. Joint optimization for streaming and conversational videos

In this experiment, we compare our joint optimizer for streaming and conversational videos with a reference scheme that uses PRED/RR for these two types of video applications, respectively. In particular, streaming video provides three types of static priority labels, as explained earlier. Therefore, in the reference scheme we can perform PRED on the streaming videos by early dropping of B- or P-frames. On the other hand, for conversational video, all the frames, except the very first one, are P-frames and hence there is no static priority difference among them. Therefore, when multiple frames arrive simultaneously at the network node, a simple round-robin scheme (over the conversational videos to which these frames belong) is employed to determine how many of them can be placed into the corresponding buffer for conversational video.

Our proposed optimizer uses the *distortion matrix* for streaming video and *hint tracks* for conversational video. In the case of conversational video, we employ (11) and (12) to estimate the overall distortion associated with the dropping of a single frame, as explained in Section 3.1. In the equations, the equivalent intraupdate period M is set to be 18 frames and the attenuation factor r is set to be 0.997. Finally, for comparison purposes we also consider the hypothetical

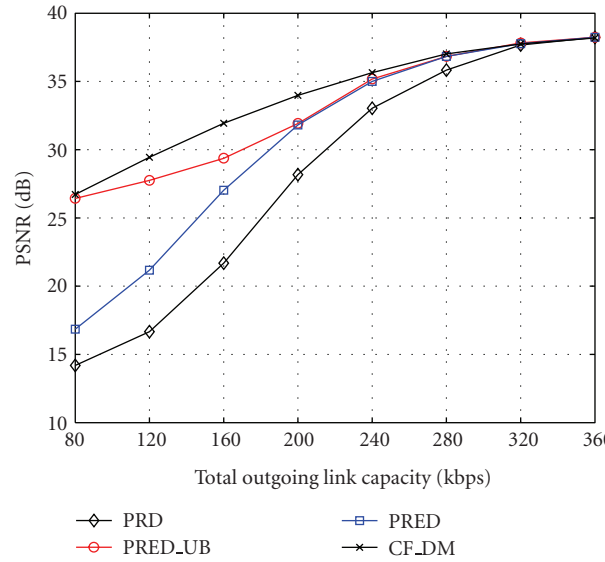


Figure 8: Performance comparison of different frame dropping schemes for streaming video.

case when the distortion incurred by dropping frames can also be precalculated for conversational video.

Figure 9 shows the performance improvement achieved by the proposed RD-optimized strategy for dropping frames from both streaming and conversational videos. Several instances of the proposed optimizer are considered in Figure 9. In particular, RD_FIX denotes the proposed optimizer with fixed resource assignment of 40% to the streaming videos and 60% to the conversational videos. Note that this assignment corresponds to the overall average data rates for these two types of videos. Furthermore, RD_BUF and RD_RAT in the figure represent the buffer fullness-based and the short-term mean-rate-based scheduling strategies introduced in Section 4.2, respectively. In the case of RD_RAT, F in (13) and (14) is set to be 10 frame slots in this experiment.

First, it can be seen that when the outgoing link rate is larger than the mean $B_{\min}$ incoming rate (when the normalized rate is larger than 1), the performances of the RD-optimizer and PRED/RR are similar. However, there is still a performance improvement of 1 dB at 900 kbps. This is because even at this rate, frame dropping from the conversational videos need to occur in PRED/RR, whenever the incoming data rate of the video streams peaks, as the small buffer for conversational videos cannot hold too many frames at once. The RD-optimizer deals more successfully with this situation, since the optimized frame dropping has more opportunities to drop the least important frames even if they have been in the classification buffer waiting to be scheduled. At the same time, the dynamic resource assignment saves away some spare transmission slots from the streaming videos, that can be appropriately reallocated to the conversational videos afterwards, as explained in Section 4.2. When the outgoing rate is smaller than the total traffic rate, an improvement of around 3 dB is observed, as shown in Figure 9.

Table 4: Comparison of utility-based dropping and random dropping.

Outlink rate (kbps)	120	180	240	300	360	420	480	540
Normalized rate	0.282	0.423	0.564	0.705	0.846	0.987	1.128	1.270
Utility-based frame dropping								
CV1(dB)	15.01	19.60	22.53	27.23	29.96	32.20	34.01	36.07
CV2(dB)	15.60	21.07	23.30	28.46	31.11	33.03	35.12	36.45
CV3(dB)	20.50	21.30	23.34	27.23	30.39	32.76	34.25	36.06
CV4(dB)	18.52	21.37	25.12	27.83	31.40	34.02	35.82	36.93
Mean(dB)	**17.41**	**20.83**	**23.57**	**27.69**	**30.71**	**33.00**	**34.80**	**36.38**
Random frame dropping								
CV1(dB)	13.68	19.96	21.09	25.25	26.57	29.75	31.71	35.17
CV2(dB)	16.40	16.64	21.15	23.52	28.93	28.60	34.43	33.34
CV3(dB)	16.44	26.26	26.25	29.17	31.69	34.99	34.11	37.02
CV4(dB)	23.67	24.49	28.75	29.97	32.61	33.84	36.38	37.21
Mean(dB)	**17.55**	**21.84**	**24.31**	**26.98**	**29.95**	**31.79**	**34.16**	**35.69**

Table 5: Assignment of forwarding date rate.

Total_rate (kbps)	200	300	400	500	600	700	800	900
Normalized rate	0.283	0.425	0.567	0.708	0.850	0.992	1.133	1.275
Assigned_rate_SV (kbps)	72.86	104.82	140.40	180.10	227.64	258.44	272.88	279.18
(%)	(0.36)	(0.35)	(0.35)	(0.36)	(0.38)	(0.37)	(0.34)	(0.31)
Assigned_rate_CV (kbps)	127.14	195.18	259.60	319.90	372.36	441.56	527.12	620.82
(%)	(0.64)	(0.65)	(0.65)	(0.64)	(0.62)	(0.63)	(0.66)	(0.69)

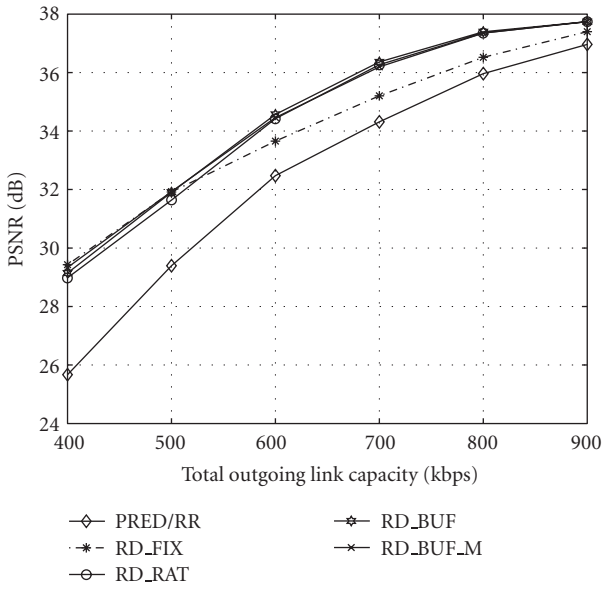


Figure 9: Performance comparison of the proposed RD-optimizer and PRED/RR for streaming and conversational videos.

Furthermore, at low rates, the performances of RD_FIX, RD_BUF, and RD_RAT are almost the same. However, at high rates the schemes with dynamic resource assignment perform much better, because reassigning some of the resources from the streaming video buffer to the conversational video buffer will not influence the quality of streaming video significantly, as these resources are typically saved when the low data rate sections of the incoming streams occur at the node. But with fixed resource allocation, these unused resources from the streaming video are wasted, which leads to degraded performance at high outgoing link rates compared to the case of dynamic resource assignment. Table 5 gives the assigned transmission resources to the streaming videos and conversational videos when the buffer fullness-based scheduling strategy is used. More transmission resources are assigned to the conversational videos compared to their mean bitrates. This is the consequence of the small size of the classification buffer due to the tight delay constraint of the conversational videos.

Finally, precomputed *hint tracks* for conversational video are not available in practice, but here we compute them anyhow in order to examine if the approximation from (11) and (12) leads to accurate results. Our experiments show that precomputed *hint tracks* (RD_BUF_M) for the conversational videos and the approximation (RD_BUF) obtained using (11) and (12) lead to almost identical performance results, as can be seen from Figure 9. The estimation bias from the model in (11) and (12) does not affect the results, because the relative values of the distortion-per-bit utility among the individual frames are preserved in either case.

7. CONCLUSIONS

We have presented RD-optimized frame dropping strategies for streaming and conversational video applications that can be applied at active network nodes. The proposed techniques employ side information about the packetized video content that is extracted at compression time and that is sent along the video streams. The only additional information that the techniques need to operate at an active node is the fullness of the outlink buffer and the mean traffic rate of the video streams passing through the node. It is shown through simulations that a significant improvement in video quality is achieved over previous approaches, by a judicious selection of side information and optimized frame dropping strategy.

ACKNOWLEDGMENT

This work has been supported in part by DFG Grant STE 1093/3-1.

REFERENCES

[1] A. Vetro, C. Christopoulos, and H. Sun, "Video transcoding architectures and techniques: an overview," *IEEE Signal Processing Magazine*, vol. 20, no. 2, pp. 18–29, 2003.

[2] J. Xin, C.-W. Lin, and M.-T. Sun, "Digital video transcoding," *Proceedings of the IEEE*, vol. 93, no. 1, pp. 84–97, 2005.

[3] A. Vetro, J. Cai, and C. W. Chen, "Rate-reduction transcoding design for wireless video streaming," *Journal of Wireless Communications and Mobile Computing*, vol. 2, no. 6, pp. 625–641, 2002.

[4] W. Feng, M. Liu, B. Krishnaswami, and A. Prabhudev, "A priority-based technique for the best-effort delivery of stored video," in *Multimedia Computing and Networking*, vol. 3654 of *Proceedings of SPIE*, pp. 286–300, December 1999.

[5] Y. Lu and K. J. Christensen, "Using selective discard to improve real-time video quality on an ethernet local area network," *International Journal of Network Management*, vol. 9, no. 2, pp. 106–117, 1999.

[6] H. Cha, J. Oh, and R. Ha, "Dynamic frame dropping for bandwidth control in MPEG streaming system," *Multimedia Tools and Applications*, vol. 19, no. 2, pp. 155–178, 2003.

[7] R. Mahajan, S. Floyd, and D. Wetherall, "Controlling high-bandwidth flows at the congested router," in *Proceedings of IEEE International Conference on Network Protocols (ICNP '01)*, pp. 192–201, Riverside, Calif, USA, November 2001.

[8] J. Kritzner, M. Kampmann, and U. Horn, "Comparison of frame dropping strategies for adaptive video streaming," in *Proceedings of International Picture Coding Symposium (PCS '04)*, pp. 325–329, San Francisco, Calif, USA, December 2004.

[9] J. Kritzner, U. Horn, and M. Kampmann, "Priority generation for video streaming using stream decodability," in *Proceedings of the 14th International Packet Video Workshop (PV '04)*, Irvine, Calif, USA, December 2004.

[10] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," Tech. Rep. MSR-TR-2001-35, Microsoft Research, Mountain View, Calif, USA, February 2001.

[11] F. Zhai, R. Berry, T. N. Pappas, and A. K. Katsaggelos, "A rate-distortion optimized error control scheme for scalable video streaming over the internet," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME '03)*, Baltimore, Md, USA, July 2003.

[12] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74–90, 1998.

[13] H. Schwarz and T. Wiegand, "An improved MPEG-4 coder using lagrangian coder control," in *Proceedings of the 4th International ITG Conference on Source and Channel Coding (SCC '02)*, Berlin, Germany, January 2002.

[14] R. Zhang, S. L. Regunathan, and K. Rose, "End-to-end distortion estimation for RD-based robust delivery of pre-compressed video," in *Proceedings of Asilomar Conference on Signals, Systems and Computers (ASILOMAR '01)*, vol. 1, pp. 210–214, Monterey, Calif, USA, November 2001.

[15] E. Masala, H. Yang, K. Rose, and J. C. De Martin, "Rate-distortion optimized slicing, packetization and coding for error resilient video transmission," in *Proceedings of IEEE Data Compression Conference (DCC '04)*, pp. 182–191, Snowbird, Utah, USA, March 2004.

[16] I. Bouazizi, "Size-distortion optimized proxy caching for robust transmission of MPEG-4 video," in *Proceedings of International Workshop on Multimedia Interactive Protocols and Systems (MIPS '03)*, no. 2899, pp. 131–142, Napoli, Italy, November 2003.

[17] P. Yin, A. Vetro, M. Xia, and B. Liu, "Rate-distortion models for video transcoding," in *Proceedings of Conference on Image and Video Communications and Processing*, vol. 5022 of *Proceedings of SPIE*, pp. 479–488, Berlin, Germany, May 2003.

[18] J. Chakareski and P. Frossard, "Rate-distortion optimized bandwidth adaptation for distributed media delivery," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME '05)*, vol. 2005, pp. 763–766, Amsterdam, Netherlands, July 2005.

[19] W. Tu, W. Kellerer, and E. Steinbach, "Rate-distortion optimized video frame dropping on active network nodes," in *Proceedings of the 14th International Packet Video Workshop (PV '04)*, Irvine, Calif, USA, December 2004.

[20] J. Chakareski, J. Apostolopoulos, S. Wee, W.-T. Tan, and B. Girod, "R-D hint tracks for low-complexity R-D optimized video streaming," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME '04)*, vol. 2, pp. 1387–1390, Taipei, Taiwan, June 2004.

[21] J. Chakareski, J. G. Apostolopoulos, S. Wee, W.-T. Tan, and B. Girod, "Rate-distortion hint tracks for adaptive video streaming," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 10, pp. 1257–1269, June 2005.

[22] Y. J. Liang, J. G. Apostolopoulos, and B. Girod, "Analysis of packet loss for compressed video: does burst-length matter?" in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '03)*, vol. 5, pp. 684–687, Hong kong, April 2003.

[23] W. Tu, J. Chakareski, and E. Steinbach, "Rate-distortion optimized frame dropping and scheduling for multi-user conversational and streaming video," in *Proceedings of the 16th International Packet Video Workshop (PV '06)*, Hangzhou, China, April 2006.

[24] HHI. The JVT H.264/MPEG-4 AVC reference software. http://iphome.hhi.de/suehring/tml/index.htm.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2008, Article ID 543674, 10 pages
doi:10.1155/2008/543674

Research Article

A Theoretical Framework for Quality-Aware Cross-Layer Optimized Wireless Multimedia Communications

Song Ci,[1] Haohong Wang,[2] and Dalei Wu[1]

[1] *Department of Computer and Electronics Engineering, University of Nebraska-Lincoln, NE 68182, USA*
[2] *Marvell Semiconductors, Santa Clara, CA 95054, USA*

Correspondence should be addressed to Song Ci, sci@engr.unl.edu

Received 20 May 2007; Revised 5 September 2007; Accepted 5 November 2007

Recommended by Jianwei Huang

Although cross-layer has been thought as one of the most effective and efficient ways for multimedia communications over wireless networks and a plethora of research has been done in this area, there is still lacking of a rigorous mathematical model to gain in-depth understanding of cross-layer design tradeoffs, spanning from application layer to physical layer. As a result, many existing cross-layer designs enhance the performance of certain layers at the price of either introducing side effects to the overall system performance or violating the syntax and semantics of the layered network architecture. Therefore, lacking of a rigorous theoretical study makes existing cross-layer designs rely on heuristic approaches which are unable to guarantee sound results efficiently and consistently. In this paper, we attempt to fill this gap and develop a new methodological foundation for cross-layer design in wireless multimedia communications. We first introduce a delay-distortion-driven cross-layer optimization framework which can be solved as a large-scale dynamic programming problem. Then, we present new approximate dynamic programming based on significance measure and sensitivity analysis for high-dimensional nonlinear cross-layer optimization in support of real-time multimedia applications. The major contribution of this paper is to present the first rigorous theoretical modeling for integrated cross-layer control and optimization in wireless multimedia communications, providing design insights into multimedia communications over current wireless networks and throwing light on design optimization of the next-generation wireless multimedia systems and networks.

1. INTRODUCTION

In recent years, ubiquitous computing devices such as laptop computers, PDAs, smart phones, automotive computing devices, and wearable computers have been ever growing in popularity and capability, and people have begun more heavily to rely on these ubiquitous computing devices. Therefore, there has been a strong user demand for bringing multimedia streaming to the devices such as iTunes, PPLive, MSN, and YouTube. However, bringing delay-sensitive and loss-tolerant multimedia services based on the current wireless Internet is a very challenging task due to the fact that the original design goal of the Internet is to offer simple delay-insensitive loss-sensitive data services with little QoS consideration. Therefore, this shift of design goal urges us to rethink the current Internet architecture and develop a new design methodology for multimedia communications over the current and future wireless Internet. So far, cross-layer design has been thought as one of the most effective and efficient ways to provide quality of service (QoS) over wireless networks, and it has been receiving many research efforts. The basic idea of cross-layer design is to fully utilize the interactions among design variables (system parameters) residing in different network functional entities (network layers) to achieve the optimal design performance of time-varying wireless networks.

In order to achieve the global optimality of cross-layer design, we need to consider design variables and the interactions among them as much as possible. However, more does not necessarily mean better. The more design variables we consider, the more difficult is orchestrating a large number of design variables to make them work harmonically and synergetically. From the point of view of nonlinear optimization, the number of design variables increases and the size of state space of the objective function will increase exponentially, making the optimization

problem unmanageable. To overcome this problem, one often used approach is to reduce the size of the problem at the system modeling phase and then solve the simplified problem by using various optimization algorithms such as gradient-based local search, linear/nonlinear programming, genetic algorithm, exhaustive search, and heuristic-based approach like artificial neural networks.

However, reducing a high-dimensional cross-layer optimization problem to a low-dimensional problem in the system modeling phase raises a series of questions:

(1) how to evaluate the fidelity of the simplified problem compared with the problem as what it should be,

(2) how to evaluate the quality of the suboptimal solution to the global optimum,

(3) how to evaluate the robustness of the solution, that is, whether the solution can guarantee the predictable sound results at all possible circumstances.

Unfortunately, at the time of this writing, we have no clear answers to all these three questions.

Moreover, reducing the size of the problem in the problem formulation means that only part of the current Internet architecture can be considered, causing a shift of the design goal of multimedia services from the best user experience to some layer-specific performance metrics such as distortion at the application layer, delay at the network layer, and goodput at the MAC/PHY layer. This shift of design goal may cause an "Ellsberg paradox," where each individual design variable makes good decisions for maximizing the objective function. But the overall outcome violates the expected utility function. In other words, breaking a big problem into several smaller problems in the system modeling phase can only increase the solvability of the original problem but cannot guarantee that it is a good solution. The "Ellsberg paradox" also tells us that the traditional additive measure such as probability measure may no longer hold in the context of cross-layer design due to the possible strong coupling (interdependency) among design variables. At the point of this writing, there have been many researches done on interdependency modeling in the context of cross-layer design, but they are mostly qualitative rather than quantitative approaches, and their applications are still within the scope of local cross-layer optimization.

We argue that all aforementioned difficulties in the area of cross-layer design of wireless multimedia communications are due to lacking of methodological foundation and in-depth understanding of cross-layer behavior. Our goal is to provide a flexible yet scalable theoretical cross-layer framework to accommodate all major design variables of interest, spanning from application layer to physical layer, for delay-bounded multimedia communications over wireless single/multihop networks. We start from proposing an integrated cross-layer framework for the best user experience. Although the engineering side of cross-layer design is not the main focus of this paper, we still briefly discuss how to utilize the methodological foundation to achieve real-time multimedia communications through a fast algorithm for large-scale global cross-layer optimization based on quantitative significance measure and sensitivity analysis.

The rest of the paper is organized as follows. We briefly introduce the related work in Section 2. In Section 3, we present a unified theoretical cross-layer framework for wireless multimedia communications based on link adaptation, rate-distortion theory, and dynamic programming. A further discussion of how to apply the proposed methodological foundation for real-time applications is made in Section 4, where new feature-based approximate dynamic programming is introduced, followed by the conclusion in Section 5.

2. RELATED WORK

In literature, topics involving video delivery over multihop networks such as video coding, multihop routing, QoS provisioning, link adaptation are separately studied. Therefore, the corresponding video compression efficiency and the transmission efficiency are also separately optimized. In prediction mode, selection of video coding, periodic intracoding of whole frames [1], continuous blocks [2], or random blocks [3] has been firstly proposed. These methods apply intracoding uniformly to all the regions of the frame. Then, "content-adaptive" methods are proposed to apply frequent intra-update to regions that undergo significant changes [4], or where a rough estimate of decoder error exceeds a given threshold [5, 6]. A significant advance over the above early heuristic mode switching strategies is the rate-distortion (RD) optimized mode selection. The RD optimized mode selection is achieved by choosing a mode that minimizes the quantization distortion between the original frame/macroblock and the reconstructed one under a given bit budget [7, 8]. However, the encoders in [7, 8] have no capability to accurately estimate the overall distortion. So, the selected prediction mode is not necessarily optimal. The work in [9] proposes an algorithm to optimally estimate the overall distortion of decoder frame reconstruction due to quantization, error propagation, and error concealment. The accurate estimate is integrated into a rate-distortion-based framework for optimal switching between intracoding and intercoding modes per macroblock. However, the joint optimization between mode selection and video transmission parameters under wireless environment is not addressed in [9]. The work in [10] presents an end-to-end approach to solve the fundamental problem of RD optimized mode selection over packet-switched networks, but it only aims at Internet peer-to-peer video communication.

In routing for video delivery in multihop networks, an application-centric cross-layer approach has been proposed to formulate an optimal routing problem for multiple description video communications [11]. Physical and MAC layer dynamics of wireless links are translated into network layer parameters. The application layer performance, that is, average video distortion, is considered as the function of network layer performance metrics, for example, bandwidth, loss, and path correlation. But the routing metric, that is, average video distortion, is roughly computed from a simple rate-distortion model without discussion on selection of source coding parameters. The same problem goes to [12]

and [13] even though optimal paths are selected optimizing the quality under various constraints. In addition, in [13] exhaustive algorithm is adopted for the determination of the cross-layer optimized mesh-network path selection, which may incur heavy computational load and make it unpractical for real applications.

Cross-layer optimized wireless video has been studied from different aspects, such as cross-layer architecture [14, 15], content analysis [16–18], video compression and RD optimization [2–4, 6–10, 19–21], source packetization [22, 23], QoS provisioning [24–26], application-centric routing [11–13, 27], queueing and scheduling [28–31], energy efficiency [32, 33], and link adaptation [34, 35]. To reach a global optimality at the level of frame or video sequence rather than at the level of packet, we need to evaluate the overall distortion and the effect of packet pipelining in a network on the total delay of a frame or a video sequence. To the best of our knowledge, although some works focus on the cross-layer design for video delivery over multihop wireless networks, there is still no substantial work that can reach such kinds of global optimality.

In wireless video, optimization has to be done over multiple source coding units, such as frames and pixel blocks, for the best reconstructed video quality. There is "*only one exact method for solving problems of optimization over time; in the general case of nonlinearities with random disturbance, it is dynamic programming (DP)*" [36]. However, the biggest challenge of applying dynamic programming in practical large-scale problems is *curse of dimensionality* [37], where the size of state space normally increases exponentially with the number of control variables increasing. Therefore, the most sensible way is to map a huge state space $\Re^n$ to a much smaller feature space $\Re^m$ ($m \ll n$), which is called *approximate dynamic programming (ADP)*, also known as neurodynamic programming, adaptive dynamic programming, adaptive critics, or reinforced learning, depending on in which discipline the technique is used [36, 38, 39].

Existing ADP approaches have largely ignored the interdependencies among control variables, which might lead to *loose approximation error bounds*. Nonadditive measure theory was developed to characterize the interactions among control variables [40–43], and it has been widely used in various areas. Choquet integral [44] is regarded as the most effective and efficient way to calculate nonadditive measure and has received a significant amount of research [45–48]. Since nonadditive measure is defined on the power set, fast algorithms [49] have been studied to speed up the calculation process. However, current research on nonadditive measure still focuses on static linear systems with commensurable data [50].

3. A THEORETICAL CROSS-LAYER FRAMEWORK FOR WIRELESS MULTIMEDIA COMMUNICATIONS

3.1. Problem statement

In the protocol stack of multimedia over wireless networks, each layer has one or multiple key system parameters which would significantly impact the overall system performance. At the application layer, tradeoff between rate and distortion is an inherent feature of every lossy compression scheme for video source coding. Prediction mode and quantization level are two critical parameters. At the network layer, routing algorithm is important to find the best delivery path over a single/multihop wireless network. At the data link layer, hybrid automatic repeat request (HARQ), media access control protocols, and packetization are often used to maintain a low packet loss rate. However, the choice of maximum retransmission number is a tradeoff between resultant packet delay and packet loss rate. Note that for real-time multimedia applications, we might not consider HARQ due to strict delay constraints. At the physical layer, adaptive modulation and coding scheme is an important tradeoff between transmission rate and packet loss rate. Furthermore, the end-to-end performance is not completely determined by the parameters of individual layer, but rather by all parameters of all layers. For example, the end-to-end delay consists of propagation delay (determined by the number of hops of the selected path), transmission delay (determined by channel conditions, modulation and channel coding, maximum retransmission number, and source rate), and queueing delay (determined by source rate, transmission rate, and the selected path). Moreover, due to the time-varying nature of wireless channels, each node in the network should be capable of adjusting these parameters quickly to maintain a good instantaneous performance. Clearly, the layer-separated design no longer guarantees an optimal end-to-end performance for multimedia delivery over wireless networks.

3.2. Methodology

We develop a cross-layer framework to optimize multimedia communications over single/multihop wireless networks. In order to demonstrate the main idea of the proposed framework as shown in Figure 1, at the application layer, we implement our framework based on the ITU-T H.264 standard. The rate-distortion tradeoff in video source coding makes it very critical to select suitable video coding parameters such as prediction mode (PM) and quantization parameter (QP). Without losing generality, we consider a multihop wireless network scenario in which all nodes can act as either a source or destination as well as a router for other nodes. To carry out end-to-end delay-bounded multimedia communications, at the network layer, we assume that certain routing protocols are used to come up with the routing table. Then, a quality-aware routing algorithm needs to be developed to select the best multihop path from the source to the destination. Each hop adopts adaptive modulation and coding (AMC) at the physical layer to overcome the adverse effects caused by the time-varying channel condition.

Let us denote by W the number of frames of a video clip, $f_1, f_2, \ldots, f_W$, and let $m_i^1, m_i^2, \ldots, m_i^M$ be the macroblocks of frame f_i. Since each frame is processed in units of macroblock (corresponding to $16*16$ pixels in the original frame), let v_i^j denote the coding parameter vector

of macroblock j in frame i as quantization parameter (QP) and prediction mode choice (I or P frame). Let $B_i^j(v_i^j)$ denote the consumed bits in coding the macroblock m_i^j with the coding parameter vector v_i^j; then the total bits consumed by the frame can be expressed as $F_i = \sum_{j=1}^{M} B_i^j(v_i^j)$.

We assume that the considered multihop network consists of Z nodes $\{N_1, N_2, \ldots, N_z\}$. For any two nodes N_x and N_y, if N_x can directly communicate with N_y, we say that there exists a hop between N_x and N_y. Let $l_i(x \rightarrow y)$ denote the hop between the node N_x and the node N_y. Considering the time-varying nature of the network, let $L^i(x \rightarrow y) | 1 \le x \le Z, 1 \le y \le Z, x \ne y$, denote all the connectivity information within the network when transmitting the frame f_i. Accurate L^i can be obtained from certain routing protocols such as OLSR routing protocol. Let $P^i = \{l_1, l_2, \ldots, l_G\}$ be a path $\{l_1 \rightarrow l_2 \rightarrow \cdots \rightarrow l_G\}$ for transmitting frame f_i from the source node to the destination node. Clearly, there exist $P^i \subseteq L^i$ and $1 \le G_i \le Z - 1$. Let us denote $\{\gamma_i^p\}$ and $\{R_i^p\}$ as the channel SNR and transmission rate of the link l_p with $1 \le p \le G$, $\{A_i^p\}$ and C_i^p as the modulation mode and associated channel coding rate, and $N_{R_i}^p$ as the number of retransmissions. Then, the delay in transmitting the frame f_i on the link l_p can be written as $\{T_i^p(A_i^p, C_i^p, N_{R_i}^p, \gamma_i^p, R_i^p, F_i)\}$. Clearly, the total delay in transmitting the whole video clip can be expressed by

$$T = \sum_{i=1}^{W} \sum_{p=1}^{G_i} T_i^p(A_i^p, C_i^p, N_i^p, \gamma_i^p, R_i^p, F_i). \tag{1}$$

Let $\widetilde{f}_i$ denote the reconstructed ith frame at the receiver side. Using the mean square error as distortion metric, the overall expected distortion for the whole video clip is

$$E[D] = \sum_{i=1}^{W} E[d(f_i, \widetilde{f}_i)]. \tag{2}$$

Note that in this work, $d(\cdot)$ of $E[D]$ can be calculated by any distortion estimation method such as the mean square error (MSE) estimation method and the recursive optimal per-pixel estimate (ROPE) method. Likewise, any error concealment schemes can be used at the receiver side to further enhance the perceivable video quality. Since the formulation discussed above considers W consecutive video frames, the spatial-temporal correlation among frames and macroblocks has been taken into account in the global optimization framework.

Thus, the proposed cross-layer framework for wireless multimedia communications can be formulated as

$$\text{Min } E[D], s.t. : T \le T_{\max}, \tag{3}$$

where $T_{\max}$ is a predefined delay budget for delivering the given video clip.

Recall that the focus of the proposed framework is to jointly find the optimal parameter set for each frame f_i, including the source coding v_i^j, the delivery path P^i, the maximum number of retransmissions N_i^p, and the modulation A_i^p with the associate coding C_i^p. Here, p is the index of each hop on the path P^i. Clearly, the optimal solution for the problem described by (3) can be written as

$$\min_{\{v,P,N_R,A,C\}} \sum_{i=1}^{W} E[d(f_i, \widetilde{f}_i)] \tag{4}$$

with the delay constraint

$$\sum_{i=1}^{W} \sum_{p=1}^{G_i} T_i^p(A_i^p, C_i^p, N_i^p, \gamma_i^p, R_i^p, F_i) \le T_{\max}. \tag{5}$$

Clearly, in (4) we assume that the decoder side has a sufficient size buffer to hold part of the decoded video frames, say, a group of pictures. Given the dramatically fast growing silicon performance and the decreasing size and cost for the memory and silicon, the assumption is reasonable for most scenarios. But when the size of decoder buffer is constrained, (4) would be rewritten as follows:

$$\min_{\{v,P,N_R,A,C\}} E[d(f, \widetilde{f})] \tag{6}$$

with the delay constraint

$$\sum_{p=1}^{G} T^p(A^p, C^p, N^p, \gamma^p, R^p, F) \le T_{\max}, \tag{7}$$

where f represents each of the W frames, which has a delay constraint. Clearly, (6) does add difficulties on top of (4), although a number of constraints are included to eliminate some valid solutions for the original problem.

Note that the unique feature of (4) is that it is essentially a convex function, which has been shown in large amount of research done on rate-distortion relationship under the context of multimedia processing and transmissions. In other words, there always exists a global optimality of this formulation. This is a very important conclusion, since other existing global cross-layer optimization frameworks focusing on network QoS or using decomposition approach cannot guarantee the convexity of all decomposed subproblems. For a given multimedia application, the global optimization problem described in (4) turns into a constrained nonlinear optimization problem, which can be solved by *Lagrangian multipliers (LMs)* and *Lagrangian relaxation (LR)* [51]. So, we can use the derived Lagrangian cost function as the unified cost function. In this work, the cost function J is the average distortion over the given video clip $E(D)$.

For the global optimality of system performance, we need to optimize current control action u_t over time $t + 1, t + 2, \ldots, N$; in other words, current control action $u_t = \{v, P, N_R, A, C\}$ needs to be chosen with considerations of future cost J. For example, the end user will evaluate the perceivable video quality based on the overall quality of the whole video clip rather than the quality of each individual

video frame. Therefore, the cost function for optimization over time based on (4) is

$$\arg\min_{u_t \in U} C_i^{u_t}(x_t) + J(x_{t+1}) \mid x_t. \tag{8}$$

Here, x_t is the state at time t, and the value $J(x_{t+1})$ is introduced to capture the future cost (i.e., $E(D)$) at time $t+1$ incurred as a result of taking the control action u_t at time t.

So far, there is only one exact method for global optimization over time with nonlinearities and random disturbances [36], which is *dynamic programming (DP)*. DP provides methods for choosing a value function $J(\cdot)$ to derive an optimal policy $\pi = \{u_0, u_1, u_2, \ldots, u_{N-1}\}$. There has been a plenty of research on how to use DP-based algorithms for multimedia processing and transmission. In order to use DP to find the global optimality of (4), a unified cost-to-go function J has to be constructed:

$$J^{u_t}(x_t) = \min_{u_t,\ldots,u_{N-1}} \sum_{i=t}^{N-1} J(x_i, u_i(x_i), x_{i+1}) | x_t. \tag{9}$$

Then, the global optimization problem turns into calculating the cost-to-go function $J_0(x_0)$, which is the overall cost to be incurred in the finite horizon of N steps.

3.3. Numerical results

We have evaluated the performance of the proposed integrated cross-layer framework through extensive simulations based on H.264 JM12.2 codec. In general, we are interested in comparing our integrated cross-layer design with the best possible results of H.264 codec. Our goal here is to illustrate the difference of performance gain between the global optimality achieved by the proposed framework and the superposition of multiple local optimality done separately at different network layer (s). In this paper, the best baseline performance is derived: (1) at the application layer, it uses the rate control scheme of H.264 codec; (2) at the network layer, it always chooses the path with the best average SNR at each hop; (3) at the MAC and PHY layers, it always chooses the AMC scheme for the shortest delay while keeping the predefined PER performance.

From the simulation results, up to 3 dB PSNR gain can be achieved by using the proposed approach compared with using the existing piecemeal approach, as shown in Figure 2.

Remark 1. We have proposed a top-down theoretical cross-layer framework for multimedia over wireless networks, and the correctness of the proposed methodology is based on its rigorous theoretical foundation. Moreover, the proposed methodology is based on dynamic programming, which means that it is very flexible and scalable; any interaction of interest in the system can be easily integrated into the proposed framework. Since we consider all the major interactions of interest spanning from application layer to physical layer, we have overcome the major drawback of existing cross-layer designs where the simplification occurs at the system modeling phase rather than the problem solving phase. Therefore, the proposed methodology provides the true global optimality and a new design guidance to the cross-layer design for multimedia over various wireless networks.

4. FURTHER DISCUSSION

In this section, we will further discuss how to apply the aforementioned global optimization framework for real-time multimedia communications as formulated in (4). This is not only practically important but also theoretically interesting.

4.1. Problem statement

So far, we have presented a new theoretical framework for cross-layer design of multimedia communications over wireless networks, which provides a sound methodological foundation for us to evaluate cross-layer designs using dynamic programming (DP) which has been widely adopted to study sequential decision-making problems (stochastic control). However, the practical applications of dynamic programming are limited mostly due to the dual curses of dimensionality and uncertainty, that is, the large size of underlying state space of the *cost-to-go function* which is a function of the current state for evaluating the expected future cost to be incurred. The "curse of dimensionality" means that the computational complexity of the cross-layer design can be increased exponentially when the number of considered design variables increases. The "curse of uncertainty" (modeling) indicates the fact that in a complex networking system there exist various uncertainties making it very difficult to know the explicit system model and/or states. Generally speaking, uncertainties can be classified into two categories: measurement uncertainties and model uncertainties. Under the context of cross-layer design, measurement uncertainties are mainly caused by randomness in data collecting process such as inaccurate channel feedback, while model uncertainties are mainly caused by various approximations made in system modeling process such as approximations made on channel quality, traffic load, node mobility, number of users, and user behaviors. For cross-layer design, uncertainties existing in interdependency among design variables may cause severe performance degradation. Therefore, the "dual curses" make cross-layer optimization a very challenging problem.

4.2. Methodology

4.2.1. Feature-based approximate dynamic programming

The most sensible and rational way to deal with the difficulty caused by "dual curses" is to generate a compact parametric representation (compact representation, for brevity) to approximate the cost-to-go function for a significant complexity reduction through mapping the huge state space to a much smaller feature space characterized by a compact representation.

Currently, the selection of a compact representation largely relies on heuristics which somewhat contradicts

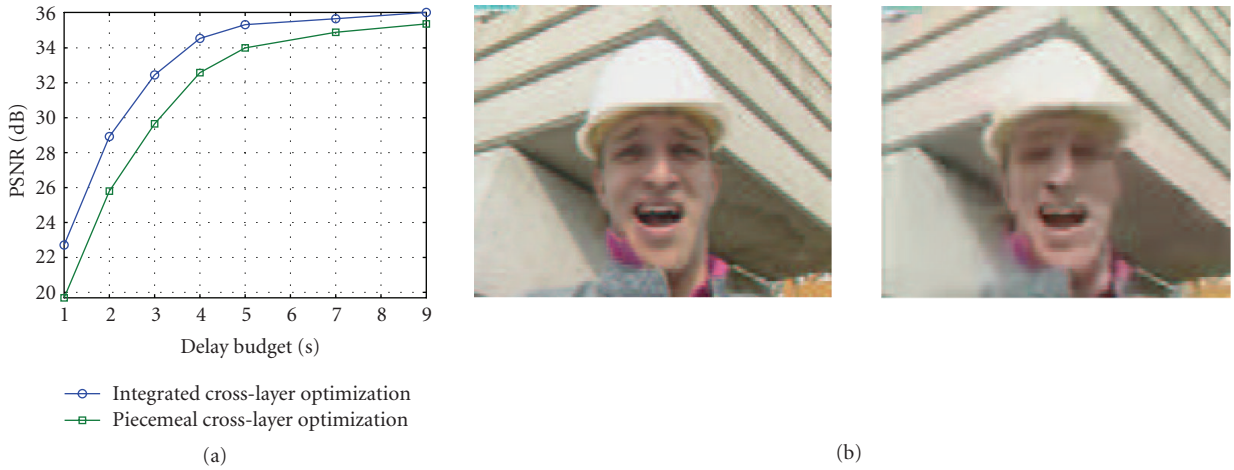


Figure 1: (a) Performance comparison using sample video clip: global cross-layer optimization versus existing piecemeal cross-layer optimization. Here, assume that multihop paths and their link quality can be found by a multihop routing protocol, such as optimized link state routing protocol (OLSR) [52]. In this simulation, the average link SNRs (in dB) of three multihop paths are $P_1 = \{5, 10, 15\}$, $P_2 = \{5, 10, 20\}$, and $P_3 = \{5, 15, 25\}$. Six AMC schemes as listed in [51] are adopted at the PHY layer. At the receiver side, a simple error concealment algorithm is adopted where the lost macroblock will be replaced by the latest correctly received one. (b) Perceptual video quality comparison based on H.264 codec with the same delay budget, where (a) is global optimality and (b) is the best baseline.

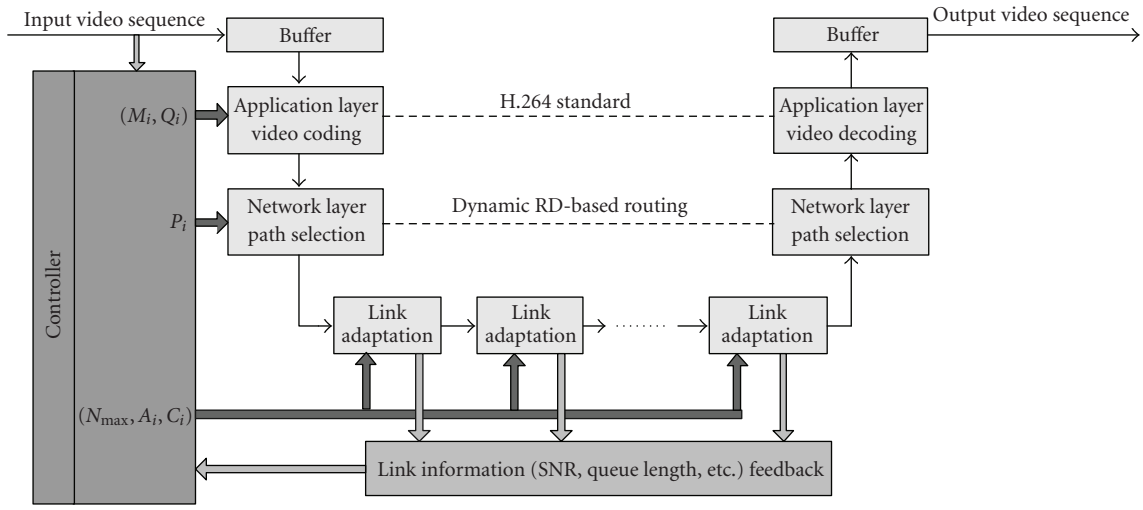


Figure 2: An integrated cross-layer framework of multimedia communications over multihop wireless networks.

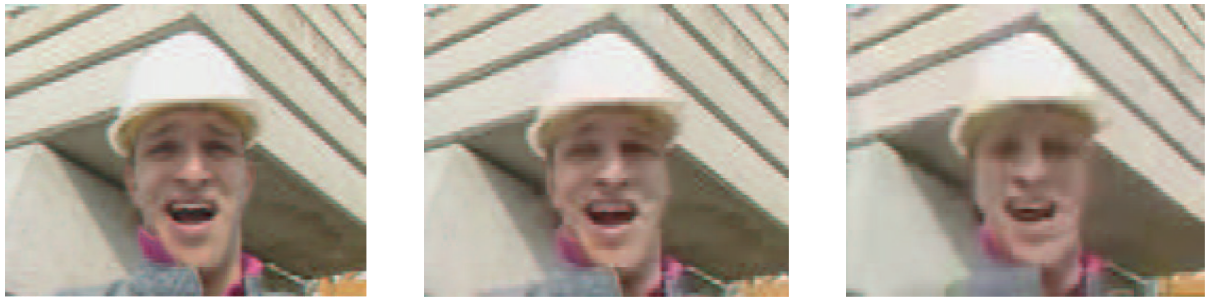


Figure 3: An integrated cross-layer framework of multimedia communications over multihop wireless networks.

Table 1: Perceptual video quality comparison based on H.264 codec in a real-time environment $T_{\text{frame}} = 35\,ms$), where (a) globally optimal, (b) near optimal, and (c) the best baseline.

u[1]	QP	179.23	T	Global	$ADP - P_1$	$ADP - P_2$	$ADP - P_3$	Baseline
u[2]	Path	31.95	0.01312	146433	164824	156336	149483	194813
u[3]	QP,Path	93.31	0.02217	88552	98207	97470	95454	123607
u[4]	AMC	23.18	0.03454	56983	67341	64555	60364	78341
u[5] QP,AMC 252.49	0.03960	49719	54891	51555	50201	63891		
u[6]	Path,AMC	3.22	0.06105	33702	36535	35623	34564	42535
u[7]	QP,Path,AMC	157.31	0.09725	21573	23551	23337	22811	27551

the nonheuristic aspects of the dynamic programming methodology. Therefore, we propose a new method based on nonadditive measure theory, which can dynamically generate compact representations of the huge state space. Unlike other nonlinear feature-extraction approaches such as artificial neural network, the proposed method is adaptive and nonheuristic in the sense that it allows us to quantitatively characterize the significance or the desirability of state vectors with considerations of interactions among different state variables. Therefore, new feature-based approximate dynamic programming can be developed based on the adaptive feature extraction and compact representation.

We consider a large-scale dynamic programming problem defined on a finite state space S. Let n denote the cardinality of S; thus we have $S = 1, \ldots, n$, and $n = \Pi_{k=1}^{\omega} N_k$, where N_k is the number of control actions for the parameter $k \in [1, \omega]$. Our goal is to quantitatively characterize the significance effect of parameters on the cost-to-go function J.

4.2.2. Feature-based compact representation

In the context of dynamic programming, the cost-to-go vector J is defined as a vector whose components are the cost-to-go values of various states. The cost-to-go function specifies the mapping from states to cost-to-go values. Therefore, the optimal cost-to-go vector J^* of policy π with initial state i is defined by

$$J_i^{\pi^*} = \min_{\pi} J_i^{\pi}, \quad i \in S, \tag{10}$$

and the policy at state i is defined by

$$\pi_i^* = \arg\min_{u \in U(i)} \left(C_{iu} + \sum_{j \in S} J_j^* \right), \quad \forall i \in S. \tag{11}$$

The dynamic programming problem is to seek the optimal policy π^* to achieve

$$J_i^{\pi^*} = J_i^*, \quad \forall i \in S. \tag{12}$$

In large-scale dynamic programming problems, the size of state space normally increases exponentially with the number of state variables, making it extremely difficult to compute and store each component of the cost-to-go function. Therefore, the most sensible way is to map a huge state space $\Re^n$ to a much smaller feature space $\Re^m$ ($m \ll n$). Formally, a compact representation can be described as a scheme for recording a high-dimensional cost-to-go vector $V \in \Re^n$ using a lower-dimensional parameter vector $W \in \Re^m$. So, if we can obtain an approximation of $J \in \Re^m$ to $J^* \in \Re^n$, we may still generate a near optimal control policy π_J but with significant computational acceleration satisfying

$$\pi_J = \arg\min_{u \in U(i)} \left(C_{iu} + \sum_{j \in S} J_j \right), \quad \forall i \in S. \tag{13}$$

In the context of approximate dynamic programming, we would like to see that when J approaches J^*, π_J is getting close to π^*. Therefore, a compact representation can be described as a mapping of $J : \Re^n \mapsto \Re^m$ to $W :\in \Re^m$ associated with a cost-to-go vector. Each component of $\widetilde{J}_i(W)$ of the mapping is the i th component of a cost-to-go vector represented by the parameter vector W.

Formally, a feature f is defined as a function from the state space S into a finite set Q of feature values. In stochastic multistage decision processes, we might need several features, $f_1, f_2, \ldots, f_K$, forming a feature vector $F(i) = (f_1(i), \ldots, f_K(i))$ for each state $i \in S$. The feature vector $F(i)$ indicates the desirability or significance of the associated state i. Therefore, for a feature-based compact representation, the component $\widetilde{J}_i(W)$ of $\widetilde{J}(W)$ can be written as $\widetilde{J}_i(F(i), W)$.

For approximate dynamic programming using feature-based compact representation, the approximate cost-to-go function is

$$\widetilde{J}(W) = g(F(i), W), \tag{14}$$

where g is defined as an approximation architecture $g : \Re^K \times \Re^m \mapsto \Re$ with $\Re \in \Re^n$, meaning that g will only cover the most significant finite region of $\Re^n$. In order to achieve the best quality of approximation, it would be highly desirable to have effective and efficient parameter-selection and feature-extraction algorithms. Unfortunately, the existing feature-extraction and parameter-selection algorithms are mainly based on heuristics such as Q-learning and neural network, but those methods lack for sound engineering judgement.

4.2.3. Feature extraction and parameter selection based on significance measure

Feature extraction requires us to catch the "dominant nonlinearities" in the optimal cost-to-go function J^*. Then,

based on the extracted features, the parameter vectors W can be determined and so can be $\Re^m$.

In our preliminary study [53], a new method for feature extraction, called *significance measure*, has been proposed based on nonadditive measure theory [40]. The unique feature of significance measure is that the nonlinear interactions among state variables on the cost-to-go function can be quantitatively measured by solving a generalized nonlinear Choquet integral. As shown in our preliminary study [53], the feature-based approximation can be expressed as

$$\Delta\widetilde{J} = \sum_{k=1}^{m} \Delta f(x) \cdot \mu_k + \xi, \tag{15}$$

where x is state variable, J is the cost-to-go function, and $f(x)$ is observation of state variable. The impact of interactions among state variables on the cost-to-go function is described by a *set function* μ defined on the power set of state variables satisfying the condition of vanishing at the empty set, that is, $\mu : P(X) \to (-\infty, +\infty)$ with $\mu(\varnothing) = 0$. The set function μ is called nonadditive measure [40]. There has been a lot of research done to find the optimal μ by solving the nonlinear integral equation such as Choquet integral [48, 54] based on a set of observation data. An advantage of the proposed significance measure method described above is that it only needs system operation data (simulations), which can be easily acquired from the device drivers. Therefore, it is fairly efficient in terms of computation and storage. Significant measure and sensitivity analysis.

Once, we determine the significance measure of state variables $\mu_1, \mu_2, \ldots, \mu_{2^\omega - 1}$ corresponding to different parameter sets. Then, the parameter set with the largest μ_i can be directly used for parameter selection. Furthermore, the value of each parameter set can be interpreted as feature, since it reflects the parameter significance towards the cost-to-go function. We can choose the parameter set having the largest value of μ_i to be the compact representation $W \in \Re^m$ of the high-dimensional cost-to-go vector $V \in \Re^n$. Therefore, various approximate dynamic programming approaches using feature-based compact representation can utilize the new method for compact representation, feature extraction, and parameter selection. For example, if we adopt feature-based look-up table approximate dynamic programming architecture, the approximated cost-to-go function is $\widetilde{J}_i(W) = W$, or we can use $\widetilde{J}_i(W) = W^T F(i)$ if using linear approximate architecture.

4.3. Numerical results

As discussed earlier, based on the significance measure and sensitivity analysis, we can derive a new method for feature extraction and compact representation for approximating the original large-scale dynamic programming. Using the same problem setting as of Figure 1, a simple example to illustrate the basic idea of the proposed approach is devised. First, an operational data set in the format of *[QP, Path, AMC, Value of cost-to-go function]* has been collected by uniformly sampling the dynamic programming state space. Then, the significance measure algorithm, as presented in [53, 55] was applied to the collected data. The derived significance measure of control variables and their interdependencies can be derived as shown in Tabl @ IV-B3, where columns 1–3 represent significance measure of control variables, where u (column 1) indicates the significant impact of each subset of control variables (QP, $Path$, AMC) (column 2) on the cost-to-go function (column 3) based on the collected measurements. The original three-dimension (QP, AMC, $Path$) DP problem can be approximated by a two-dimension (QP, AMC) ADP problem. Columns 4-9 represent MSE distortion of DP versus ADP versus the best baseline under different frame delay budgets (T), where three ADP values are corresponding to adopt different fixed paths (P_1, P_2, or P_3) in the approximation.

In this simulation, based on the significance measure, the interaction between QP and AMC has the most significant impact on the cost-to-go function, meaning that "path" is not as significant as the other variables. So, it could be excluded from the optimal search. This way, the cardinality of the approximated state space can be *reduced by three times*. Compared with the global optimal performance, the maximum approximation error caused by excluding path from the DP search is 12.5%, corresponding to the shortest delay budget; however, in this case, the result of ADP-based solution still outperforms the best baseline H.264 performance by 15.4%.

Remark 2. In this section, we propose a new method for feature extraction and compact representation of approximate dynamic programming, which is based on the significance measure of each set of design variables. We discuss a novel feature-based approximate dynamic programming approach for solving the large-scale dynamic programming problem in support of real-time multimedia applications. Furthermore, since all the significant measures of a power set of design variables are available, a scalable complexity framework by exploring the tradeoff between the quality of approximation (QoA) and the quality of service (QoS) could be developed in future. Note that the proposed significance measure method and the feature-based approximate dynamic programming approach are fairly generic and are applicable for any large-scale design optimization and real-time control scenarios.

5. CONCLUSION

The major challenges of current cross-layer design for multimedia communications over wireless networks are (1) lacking of understanding of cross-layer behaviors, (2) simplifying cross-layer design at the system modeling phase, and (3) relying on heuristic approaches. We argue that all these challenges are caused by lacking of a new methodology for cross-layer design of multimedia communications over wireless networks. This has motivated us to propose a new methodological foundation for cross-layer design of multimedia communications over wireless networks, which has made two major contributions to the research area: (1) the theoretical framework with major design variables spanning from application layer to physical layer for cross-layer design of multimedia communications over wireless

networks, and (2) the novel feature-based approximate dynamic programming approach based on a new significance measure method to understand cross-layer behaviors and speed up large-scale cross-layer optimization. The proposed methodological foundation is fairly general and can be applicable to other applications in multimedia communications. However, we are *not* trying to solve all the problems in this paper; rather, we are trying to look into this challenging problem from a different angle and open up a new research direction for future studies in the field of wireless multimedia communications. We believe that the proposed methodological foundation will significantly contribute to the emerging research areas such as service- and application-oriented QoS provisioning in the future Internet.

REFERENCES

[1] T. Turletti and C. Huitema, "Videoconferencing on the internet," *IEEE/ACM Transactions on Networking*, vol. 4, no. 3, pp. 340–351, 1996.

[2] Q.-F. Zhu and L. Kerofsky, "Joint source coding, transport processing, and error concealment for H.323-based packet video," in *Visual Communications and Image Processing (VCIP '99)*, vol. 3653 of *Proceedings of SPIE*, pp. 52–62, San Jose, Calif, USA, January 1999.

[3] G. Côté and F. Kossentini, "Optimal intra coding of blocks for robust video communication over the Internet," *Signal Processing: Image Communication*, vol. 15, no. 1-2, pp. 25–34, 1999.

[4] P. Haskell and D. Messerschmitt, "Resynchronization of motion compensated video affected by ATM cell loss," in *Proceedings of IEEE International Conference on Speech, Acoustics and Signal Processing (ICASSP '92)*, vol. 3, pp. 545–548, San Francisco, Calif, USA, March 1992.

[5] J. Y. Liao and J. D. Villasenor, "Adaptive intra update for video coding over noisy channels," in *Proceedings of IEEE International Conference on Image Processing (ICIP '96)*, vol. 3, pp. 763–766, Lausanne, Switzerland, September 1996.

[6] E. Steinbach, N. Färber, and B. Girod, "Standard compatible extension of H.263 for robust video transmission in mobile environments," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 6, pp. 872–881, 1997.

[7] A. Ortega and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 23–50, 1998.

[8] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74–90, 1998.

[9] R. Zhang, S. L. Regunathan, and K. Rose, "Video coding with optimal inter/intra-mode switching for packet loss resilience," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 966–976, 2000.

[10] D. Wu, Y. T. Hou, B. Li, W. Zhu, Y.-Q. Zhang, and H. J. Chao, "An end-to-end approach for optimal mode selection in internet video communication: theory and application," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 977–995, 2000.

[11] S. Mao, Y. T. Hou, X. Cheng, H. D. Sherali, S. F. Midkiff, and Y.-Q. Zhang, "On routing for multiple description video over wireless ad hoc networks," *IEEE Transactions on Multimedia*, vol. 8, no. 5, pp. 1063–1074, 2006.

[12] A. C. Begen, Y. Altunbasak, O. Ergun, and M. H. Ammar, "Multi-path selection for multiple description video streaming over overlay networks," *Signal Processing: Image Communication*, vol. 20, no. 1, pp. 39–60, 2005.

[13] Y. Andreopoulos, N. Mastronarde, and M. van der Schaar, "Cross-layer optimized video streaming over wireless multihop mesh networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 11, pp. 2104–2115, 2006.

[14] J. Villalón, P. Cuenca, L. Orozco-Barbosa, Y. Seok, and T. Turletti, "Cross-layer architecture for adaptive video multicast streaming over multirate wireless LANs," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 699–711, 2007.

[15] J. Wang, M. Venkatachalam, and Y. Fang, "System architecture and cross-layer optimization of video broadcast over WiMAX," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 712–721, 2007.

[16] Z. Li, G. M. Schuster, A. K. Katsaggelos, and B. Gandhi, "Rate-distortion optimal video summary generation," *IEEE Transactions on Image Processing*, vol. 14, no. 10, pp. 1550–1560, 2005.

[17] P. V. Pahalawatta, Z. Li, F. Zhai, and A. K. Katsaggelos, "Rate-distortion optimized video summary generation and transmission over packet lossy networks," in *Image and Video Communications and Processing (IVCP '05)*, vol. 5685 of *Proceedings of SPIE*, pp. 801–809, San Jose, Calif, USA, January 2005.

[18] M. van der Schaar, D. S. Turaga, and R. Wong, "Classification-based system for cross-layer optimized wireless video transmission," *IEEE Transactions on Multimedia*, vol. 8, no. 5, pp. 1082–1095, 2006.

[19] A. Luthra, G. J. Sullivan, and T. Wiegand, "Introduction to the special issue on the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 557–559, 2003.

[20] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

[21] Q. Xu, V. Stanković, and Z. Xiong, "Distributed joint source-channel coding of video using raptor codes," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 851–861, 2007.

[22] S. Choudhury and J. D. Gibson, "Payload length and rate adaptation for multimedia communications in wireless LANs," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 796–807, 2007.

[23] N. Mastronarde, D. S. Turaga, and M. van der Schaar, "Collaborative resource exchanges for peer-to-peer video streaming over wireless mesh networks," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 1, pp. 108–118, 2007.

[24] J. Chen and Y. Yang, "Multi-hop delay performance in wireless mesh networks," in *Proceedings of IEEE International Conference on Communications (ICC '07)*, Glasgow, Scotland, June 2007.

[25] C. Wang, B. Li, K. Sohraby, M. Daneshmand, and Y. Hu, "Upstream congestion control in wireless sensor networks through cross-layer optimization," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 786–795, 2007.

[26] D. Wu and R. Negi, "Effective capacity: a wireless link model for support of quality of service," *IEEE Transactions on Wireless Communications*, vol. 2, no. 4, pp. 630–643, 2003.

[27] S. Kompella, S. Mao, Y. T. Hou, and H. D. Sherali, "Cross-layer optimized multipath routing for video communications in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 831–840, 2007.

[28] B. Liang and M. Dong, "Packet prioritization in multihop latency aware scheduling for delay constrained communication," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 819–830, 2007.

[29] Q. Liu, S. Zhou, and G. B. Giannakis, "Queuing with adaptive modulation and coding over wireless links: cross-layer analysis and design," *IEEE Transactions on Wireless Communications*, vol. 4, no. 3, pp. 1142–1153, 2005.

[30] P. Pahalawatta, R. Berry, T. Pappas, and A. Katsaggelos, "Content-aware resource allocation and packet scheduling for video transmission over wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 749–759, 2007.

[31] H.-P. Shiang and M. van der Schaar, "Multi-user video streaming over multi-hop wireless networks: a distributed, cross-layer approach based on priority queuing," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 770–785, 2007.

[32] S. Mohapatra, N. Dutt, A. Nicolau, and N. Venkatasubramanian, "DYNAMO: a cross-layer framework for end-to-end QoS and energy optimization in mobile handheld devices," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 722–737, 2007.

[33] D. Rajan, "Towards universal power efficient scheduling in Gaussian channels," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 808–818, 2007.

[34] Y.-J. Chang, F.-T. Chien, and C.-C. J. Kuo, "Cross-layer QoS analysis of opportunistic OFDM-TDMA and OFDMA networks," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 657–666, 2007.

[35] Q. Liu, S. Zhou, and G. B. Giannakis, "Cross-layer combining of adaptive modulation and coding with truncated ARQ over wireless links," *IEEE Transactions on Wireless Communications*, vol. 3, no. 5, pp. 1746–1755, 2004.

[36] P. Werbose, "ADP: goals, oppotunities and principles," in *Handbook of Learning and Approaximate Dynamic Programming*, J. Si, A. G. Barto, W. B. Powell, and D. Wunsch, Eds., Wiley-IEEE Press, New York, NY, USA, 2004.

[37] R. Bellman, *Dynamic Programming*, Princeton University Press, Princeton, NJ, USA, 1957.

[38] D. Bertsekas and J. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, Belmont, Mass, USA, 1996.

[39] J. Si, A. G. Barto, W. B. Powell, and D. Wunsch, Eds., *Handbook of Learning and Approaximate Dynamic Programming*, Wiley-IEEE Press, New York, NY, USA, 2004.

[40] D. Denneberg, *Non-Additive Measure and Integral*, Kluwer Academic, Dordrecht, The Netherlands, 1994.

[41] D. Denneberg and M. Grabisch, "Interaction transform of set functions over a finite set," *Information Sciences*, vol. 121, no. 1-2, pp. 149–170, 1999.

[42] M. Grabisch, "Fuzzy measures and integrals: a survey of applications and recent issues," in *Fuzzy Sets Methods in Information Engineering: A Guided Tour of Applications*, D. Dubois, H. Prade, and R. Yager, Eds., John Wiley & Sons, New York, NY, USA, 1996.

[43] R. R. Yager, "Uncertainty representation using fuzzy measures," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 32, no. 1, pp. 13–20, 2002.

[44] G. Choquet, "Theory of capacities," *Annales de l'Institut Fourier*, vol. 5, pp. 131–295, 1953.

[45] J.-R. Chang, C.-T. Hung, and G.-H. Tzeng, "Non-additive grey relational model: case study on evaluation of flexible pavement," in *Proceedings of IEEE International Conference on Fuzzy Systems*, vol. 1, pp. 577–582, Budapest, Hungary, July 2004.

[46] A. Denguir-Rekik, G. Mauris, and J. Montmain, "Propagation of uncertainty by the possibility theory in Choquet integral-based decision making: application to an E-commerce website choice support," *IEEE Transactions on Instrumentation and Measurement*, vol. 55, no. 3, pp. 721–728, 2006.

[47] M.-H. Ha, Z.-F. Feng, E.-L. Du, and Y.-C. Bai, "Further discussion on quasi-probability," in *Proceedings of the International Conference on Machine Learning and Cybernetics (ICMLC '06)*, pp. 3508–3513, Dalian, China, August 2006.

[48] Z. Wang, "A new model of nonlinear multiregressions by projection pursuit based on generalized Choquet integrals," in *Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ '02)*, vol. 2, pp. 1240–1244, Honolulu, Hawaii, USA, May 2002.

[49] M. Grabisch, "k-order additive discrete fuzzy measures and their representation," *Fuzzy Sets and Systems*, vol. 92, no. 2, pp. 167–189, 1997.

[50] M. Grabisch, "Modelling data by the Choquet integral," in *Information Fusion in Data Mining*, V. Torra, Ed., pp. 135–148, Physica, Heidelberg, Germany, 2003.

[51] D. Wu, S. Ci, and H. Wang, "Cross-layer optimization for video summary transmission over wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 841–850, 2007.

[52] IETF, "RFC 3626—optimized link state routing protocol (OLSR)," http://www.faqs.org/rfcs/rfc3626.html.

[53] S. Ci and H.-F. Guo, "Quantitative dynamic interdependency measure and significance analysis for cross-layer design under uncertainty," in *Proceedings of the 16th International Conference on Computer Communications and Networks (ICCCN '07)*, pp. 900–904, Honolulu, Hawaii, USA, August 2007.

[54] M. Grabisch, "A graphical interpretation of the Choquet integral," *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 5, pp. 627–631, 2000.

[55] S. Ci and H.-F Guo, "Significance measure with nonlinear and incommensurable observations," in *Proceedings of IEEE Global Communications Conference (GLOBECOM '07)*, Washington, DC, USA, November 2007.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2008, Article ID 739192, 8 pages
doi:10.1155/2008/739192

Research Article

Optimal Multilayer Adaptation of SVC Video over Heterogeneous Environments

Truong Cong Thang,[1,2] Jung Won Kang,[2] Jeong-Ju Yoo,[2] and Yong Man Ro[1]

[1] *Multimedia Group, Information and Communications University, Daejeon 305-732, South Korea*
[2] *Broadcasting Media Research Group, Electronics and Telecommunications Research Institute, Daejeon 305-700, South Korea*

Correspondence should be addressed to Truong Cong Thang, tcthang@etri.re.kr

Received 21 August 2007; Accepted 22 November 2007

Recommended by Jianwei Huang

Scalable video coding (SVC) is a new video coding format which provides scalability in three-dimensional (spatio-temporal-SNR) space. In this paper, we focus on the adaptation in SNR dimension. Usually, an SVC bitstream may contain multiple spatial layers, and each spatial layer may be enhanced by several FGS layers. To meet a bitrate constraint, the fine-grained scalability (FGS) data of different spatial layers can be truncated in various manners. However, the contributions of FGS layers to the overall/collective video quality are different. In this work, we propose an optimized framework to control the SNR scalability across multiple spatial layers. Our proposed framework has the flexibility in allocating the resource (i.e., bitrate) among spatial layers, where the overall quality is defined as a function of all spatial layers' qualities and can be modified on the fly.

1. INTRODUCTION

In the context of Universal Multimedia Access (UMA), multimedia contents should be adapted to meet various constraints of heterogeneous environments [1]. Among existing media types, video content imposes many challenges to the development of a transparent delivery chain [2]. Currently, there are two main technologies for video adaptation, namely, transcoding and scalable coding. Due to the high complexity of transcoding, many efforts have been focused on the development of scalable coding [3, 4].

Scalable video coding (SVC) [5] is a promising video format for applications of multimedia communication. SVC format, which is extended from the latest advanced video coding (AVC) [6], is appropriate to create a wide variety of bitrates with high-compression efficiency. An original SVC bitstream can be easily truncated in different manners to meet various characteristics and variations of devices and connections. The scalability is possible in 3 dimensions: spatial, temporal, and SNR. The spatial scalability of SVC intelligently combines multiple spatial layers into a single bitstream, which has much better coding efficiency than simulcasting multiple streams of different spatial sizes. The temporal scalability is supported by hierarchical B pictures which enable both the ease of truncation and high-coding efficiency. Besides, fine-grained scalability (FGS) data of SNR scalability can be truncated arbitrarily to meet the bitrate constraint of connection. Usually, FGS data is truncated in a top-down manner [7], that is, starting from the highest spatial layer to the lowest spatial layer.

Though scalable coding formats in general and SVC in particular provide flexibility in truncating the coded bitstream, there is a strong demand for the optimal adaptation strategies and solutions in various contexts [8]. In recent years, much research has been focused on the adaptation of MPEG-4 FGS video (e.g., [9, 10]), where the bitstream contains only one spatial layer. In our previous works [11, 12], we have developed an MPEG-21-enabled adaptation system, where the SVC bitstream is adapted in the full spatio-temporal-SNR space. However, the goal is still to optimize the quality of only one resolution.

In this work, we focus on FGS data truncation of multispatial layer (or multilayer for short) SVC bitstream, so as to maximize the overall/collective quality of the spatial layers provided by the adapted bitstream. For example, let us consider the following scenario (Figure 1). Suppose that

a surveillance video is encoded by SVC format with two spatial layers, each of which is enhanced by FGS data. That video is streamed to a remote building where two users will consume the content. The first user has a PC which will decode the highest spatial layer and the second user has a PDA which decodes the lowest spatial layer. To meet the connection bitrate of that building, the FGS data will be truncated. Note that the FGS data may account for a significant portion (e.g., two thirds) of the total bitrate.

Currently, the FGS data of the above bitstream can be truncated with a few approaches. With the conventional approach of top-down truncation [7], the lowest spatial layer always gets the best possible quality while the highest spatial layer may be much degraded. On the contrary, with the approach of [13], some FGS data in the lower spatial layer can be removed so as the highest spatial layer always has the best possible quality. We call this approach as highest-max, implying the maximization of the highest spatial layer's quality. It should be noted that the highest-max truncation is not "bottom-up" truncation, in which truncation simply starts from the lowest spatial layer to the highest spatial layer. As discussed later, the bottom-up truncation is actually not useful.

Additionally, in practice the requirements from users may be complex and variant in time. For example, the above two users request a "weighted balance" of qualities between them (or between the two spatial layers); or when a key (primary) user moves between end-devices, the quality should be reallocated accordingly. We consider this fact as a kind of user collaboration [14], which should be exploited to improve the overall/collective quality across multiple users.

In this paper, we propose a general framework to adapt SVC bitstream having multiple spatial layers. Our proposed framework has the flexibility in allocating the resource (i.e., bitrate) among spatial layers, where the overall quality is defined as a function of all spatial layers' qualities and can be modified on the fly. The adaptation process is first formulated as a constrained optimization problem. Then we propose a solution based on the Viterbi algorithm to find the optimal bitrate allocation between spatial layers. We will also show that the approaches of [7, 13] are just two extreme cases of our general framework.

This paper is organized as follows. In Section 2, we present the problem formulation. The solution to this problem, which is based on Viterbi algorithm, is proposed in Section 3. Section 4 presents the experiments to show the effectiveness and performance of our framework. Finally, conclusion is provided in Section 5.

2. PROBLEM FORMULATION

The FGS truncation process in SVC can be conceptually illustrated in Figure 2. Suppose that we have an SVC bitstream which consists of 2 spatial layers. Each spatial layer is composed of a base quality layer and FGS data which progressively enhance the SNR quality of that spatial layer. FGS data of a lower spatial layer can be used for interlayer prediction of a higher spatial layer. However, the FGS data can be truncated arbitrarily, regardless of the location. Anyway, the FGS data of a given spatial layer should be truncated "top-down", that is, from the highest quality to the base quality.

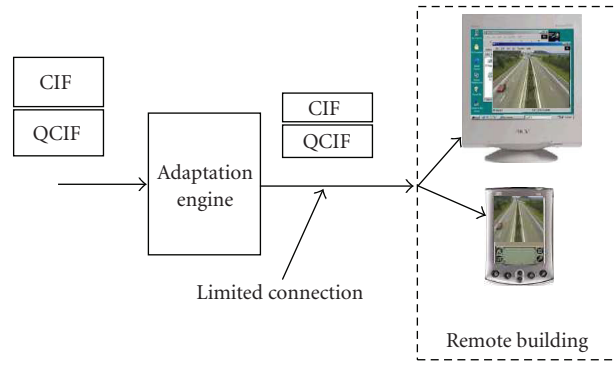


Figure 1: A scenario of two users with one SVC bitstream.

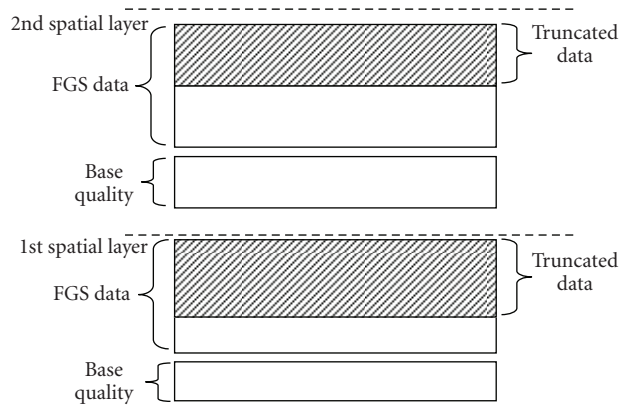


Figure 2: FGS data truncation of an SVC bitstream with multiple spatial layers.

Note that, the base quality layer represents the minimum quality of a spatial layer. Nonetheless, in practice, users could request quality thresholds of their own, which may be higher than those of base quality layers.

Denote OQ as the "overall quality" (or collective quality) of the truncated bitstream, N the number of spatial layers, R_i and Q_i the "FGS bitrate" and corresponding quality of spatial layer i, and $Q_i^{\min}$ the requested minimum quality of spatial layer i. Also let R^c denote the bitrate constraint of all FGS data, which is the difference of the overall bitrate constraint and the base quality bitrate. The adaptation framework can be formulated as follows:

maximize OQ subject to

$$\sum_{i=1}^{N} R_i \le R^c, \quad Q_i \ge Q_i^{\min} \quad \text{with } i = 1, \ldots, N. \tag{1}$$

OQ is generally defined as a function of spatial layers' qualities:

$$OQ = f(Q_1, Q_2, \ldots, Q_N). \tag{2}$$

Currently, we compute the overall quality using the weighted sum as follows:

$$OQ = \sum_{i=1}^{N} w_i \cdot Q_i, \quad (3)$$

where w_i is the weight of layer i, $0 \le w_i \le 1$.

With (3), the quality harmonization between different spatial layers can be adjusted by changing the values of w_i's. For example, given the scenario described in Section 1, if $w_1 = 1$ and $w_2 = 0$, the truncation will be top-down so as the first spatial layer always has the best possible quality.

It should be noted that, due to interlayer prediction in SVC, the quality of a higher spatial layer depends on the qualities, or more exactly on the bitrates, of lower spatial layers. That is,

$$Q_i = g_i(Q_{i-1}). \quad (4)$$

So truncating all FGS data of lower spatial layers to "make place" for FGS data of the highest spatial layer may not always give the best possible quality for the highest spatial layer. This will be discussed in more detail in the experiments.

As this framework is essentially a resource allocation problem, it can be extended to cover temporal scalability as long as we employ a quality metric that support multidimensional adaptation (e.g., [15]). In the following section, we will present a method based on Viterbi algorithm to solve optimization problem (1).

3. SOLUTION BY THE VITERBI ALGORITHM

Although the FGS data can be truncated finely, the truncation in practice is done in discrete steps (e.g., with a unit of 1 Kbps). So the bitrates R_i's in the above problem formulation can take discretized values with some step size. Further, as described above, the dependency between spatial layers should be considered in optimization problem (1). So this problem can be solved optimally by the Viterbi algorithm of dynamic programming [16–18]. In the following, we call a *selection* as a discretized truncation operation at a given spatial layer.

The principle of the Viterbi algorithm lies in building a trellis to represent all viable allocations at each instant, given all the predefined constraints. The basic terms used in the algorithm are defined as follows (Figure 3).

(i) *Trellis*: A trellis is made of all surviving paths that link the initial node to the nodes in the final stage.

(ii) *Stage*: Each stage corresponds to a spatial layer to be truncated.

(iii) *Node*: In our problem, each node is represented by a pair (i, a_i), where i is the stage number, and a_i is the accumulated bitrate of all FGS data until this stage.

(iv) *Branch*: Given selection k_i at stage i which has the bitrate R_{ik_i}, a node $(i-1, a_{i-1})$ in the previous stage $(i-1)$ will be linked by a branch of value $Q_i(k_i, a_{i-1})$ to node (i, a_i) with

$$a_i = a_{i-1} + R_{ik_i}, \quad (5)$$

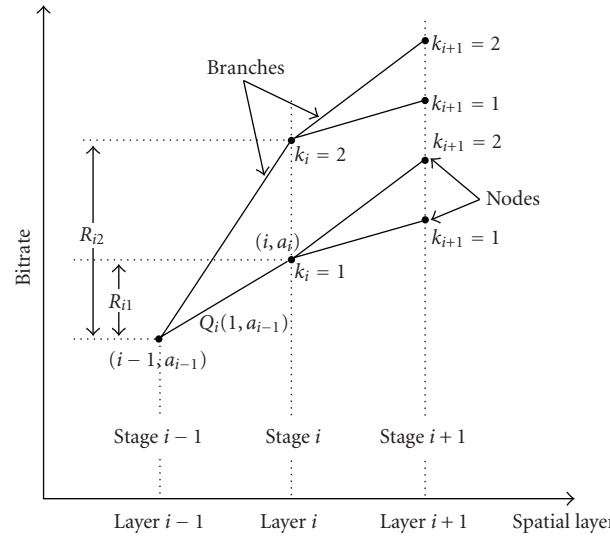


Figure 3: Trellis diagram grown by the Viterbi algorithm. Each stage corresponds to a spatial layer, and each branch corresponds to a *selection* for a given spatial layer.

satisfying

$$a_i \le R^c. \quad (6)$$

(v) *Path*: A path is a concatenation of branches. A path from the first stage to the final stage corresponds to a set of possible selections for all spatial layers.

In SVC, the higher spatial layers are dependent on the lower spatial layers (but not vice versa). So when the trellis is growing, the stages are arranged in the increasing order of spatial layers (i.e., from the lowest spatial layer to the highest spatial layer). Note that, the first stage (stage 0) is just an initial point, which does not correspond to any spatial layers. Similarly, the quality $Q_i(k_i, a_{i-1})$ depends on not only selection k_i of layer i but also the selections corresponding to previous nodes in the path. Moreover, thanks to the pruning described below, each node (i, a_i) will correspond to only one selection k_i. So we can rewrite $Q_i(k_i, a_{i-1}) = Q_i(k_i, k_{i-1}, \ldots, k_1)$.

From the above, we can see that the optimal path, corresponding to the optimal set of selections, is the one having the highest weighted sum $\sum_{i=1}^{N} w_i \cdot Q_i$. We now apply the Viterbi algorithm to generate the trellis and to find the optimal path as shown in Algorithm 1 [17, 18].

Let K_i denote the number of selections for spatial layer i. With the above algorithm, from the initial node (0, 0), there will be at most K_1 branches growing to K_1 nodes of stage 1. The number of branches will be K_1 if all values of a_1 are not greater than R^c. Similarly, there will be at most K_2 branches grown from each node of stage 1. Due to this growing, there may be more than one branch reaching to the same accumulated bitrate (or arriving to the same node). However, thanks to step 2, there remains only one branch (i.e., the best one) that arrives to a node.

We see that the complexity of this solution depends on the number of layers and the number of selections which is determined by the truncation step size. Officially, the number

Step 0: $i = 0$. Start from the initial node $(0, 0)$.

Step 1: At each stage i, add possible branches to the end nodes of the surviving paths. At each node, a branch is grown for each of the available selections; the branch must satisfy condition (6).

Step 2: Among all paths arriving at a node in stage $i + 1$, the one having the highest accumulated sum of $\sum_{t=1}^{i+1} w_t \cdot Q_t$ is kept, and the rest are pruned.

Step 3: $i = i+1$. If $i \leq N$, go back to step 1, otherwise go to Step 4.

Step 4: At the final stage, compare all surviving paths then select the path having the highest value of $\sum_{i=1}^{N} w_i \cdot Q_i$. That path corresponds to the optimal set of selections for all spatial layers.

Algorithm 1

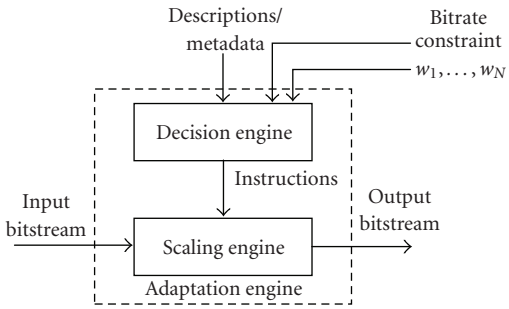


Figure 4: Architecture of an SVC adaptation system.

of spatial layers in SVC can be up to 8. However, to maintain a good coding efficiency, an SVC bitstream contains at most three spatial layers (with different resolutions) [7]. As shown later in the next section, with practical conditions, the optimal solution based on the Viterbi algorithm can be found in real time.

It should be noted that the solution provided by the above algorithm is optimal for the "discretized" problem. However, as mentioned earlier, the practical truncation is often based on a specific step size. From our experience, a truncation equal to 1% of the total FGS bitrate would not result in any perceptual difference. So, practitioners would look for a solution of the discretized problem, rather than the *continuous-valued* problem.

Currently, the R-D information (i.e., R_i, Q_i) in our framework is operational. Although the operational R-D data is not easy to obtain in real time, they can be computed in advance and used as metadata to adapt the bitstream on the fly as in previous work of video coding [16, 19]. Moreover, some analytical models can be used to represent the R-D information in a compact manner [9, 19].

4. EXPERIMENTS

In this section, some experiments are presented to show the flexibility and usefulness of our proposed framework. We developed an SVC adaptation engine which consists of a decision engine and a scaling engine (Figure 4). The decision engine employs metadata about the operational R-D information of input bitstream, and other metadata including bitrate constraint, the weights w_i's of spatial layers, and then provides as output the adaptation instructions. The instructions here are the amount of FGS bitrate which should be truncated in each spatial layer. The scaling engine takes the instructions and adapts the input bitstream accordingly.

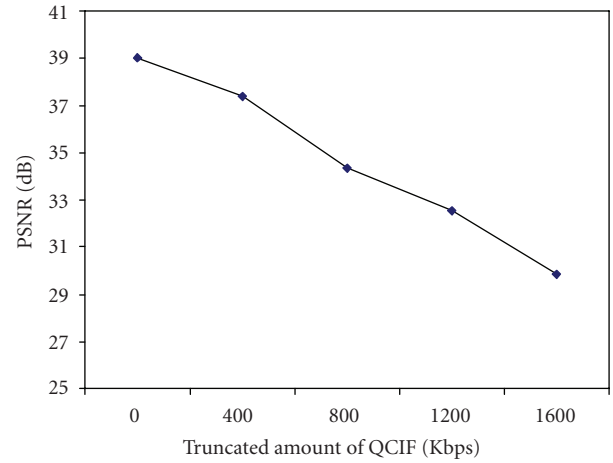


Figure 5: R-D information of QCIF layer. The FGS truncation is applied to QCIF layer only.

4.1. Allocation results

Test videos are encoded by the recent software JSVM7.12. The results presented below are for the football video, encoded with 2 spatial layers, QCIF and CIF both having frame rate of 30 fps and GOP size of 16. Correspondingly, two users will consume this content as in the scenario of Section 1. The base quality QP values of both spatial layers are 38. QCIF spatial layer is enhanced by 3 FGS layers and CIF spatial layer by 2 FGS layers. The FGS bitrates of CIF and QCIF layers are, respectively, 1924 (Kbps) and 1877 (Kbps). We assume that users have no special requests on the quality threshold (i.e., $Q_i^{\min}$). Quality metric used in optimization problem (1) is PSNR value averaged over all video frames. The overall quality is given by

$$OQ = w_1 \cdot Q_1 + w_2 \cdot Q_2. \tag{7}$$

For ease of presentation and discussion, the step size for FGS truncation is set to be 400 (Kbps) and the quality is shown according to the amount of truncated bitrate. Each spatial layer will be truncated at four points, namely, 400, 800, 1200, and 1600. Figures 5 and 6 show the operational R-D information of QCIF layer and CIF layer according to the amount of truncated data.

Now suppose that $w_1 = 0.33$ and $w_2 = 0.67$. These weight values would give some balance between the two spatial layers as the PSNR value of QCIF layer is often higher than that of CIF layer. The objective of truncation will be to optimize the overall quality $OQ = 0.33 \cdot Q_1 + 0.67 \cdot Q_2$. The optimal selections are represented by the solid path (denoted by

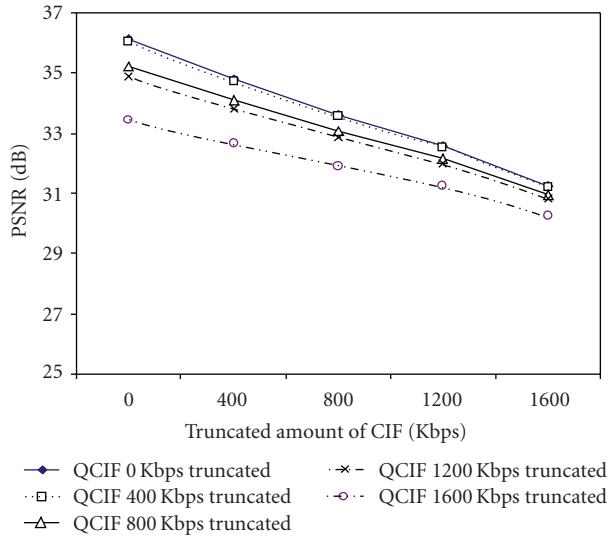


FIGURE 6: R-D information of CIF layer. The FGS truncation is applied to both QCIF and CIF layers.

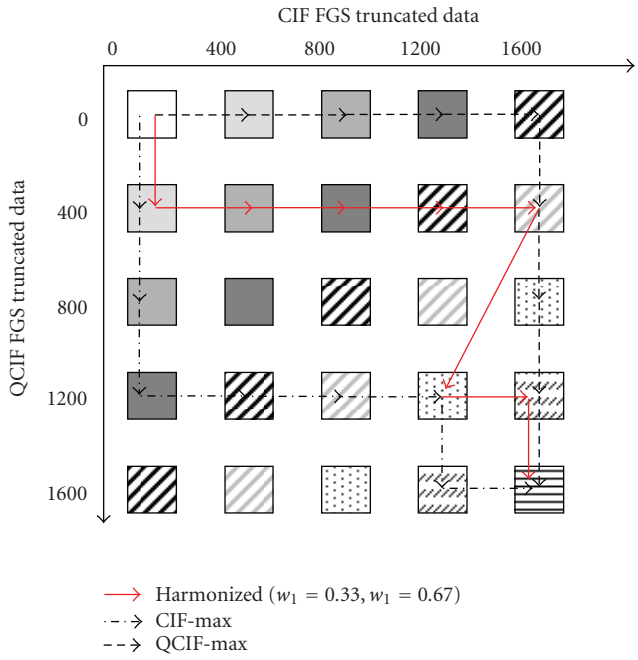


FIGURE 7: Illustration of different FGS truncation methods. Here FGS data in CIF and QCIF layers are truncated flexibly.

harmonized path) in Figure 7. We can see that when the total truncated amount is increased (from 0 Kbps to 3200 Kbps, with step size of 400 Kbps), the selections of multilayer truncation correspond to the boxes (400, 0), (400, 400), (400, 800), (400, 12000), (400, 1600), (1200, 1200), (1200, 1600), (1600, 1600), where (a, b) indicates that truncated amounts of QCIF and CIF layers are, respectively, a Kbps and b Kbps. Note that, in Figure 7, the boxes of the same pattern and gray level have the same total amount of truncated data (in both CIF and QCIF layers).

If $w_1 = 1$ and $w_2 = 0$, this implies a top-down truncation used always to maximize QCIF layer's quality. Obviously, the selections in this case are represented by the dashed path (denoted as *QCIF-max* path), where FGS data of CIF layer are truncated first.

If $w_1 = 0$ and $w_2 = 1$, this implies a truncation that aims to maximize CIF layer's quality. The selections in this case are represented by the dashed-doted path (denoted as *CIF-max* path). As shown by this path, FGS data of QCIF layer are first truncated until the amount of 1200 (Kbps), then FGS data of CIF layer are truncated. Here, the selections of (1600, 400) and (1600, 800) are not used because a truncated amount of 1600 (Kbps) in QCIF layer would result in a significant degradation in CIF layer due to interlayer prediction. So, FGS data of QCIF layer will not be completely truncated before truncating CIF FGS data. That is, a bottom-up truncation would not be a good choice for most practical conditions.

Figure 8 shows the advantage of the harmonized truncation in detail. The weight values are as above, $w_1 = 0.33$ and $w_2 = 0.67$. In these figures, the horizontal axis represents the total amount of truncated FGS data (in both CIF and QCIF layers), and the vertical axis represents the PSNR values of each spatial layer (QCIF in Figure 8(a) and CIF in Figure 8(b)). We can see that, with CIF-max truncation, the quality of the CIF layer is always maximized (Figure 8(b)), but the quality of QCIF layer decreases very quickly (Figure 8(a)). With QCIF-max truncation, the phenomenon is inversed. Meanwhile, the curve of harmonized truncation shows an intermediate solution between these two extreme cases. For example, when the total amount of truncated data is 1600 Kbps, the quality of QCIF layer is 37.4 dB, that is, 4.9 dB higher than that of CIF-max truncation; and the quality of CIF layer is 32.54 dB, that is, 1.3 dB higher than that of QCIF-max truncation.

Now let $w_1 = 0.15$ and $w_2 = 0.85$, which implies an emphasis on the CIF layer. The solution provided by the above algorithm corresponds to the path of (400, 0), (400, 400), (1200, 0), (1200, 400), (1200, 800), (1200, 1200), (1200, 1600), and (1600, 1600). Figure 9 shows the corresponding quality comparison. We can see that the harmonized curve now gets close to the CIF-max curve. However, at some points, the gain in QCIF layer is still several dBs compared to QCIF-max method (Figure 9(a)). So, by adjusting the weight values, we can flexibly control the tradeoff between the two layers. We found that the shapes of curves having finer steps are very similar to those of the current curves. This means that the current curves (with step size of 400 kbps) represent sufficiently the adaptation behavior.

When the weight values are equal ($w_1 = 0.5$ and $w_2 = 0.5$), the harmonized truncation of this given bitstream turns out to be the same as QCIF-max truncation. This is due to the fact that the PSNR value of QCIF layer is often higher than that of CIF layer (as mentioned above), so the QCIF layer is always "emphasized" in truncation process. This means that the *intuitive* nonweighted sum of PSNR values of CIF and QCIF layers would not give any tradeoff for the two layers.

Figures 10 and 11 show the optimality of the harmonized path compared to the CIF-max and QCIF-max paths for two case, ($w_1 = 0.33$, $w_2 = 0.67$) and ($w_1 = 0.15$, $w_2 = 0.85$). The

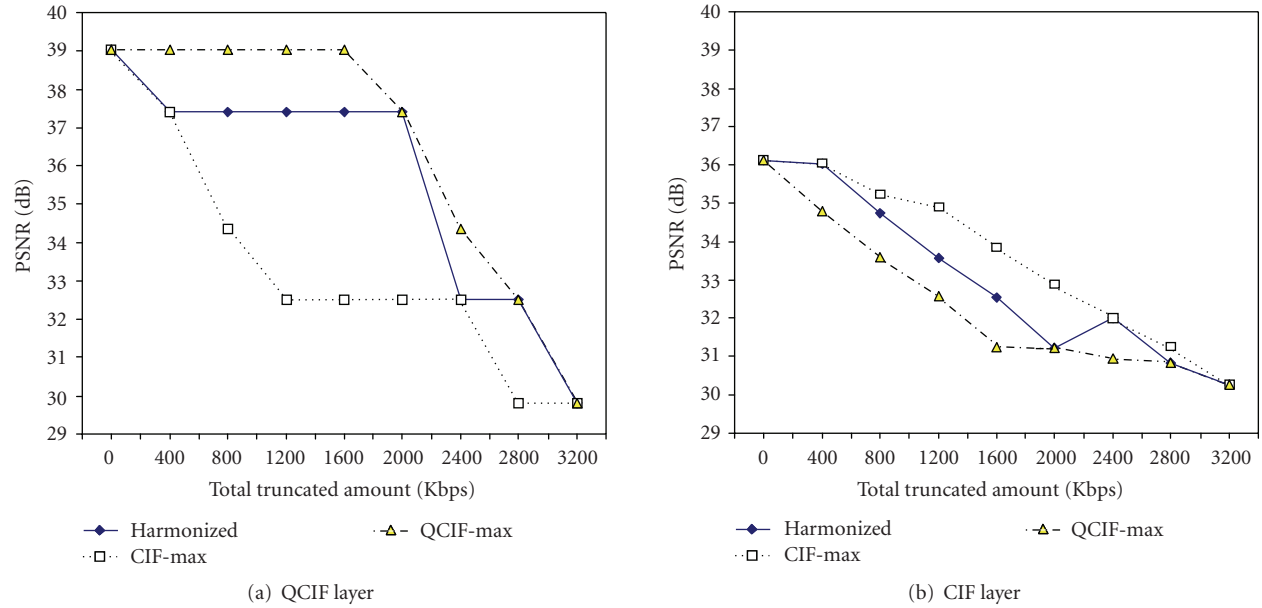


Figure 8: Comparison of three truncation methods: harmonized (with $w_1 = 0.33$, $w_2 = 0.67$), CIF-max, and QCIF-max.

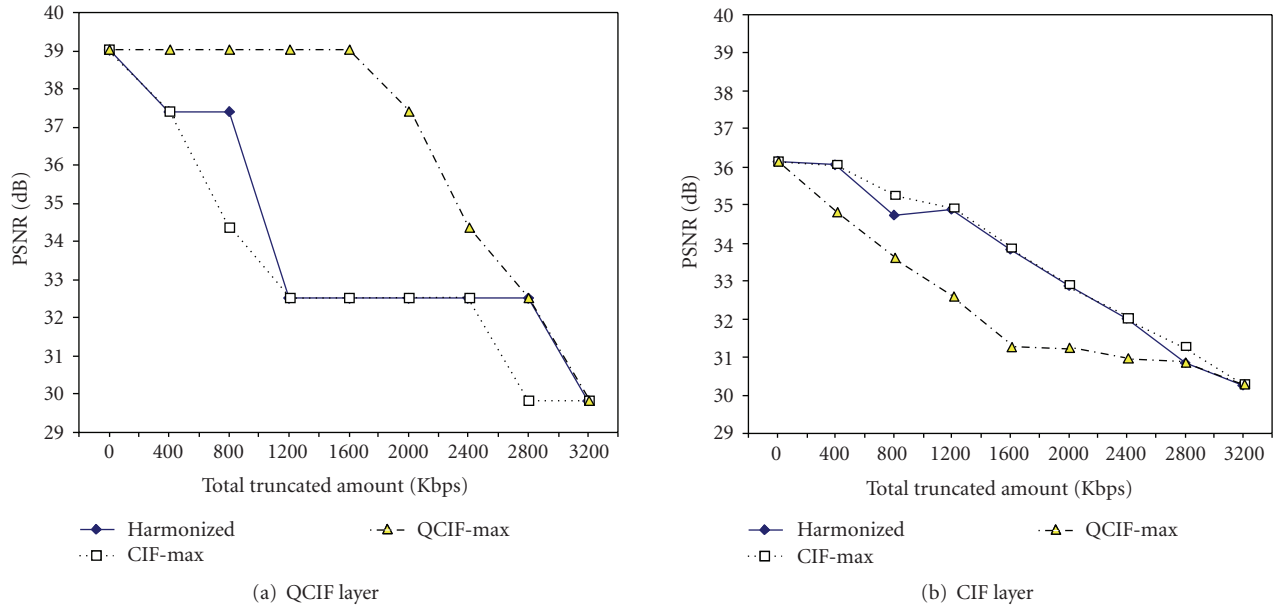


Figure 9: Comparison of three truncation methods: harmonized (with $w_1 = 0.15$, $w_2 = 0.85$), CIF-max, and QCIF-max.

horizontal axis represents the total amount of truncated FGS data, and the vertical axis represents the overall quality computed by (7). We can see that the overall quality of the harmonized path is always higher than or equal to those of the other two paths. This means that the truncations based on CIF-max and QCIF-max paths cannot provide the optimal results.

It should be noted that the PSNR value in Figures 10 and 11 just represents the collective quality, which is used to guarantee the optimal tradeoff between layers. In order to see the advantage of our proposed method in improving users' quality, one should also consider the R-D curves of specific spatial layers (i.e., Figures 8 and 9). For example, though the gaps between the curves of Figure 10 are sometimes small, the actual improvement for specific users may be up to several dBs as seen in Figures 8(a) and 8(b). We have found similar observations with other sequences. In fact, as long as there exists a gap between the two extreme truncations, a tradeoff between them can always be achieved.

4.2. Algorithm complexity

To check the complexity of the algorithm, we measure the processing time of the algorithm with different step sizes,

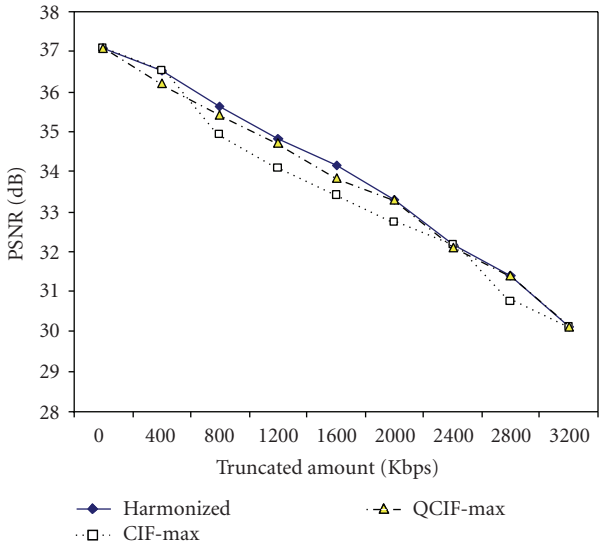


Figure 10: Overall quality of different truncation solutions (w_1 = 0.33 and w_2 = 0.67).

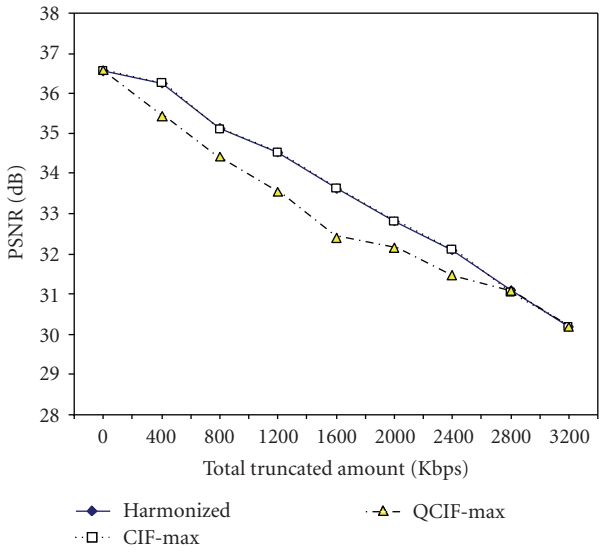


Figure 11: Overall quality of different truncation solutions (w_1 = 0.15 and w_2 = 0.85).

namely, 1 Kbps, 2 Kbps, 5 Kbps, and 10 Kbps. The quality values of new truncation selections are linearly interpolated from the previous sample points obtained with the step size of 400 Kbps (which is similar to [20]). The complexity is represented by processing time which is measured by the number of system clock ticks (1000 ticks per second). The proposed algorithm is run on a notebook having Pentium M 1.86 GHz processor and 1 G RAM. Figure 12 shows the processing time with respect to the total amount of truncated bitrate. We can see that when the step size is 1 Kbps, the processing time can be up to 80 milliseconds; however, with the other step sizes, the processing time is just around 20 milliseconds. Especially, when step size is 10 Kbps, the complexity become so small that the processing time is mostly zero (more exactly, less than 1 tick).

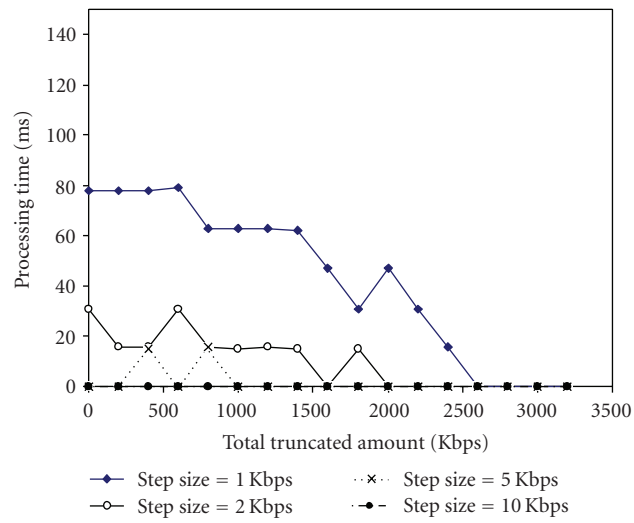


Figure 12: Processing time with different step sizes (2-layer bitstream).

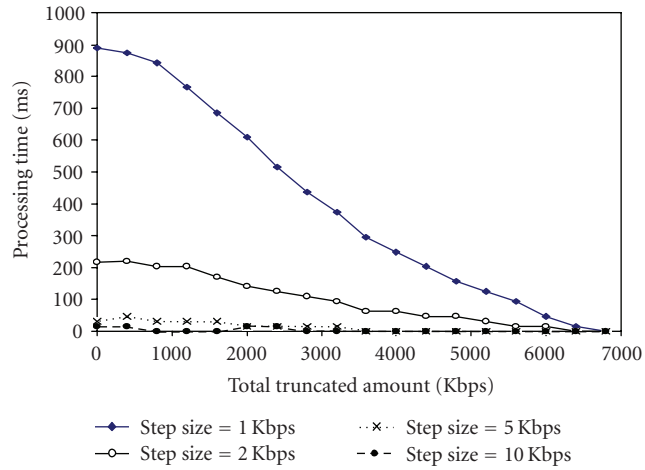


Figure 13: Processing time with different step sizes (3-layer bitstream).

As the number of spatial layers of an SVC bitstream is at most 3 in practice [7], we add to the bitstream one more spatial layer (4CIF), of which the amount of FGS data is 3500 Kbps. The algorithm is run again with step sizes of 1 Kbps, 2 Kbps, 5 Kbps, 10 Kbps and the corresponding results are shown in Figure 13. Now we see that the processing time with step size of 1 Kbps increases significantly which is up to 900 milliseconds. However, when step size is 10 Kbps, still the processing time is usually less than 1 millisecond, sometimes reaching to 15 milliseconds. Note that, with this bitstream, even the step size of 10 Kbps is less than 0.2% of the total FGS bitrate.

Meanwhile, it should be noted that in practical video communication, the acceptable processing delay can be up to 400 milliseconds for two-way application and 10 seconds for one-way application [21].

Obviously, with a bitstream of higher bitrate, the step size should be increased proportionally. Whereas, from the above

example we can see that even if the step size is just 0.5% or 1% of the total bitrate, the processing time of the Viterbi algorithm would become negligible. Moreover, from our previous experience with subjective tests on video quality [22], with quality scale of just 9 or 10 levels, it is still very difficult for end-users to differentiate the adjacent quality levels. This means that the step size may not need to be as small as 1% of the total bitrate. The exact step size which results in the just noticeable difference (JND) in user perception is an interesting issue in our future work.

From the above, we can see that when there is any change in user requests or in bitrate constraint, the optimization problem can be recomputed on the fly and the adaptation will be seamless to the users. This means that our proposed framework can provide the truncation flexibility with optimal result for any conditions of bitrate constraint and quality tradeoff between layers.

5. CONCLUSIONS

In this paper, we proposed a general framework to adapt SVC bitstream through FGS truncation across multiple spatial layers. Our proposed framework has the flexibility in allocating the resource (i.e., bitrate) among spatial layers, where the overall quality is defined as a function of all spatial layers' qualities and can be modified on the fly. The adaptation process of the proposed framework was formulated as a constrained optimization problem and then optimally solved by the Viterbi algorithm. Through experiments, we also showed that the current approaches of FGS truncation were special cases of our general framework. For future work, we will consider some perceptual quality metrics in our adaptation system and employ analytical models for R-D representation. Also, the framework will be extended to cover other constraints of heterogeneous environments, such as terminal capability and packet loss.

ACKNOWLEDGMENTS

The authors would like to thank Dong Su Lee of ICU for his help in this work. This work was supported by the IT R&D program of MIC/IITA [2005-S-103-03, Development of Ubiquitous Content Access Technology for Convergence of Broadcasting and Communications] and by 2nd Phase of Brain Korea 21 project sponsored by Ministry of Education and Human Resources Development (Seoul, South Korea).

REFERENCES

[1] A. Vetro, "MPEG-21 digital item adaptation: enabling universal multimedia access," *IEEE Multimedia*, vol. 11, no. 1, pp. 84–87, 2004.

[2] S.-F. Chang and A. Vetro, "Video adaptation: concepts, technologies, and open issues," *Proceedings of the IEEE*, vol. 93, no. 1, pp. 148–158, 2005.

[3] A. Vetro, "Transcoding, scalable coding and standardized metadata," in *Proceeding of the 8th International Workshop on Visual Content Processing and Representation (VLBV '03)*, vol. 2849, pp. 15–16, Madrid, Spain, September 2003.

[4] A. Vetro, C. Christopoulos, and H. Sun, "Video transcoding architectures and techniques: an overview," *IEEE Signal Processing Magazine*, vol. 20, no. 2, pp. 18–29, 2003.

[5] H. Schwarz, D. Marpe, and T. Wiegand, "SNR-scalable extension of H.264/AVC," in *Proceedings of IEEE International Conference on Image Processing (ICIP '04)*, vol. 5, pp. 3113–3116, Singapore, October 2004.

[6] ITU-T ISO/IEC JTC1, "Video coding for generic audiovisual services," ITU-T Recommendation H.264 ISO/IEC 14496-10 AVC, 2003.

[7] Joint Scalable Video Model (JSVM)9.0. ITU-T VCEQ JVT-V202, Marrakech, January 2007.

[8] D. Mukherjee, E. Delfosse, J.-G. Kim, and Y. Wang, "Optimal adaptation decision-taking for terminal and network quality-of-service," *IEEE Transactions on Multimedia*, vol. 7, no. 3, pp. 454–462, 2005.

[9] C. Hsu and M. Hefeeda, "Rate-distortion models for FGS-encoded video sequences," in *Proceeding of the 17th International Conference on Computer Theory and Applications (ICCTA '06)*, pp. 334–337, Alexandria, Egypt, September 2006.

[10] T. Kim and M. H. Ammar, "Optimal quality adaptation for MPEG-4 fine-grained scalable video," in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '03)*, vol. 1, pp. 641–651, San Francisco, Calif, USA, March-April 2003.

[11] J. W. Kang, S.-H. Jung, J.-G. Kim, and J.-W. Hong, "Development of QoS-aware ubiquitous content access testbed," in *Proceedings of the International Conference on Consumer Electronics (ICCE '07)*, pp. 1–2, Las Vegas, Nev, USA, January 2007.

[12] T. C. Thang, Y. S. Kim, Y. M. Ro, J. W. Kang, and J.-G. Kim, "SVC bistream adaptation in MPEG-21 multimedia framework," in *Proceeding of the International Packet Video Workshop (PV '06)*, Hangzhou, China, April 2006.

[13] M. Mathew, K. Lee, and W.-J. Han, "Multi layer quality layers," ITU-T VCEQ JVT-S043, Geneva, April 2006.

[14] Z. Li, J. Huang, and A. K. Katsaggelos, "Pricing based collaborative multi-user video streaming over power constrained wireless down link," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '06)*, vol. 5, Toulouse, France, May 2006.

[15] M. H. Pinson and S. Wolf, "A new standardized method for objectively measuring video quality," *IEEE Transactions on Broadcasting*, vol. 50, no. 3, pp. 312–322, 2004.

[16] A. Ortega, K. Ramchandran, and M. Vetterli, "Optimal trellis-based buffered compression and fast approximations," *IEEE Transactions on Image Processing*, vol. 3, no. 1, pp. 26–39, 1994.

[17] G. D. Forney, "The Viterbi algorithm," *Proceedings of IEEE*, vol. 61, pp. 268–278, 1973.

[18] T. C. Thang, Y. J. Jung, and Y. M. Ro, "Effective adaptation of multimedia documents with modality conversion," *Signal Processing: Image Communication*, vol. 20, no. 5, pp. 413–434, 2005.

[19] X. M. Zhang, A. Vetro, Y. Q. Shi, and H. Sun, "Constant quality constrained rate allocation for FGS-coded video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 2, pp. 121–130, 2003.

[20] L.-J. Lin and A. Ortega, "Bit-rate control using piecewise approximated rate-distortion characteristics," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 4, pp. 446–459, 1998.

[21] ITU-T, "End-user multimedia QoS categories," Recommendation G.1010, 2001.

[22] T. C. Thang, Y. J. Jung, and Y. M. Ro, "Modality conversion for QoS management in universal multimedia access," *IEE Proceedings: Vision, Image and Signal Processing*, vol. 152, no. 3, pp. 374–384, 2005.