

Markov Decision Model for Adaptive Scheduling of Stored Scalable Videos

Chao Chen, *Student Member, IEEE*, Robert W. Heath Jr., *Fellow, IEEE*,
Alan C. Bovik, *Fellow, IEEE*, and Gustavo de Veciana, *Fellow, IEEE*

Abstract—We propose two scheduling algorithms that seek to optimize the quality of scalably coded videos that have been stored at a video server before transmission. The first scheduling algorithm is derived from a Markov Decision Process (MDP) formulation developed here. We model the dynamics of the channel as a Markov chain and reduce the problem of dynamic video scheduling to a tractable Markov decision problem over a finite state space. Based on the MDP formulation, a near-optimal scheduling policy is computed that minimize the mean square error. Using insights taken from the development of the optimal MDP-based scheduling policy, the second proposed scheduling algorithm is an online scheduling method that only requires easily measurable knowledge of the channel dynamics, and is thus viable in practice. Simulation results show that the performance of both scheduling algorithms is close to a performance upper bound also derived in this paper.

Index Terms—Videos transport, Scheduling algorithm, Wireless communication.

I. INTRODUCTION

THE variation of wireless channel capacity and tight delay constraints make the delivery of video difficult. Although adaptive transmission strategies such as adaptive video data scheduling can be employed, deriving the optimal adaptive transmission policy is difficult because the transmission strategies taken at different time are coupled with each other via receiver buffer state. Furthermore, due to the nature of predictive video coding algorithms, a video frame can be decoded only when its predictors have been received. Hence, the prediction structure of the video codec enforces a partial order on the transmissions of the video packets, which limits the flexibility of adaptive video transmission.

Scalable video coding (SVC) is one approach to enable flexible video transmission over channels with varying throughput [1], [2]. An SVC video encoder produces a layered video stream that contains a base layer and several enhancement layers. If the throughput is low, the transmitter can choose to transmit the base layer only, which provides a moderate, but acceptable, degree of visual quality at the receiver. If the channel conditions improve, the transmitter can transmit one, or more, enhancement layers to further improve the visual quality. Conceptually, SVC provides a means to adapt the data

rate for wireless video transmission. The wireless transmitter can adapt the data rate by selectively scheduling video data associated with various layers for transmission rather than transcoding the video sequence into a different rate.

Designing scalable video scheduling algorithms for wireless channels is a complex task. The scheduling policy depends, not only on the channel conditions, but also, on the receiver buffer state. For example, if the receiver has successfully buffered base layer data over many frames, the scheduler could choose to transmit some enhancement layer data to improve the video quality even if the throughput is low. At any time, the scheduling decision will determine the receiver buffer state which, in turn, affects the future scheduling decisions. Therefore, adaptive video data scheduling is a sequential decision problem. The most natural way to address such problems is to model the dynamics of the channel as a finite state Markov chain and to employ a Markov decision process (MDP)-based formulation to study scheduling methods. For stored video transmission, however, directly determining an optimal scheduling policy using an MDP formulation is not possible, because the system state space is infinitely large (see Section. III-A). Moreover, in a practical wireless network, a model for the dynamics of the channel states is not typically available, which limits the applicability of this approach.

A. Contributions

The objective of this paper is to leverage the MDP framework to develop practical scheduling algorithms and optimize the receiver video quality for stored scalable video transmission over wireless channels. First, we propose a tractable MDP formulation based on a reasonable approximation of the state space. Near optimal scheduling policies can be derived from this MDP formulation. Second, we propose a scheduling algorithm that substantially simplifies the MDP-based scheduling policy as it requires only limited information regarding the channel state dynamics. Third, we prove an upper bound on the achievable video quality of all possible scheduling algorithms. Finally, we provide simulation results that show, under different channel conditions, the performance of the proposed scheduling algorithms is indeed very close to the upper bound.

Our contributions are summarized as follows:

- 1) *An MDP formulation is proposed to facilitate the design of adaptive scheduling policies for stored video transmission.* In this paper, we focus on stored video transport, where video sequences have been encoded and

stored on a video server before transmission. This is quite different from real-time video transmission where video frames are generated in real-time. The video scheduler can select any data from the video sequence and send the data to the receiver buffer. Thus, the number of possible receiver buffer states can be effectively regarded as infinite. Because the performance of the scheduling policy depends on the receiver buffer state, the policy needs to be optimized over an infinitely large state space and the scheduling problem is intractable. In this paper, by applying reasonable restrictions on the set of scheduling policies considered in our MDP formulation, we prove that optimizing the transmission policy is equivalent to solving a semi-Markov decision problem on a finite state set (see Section. III). Based on this result, near-optimal scheduling policies can be derived using the proposed MDP formulation.

- 2) *A near-optimal and on-line scheduling algorithm is proposed.* In most cases, models for channel dynamics are not available. By simplifying the channel model and the scheduling decision of the MDP formulation, we devise an on-line scheduling algorithm which, unlike the MDP-based policy, only requires limited measurable knowledge of the channel dynamics. Simulation results show that the proposed on-line algorithm performs nearly as well as the MDP-based scheduling policy.
- 3) *Performance optimality is justified.* To assess the performance of the proposed scheduling algorithms, an upper bound on the achievable video quality for adaptive scheduling is proved. Simulation results show that both the MDP-based scheduling policy and the proposed on-line scheduling policy perform close to the upper bound.

B. Related Work

Adaptive video data scheduling is an important topic of research [3]–[12]. In [3], adaptive video transmission over a packet erasure channel was studied by modeling the buffer state as a controlled Markov chain. In [4], an average-rate constrained MDP formulation was proposed to optimize the quality of error concealed videos at the receiver. For time-varying wireless channels, the amount of data that can be scheduled during a time slot is limited by the channel capacity at the slot. Only considering the constraint of the average transmission rate is insufficient. In [5], an MDP-based scheduling algorithm was proposed for video transmission over packet loss networks. This work was further extended for wireless video streaming in [6], where the wireless channel was modeled as a binary symmetric channel. This channel model can only be justified for fast fading channels, where the coherence time is much less than the delay constraint. In that case, interleaving can be applied without violating the delay constraint, and the channel will appear as an i.i.d channel. For slow fading channels such as those considered here, the bit error rate cannot be modeled as a constant. In [7], adaptive scheduling of scalable videos was studied using an MDP model. The reward of each frame slot was defined as a utility function of the buffer state and the transmission

rate. A foresighted scheduling policy was derived to maximize the long-term reward over all frame slots. Comparing with a scheduling method that myopically maximizes the reward of each individual frame slot, the proposed scheduling algorithm improved the video quality significantly. In [8], [9], [10] and [11], reinforcement learning frameworks were proposed for wireless video transmission. Their proposed algorithms were based on MDP using a discounted-reward maximization formulation. The transmitter learns the characteristics of the channel and the video sequence during the transmission process. The scheduling policy is updated according to the learned characteristics. In our previous work [12], an infinite-horizon average-reward maximization MDP formulation was proposed. The channel characteristics, unlike in this paper, were assumed to be known.

The most closely related prior work is [6]–[10] and [11] which focus on scalable video transmission over wireless channels. Our work contrasts with these as follows:

- *An Infinite-State Space Problem for Stored Video Streaming.* For real-time video transmission, the number of video frames that are ready for transmission is finite because later frames have not yet been generated at the video source. Therefore, the scheduler only needs to select data from a finite set of frames [6]–[10]. In this paper, we focus on stored video streaming, where all the video frames have been encoded before transmission. In this case, the scheduler is allowed to select data from any video frame and the number of possible receiver buffer states is therefore infinitely large. In this paper, we construct a finite-state MDP model and show that the optimal policy derived from this MDP model is also optimal for the original infinite-state problem.
- *Channel Model.* We focus on slow-fading wireless channels experienced by pedestrian users. In the channel model of [6], the bit error probability of the channel was assumed constant. This assumption can only be justified for fast fading channels, where the channel coherence time is much less than the delay constraint in video applications. In that case, interleaving can be applied without violating the delay constraint, and the channel will appear to have i.i.d. bit errors. For slow fading channels, where the coherence time is much longer, it is impossible to apply interleaving over many coherence periods due to the delay constraint. In this case, i.i.d. models are no longer suitable because they do not capture information regarding channel variations. In contrast, the algorithm proposed in this paper explicitly considers channel state variation in scheduling.
- *Optimization objective.* Most of the existing MDP-based scheduling algorithms are based on a *utility* function as the optimization objective [7]–[10]. The utility function is usually written as a weighted sum of the transmission bit rate and the amount of buffered data. The weights assigned to each component of the summation, to some extent, reflect their importance, but are heuristically chosen. The resulting utility function cannot accurately indicate the quality of played out frames. Here, instead

TABLE I
FREQUENTLY USED NOTATION.

F^{intra}	Number of frames in an intra period.
F^{GOP}	Number of frames in a GOP.
L	Number of MGS layers.
z_t	The amount of received data for the frame played out at t .
ω_ℓ^k	The amount of data in the ℓ^{th} layer of a type- k frame.
d_ℓ	The distortion when the ℓ^{th} layer is correctly received.
$d^k(z_t)$	The rate-distortion model for type- k frames.
$\hat{d}^k(z_t)$	The concave envelopes of $d^k(z_t)$.
X_t	The transmission bit rate at t .
Y_t	The packet error rate at t .
R_t	The channel throughput at t .
r^{avg}	The average channel throughput.
C_t	The channel state at t .
V_t	The buffer state at t .
S_t	The system state at t .

of optimizing a utility function, we directly optimize the quality of the video frame played out in each frame slot.

- *Non-availability of channel state dynamics.* In a practical wireless video transmission application, models for the dynamics of the channel state are typically unavailable. To address this problem, a reinforcement learning algorithm can be employed to learn a good policy from making wrong scheduling actions [8]–[10]. Video quality, however, will be degraded during the learning period, which can be as long as tens of seconds. We propose an adaptive alternative to such reinforcement learning methods, that only uses the channel coherence time and current channel throughput which are easy to measure in practice. The performance of the proposed algorithm is very close to a derived performance upper bound.

C. Organization of Paper

This paper is organized as follows. The system model is introduced in Section II. The assumptions we make about the video codec and the rate-distortion model are described in Section II. In Section III, the MDP formulation and the performance upper bound are proposed. A near-optimal on-line scheduling algorithm is introduced and validated by simulations in Section IV. Section V concludes the paper.

II. SYSTEM MODEL

In this section, we describe the wireless video system to be considered. Then, we present our video codec configuration and introduce the rate-distortion model.

We briefly introduce some notation used in the paper. \mathbf{A} and \mathbf{a} are examples of a matrix and a vector, respectively. \mathcal{A} is a set. $|\mathcal{A}|$ is the cardinality of set \mathcal{A} . $\lceil \cdot \rceil$ is the ceiling function. $\mathbb{P}(\cdot)$ is the probability measure and $\mathbb{E}[\cdot]$ is the expectation. $\mathbb{N} = \{0, 1, 2, \dots\}$ is the set of non-negative integers. Other frequently used notation is summarized in Table I.

A. System Overview

We consider a time-slotted system that transmits scalable videos over a slow fading wireless channel. The video sequence is encoded with a quality-scalable video encoder and is stored in a video server. The video server transmits video data to a mobile user via a wireless transmitter. The duration of each frame ΔT is called a frame slot. In each frame slot, the server sends some video data upon request of a scheduler at the wireless transmitter. This data is packetized at the wireless transmitter for physical layer transmission. The channel and receiver buffer state is sent to the scheduler via a feedback channel with negligible delay. The scheduler operates according to a policy that maps the channel and receiver buffer state to the scheduling action (see Fig. 1).

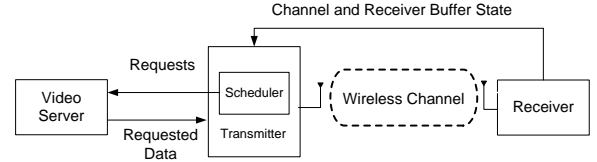


Fig. 1. Dynamic scheduling system for wireless video transmission.

In wireless communication systems such as 3GPP, using the technique of limited feedback, channel state information measured at the receiver can be fed back to the transmitter via a control channel [13] [14] [15]. The delay of the feedback channel is typically much smaller than a frame slot. For example, the feedback delay in 3GPP is 6ms [15], which is much shorter than the 33ms frame slot of 30fps videos. Similarly, the video packets received in each slot can also be acknowledged via a control channel with negligible delay. Therefore, similar to most of existing MDP formulations such as [7]–[10], we assume the feedback is instantaneous. For the case where feedback delay is longer than a frame slot, please refer to [16].

We assume that the link between the video server and the wireless transmitter is not the bottleneck for transmission to the mobile. Thus, from the perspective of the wireless transmitter, the whole video sequence is available for transmission. We also assume that the physical layer channel state information is available at the transmitter and that the modulation and coding scheme (MCS) is determined by a given physical layer link-adaptation policy.

B. Video Codec Configuration

We assume that the video sequence is encoded by an H.264/SVC video encoder. The video frames are uniformly partitioned into intra periods. Every intra period has F^{intra} frames and is further partitioned uniformly into group of pictures (GOP s). Each GOP has F^{GOP} frames. They are encoded using the “Hierarchical B” prediction structure [1], in which video frames are hierarchically organized into T temporal layers as shown in Fig. 2. The last frame in each GOP is called a *key picture*. These key pictures form the 0th temporal layer. There are two types of key pictures: *I* frames

and P frames. The first picture in an intra period is called an I frame, which is encoded without referring other frames. The other key pictures are P frames. Each P frame is encoded using a preceding key picture as reference. All the frames in higher temporal layers are B frames. A B frame in the τ^{th} temporal layer is encoded using the preceding frame and the succeeding frame in the lower temporal layers as reference. In the following, we call a frame in the τ^{th} temporal layer a B^τ frame, where $\tau \geq 1$ (see Fig. 2).

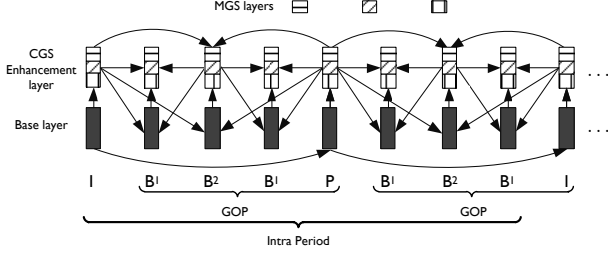


Fig. 2. An illustration of the encoder prediction structure considered in this paper. The prediction order is indicated by arrows. The length of intra period is $F^{\text{intra}} = 8$ and the GOP length is $F^{\text{GOP}} = 4$. The CGS enhancement layer is partitioned into 3 MGS layers.

Every frame is encoded into a base layer and a coarse grain scalability (CGS) layer. The base layer of an I frame is encoded independently. The base layer of a P frame is predictively encoded using the base layer of the preceding key picture. The CGS layer of all key picture are predictively encoded using their respective base layers. For a B frame, its base layer is encoded using the CGS layers of its reference frames. Its CGS layer is encoded using both its base layer and the CGS layers of its reference frames (see Fig. 2).

The CGS layer of each frame is further partitioned into L MGS layers. Each MGS layer contains a portion of the CGS layer data. Thus, the more MGS layers are received, the higher decoding quality can be achieved. In the following, we call the base layer and the MGS layers *quality layers*. We focus on adaptive scheduling of the quality layers in a video stream. The temporal scalability is not exploited.

In this paper, we only consider one CGS enhancement layer. In H.264/SVC, multiple CGS layer is supported and each CGS layer can be partitioned into several MGS layers. The switch between CGS layers, however, is only possible at instantaneous decoder refresh (IDR) frames, which are separated from each other by several intra periods. Therefore, CGS cannot support frame-by-frame rate adaptation. Since the coherence time of wireless channels is much shorter than an intra period, flexible rate adaptation can only be achieved by MGS, which allows to vary the number of quality layer for each frame. Here, we consider frame-by-frame adaptive scheduling of the MGS layers within a single CGS enhancement layer. For the video streams that contain multiple CGS enhancement layers, our scheduling algorithm can be applied to conduct adaptation in one of the CGS enhancement layers while treating all lower layers as the base layer. In the following, we call the MGS layers *enhancement layers*.

Each frame has a playout deadline at the receiver. In the following, frames whose deadlines have expired are called

expired frames, otherwise they are said to be active frames. The first active frame is called the “current frame”. The GOP that contains the current frame is called the “current GOP”. The intra period that contains the current frame is called the “current intra period” (see Fig. 3). The frames in the current GOP are decoded together when the first frame of the GOP is displayed. At any point in time, frames are indexed relative to the current frame as shown in Fig. 3. Each data unit is also tagged with a layer index ℓ . The index for base layer is $\ell = 0$ and the enhancement layers are index from 1 to L . The video data in the ℓ^{th} layer of the f^{th} frame is called the $(f, \ell)^{\text{th}}$ video data unit.

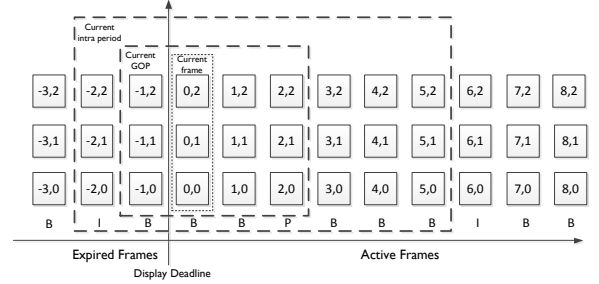


Fig. 3. Indices of data units when three quality layers are considered. At the beginning of each time slot, the frame with index $f = 0$ is played out. All the data units in the figure shift left.

C. Rate-Distortion Model

Let z_f be the amount of received data for the f^{th} frame. The rate-distortion function $d_f(z_f)$ captures the quality of the frame when it is decoded, given all its predictors have been received. Let $\omega_{(f, \ell)}$ be the amount of data in the $(f, \ell)^{\text{th}}$ data unit and $d_{(f, \ell)}$ be the distortion measured in mean square error (MSE) if the $0^{\text{th}} \sim \ell^{\text{th}}$ layers has been correctly received. As shown in Fig. 4, since a data unit can be decoded only when all its associated data has been received, $d_f(z_f)$ is a piecewise constant and right-continuous function with jumps at $z_f = \sum_{\ell=0}^m \omega_{(f, \ell)}$, $m = 0, 1, \dots, L$. Thus $d_{(f, \ell)}$ and $\omega_{(f, \ell)}$ characterize $d_f(z_f)$.

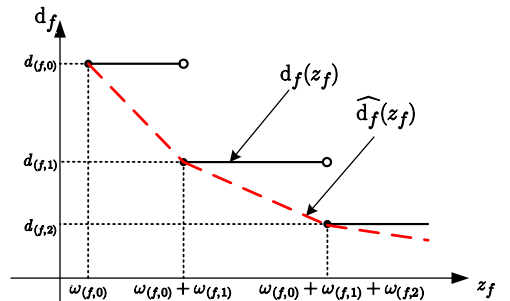


Fig. 4. An illustration of the rate-distortion function $d_f(z_f)$ for the f^{th} frame. The rate-distortion function $d_f(z_f)$ is piecewise constant and right-continuous (solid). Its convex envelope $\hat{d}_f(z_f)$ is also shown (dashed).

In a real video sequence, for a given layer ℓ , the rate-distortion characteristics $\omega_{(f,\ell)}$ and $d_{(f,\ell)}$ vary across frames. Let $\mathcal{K} = \{I, P, B^1, \dots, B^T\}$ be the set of frame types. We model $\omega_{(f,\ell)}$ of type k frames as i.i.d. realizations of a random variable Ω_ℓ^k , where $k \in \mathcal{K}$. Then, we use $\omega_\ell^k = \mathbb{E}[\Omega_\ell^k]$ as an estimate of $\omega_{(f,\ell)}$. Similarly, for a given layer ℓ , we model $d_{(f,\ell)}$ as i.i.d. realizations of a random variable D_ℓ . We use $d_\ell = \mathbb{E}[D_\ell]$ as an approximation of $d_{(f,\ell)}$. Here, we choose not to distinguish the frame types when modeling $d_{(f,\ell)}$. In a typical H.264/SVC video stream, the quantization parameters of the encoder are usually configured to minimize visually annoying quality fluctuations across different types of frames. Hence, for simplicity, we use a single random variable D_ℓ to model $d_{(f,\ell)}$ for all types of frames.

Our rate-quality models $d^k(z_f)$ for type- k frames are constructed as piecewise constant functions with jumps at $z_f = \sum_{\ell=0}^m \omega_\ell^k$, $m = 0, 1, \dots, L$. For stored video transmission, which is the focus of this paper, since the transmitted video has already been encoded, the size of each data unit is thus available. The parameters $\{\omega_\ell^k, k \in \mathcal{K}\}$ can thus be estimated by averaging across frames. If the distortion characteristic $d_{(f,\ell)}$ is calculated when the video is encoded, the parameter d_ℓ can also be estimated by averaging $d_{(f,\ell)}$ across frames. If $d_{(f,\ell)}$ is not available, d_ℓ needs to be estimated on-line. For example, the quality of frames that have been decoded at the receiver can be fed back to the transmitter for estimation.

D. Streaming Setup

We focus on scheduling for a slow fading channel. By slow fading, we mean that the coherence time of the channel is less than the duration of an intra period and larger than a frame slot. Assuming the mobile users are moving in a 1.5m/s walking speed and the carrier frequency is 2GHz, the Doppler spread is about 10Hz. The coherence time is about 100ms. A typical intra period duration is about 1 second and a frame slot is about 30ms. Hence, for pedestrian video users, wireless channels are slow fading.

As the channel state is stable during each frame slot, the scheduling decision is made on a frame-by-frame basis. At the beginning of each frame slot, a frame is played out, and video data units are scheduled for transmission. The scheduling action is defined as a set of ordered video data units

$$\mathcal{U} = \{(f_1, \ell_1), (f_2, \ell_2), \dots, (f_{|\mathcal{U}|}, \ell_{|\mathcal{U}|})\}. \quad (1)$$

When scheduling action \mathcal{U} is taken, the associated data units are transmitted sequentially. Each scheduled data unit is packetized into physical layer packets and each packet is repeatedly transmitted, i.e., if packet error occur, until acknowledged.

In this paper, we consider data unit level scheduling. If a packet level rate-quality model such as [10] is available, our MDP formulation can also be used to optimize the packet level scheduling policy.

III. MARKOV DECISION PROCESS-BASED MODEL

In this section, we propose an MDP-based model to determine the near-optimal scheduling policy. To that end, we

describe the scheduler's state space and the policies to be considered. We then show how to reduce the scheduling problem to a finite-state Markov decision problem using reasonable approximations. With the MDP-based model, the optimal scheduling policy is computed off-line via value iteration. The computed policy can then be used for on-line adaptive scheduling. To validate the optimality of the MDP-based scheduling policies, we develop a performance upper bound at the end of this section.

A. Scheduling Policy and State Space

Considering all possible scheduling actions makes defining the scheduling policy and representing the buffer state unmanageably complex. On one hand, to capture the buffer state, the frame index and the layer index of each received data unit need to be recorded. If we assume an infinite playback buffer, the number of received data units is not bounded. So we cannot represent all possible buffer states using a finite-dimensional space. On the other hand, not all possible scheduling policies need to be considered. For example, a quality layer of a frame should not be transmitted before the lower quality layers of the frame because a SVC decoder cannot decode a quality layer without the lower layers [1]. Thus we need only consider those scheduling strategies that are not dominated and have potential to achieve good performance.

Specifically, we consider scheduling policies under the following assumptions:

Assumption 1: The scheduler always schedules the base layer data unit of a frame for transmission after the base layer data unit(s) of the reference frame(s). The scheduler always schedules the enhancement layer data unit of a frame for transmission after the data units of the lower layers.

Assumption 2: The scheduler always schedule enough amount of data such that the transmitter is kept transmitting during the whole slot.

Assumption 3: We define three sets of data units: \mathcal{W}^{pre} , \mathcal{W} and $\mathcal{W}^{\text{post}}$. When the current frame is a B frame, the set \mathcal{W}^{pre} contains the data units with frame index $f \in [f^{\text{key}}, -1]$, where f^{key} is the frame index of the last expired key picture (see Fig. 5(a)). When the current frame is a key picture, we define $\mathcal{W}^{\text{pre}} = \emptyset$. Note that \mathcal{W}^{pre} contains all the expired data units that are used to predict the frames in the current GOP. The set \mathcal{W} contains the data units in all quality layers of the first W frames, where W is larger than F^{GOP} . The set $\mathcal{W}^{\text{post}}$ contains the remaining active data units. We assume the scheduler first sends the data units in \mathcal{W} . Then, if all the data in \mathcal{W} and the predictors in \mathcal{W}^{pre} have been received, the policy greedily schedules all $1+L$ quality layers of the frames in $\mathcal{W}^{\text{post}}$, i.e., starts transmitting the next frame in $\mathcal{W}^{\text{post}}$ only when all the layers of the preceding frame have been received (see Fig. 5(b)).

Assumption 4: In each slot, the scheduler only schedules data for the frames that have not been decoded.

Assumption 1 ensures that the transmission order is compatible with the prediction order given in Section II-B. Assumption 2 ensures the transmitter will not be idle during a slot and the capacity of the channel is fully exploited. Assumption 3

stems from the fact that, when many frames are buffered at the receiver, the scheduler can transmit more enhancement layers because there is sufficient time before the frames are played out. In other words, if all quality layers of W frames have been received, there is not need to worry about the channel capacity variation in the future. As will be discussed in Section. III-C, this assumption helps to simplify the policy optimization problem. It should be noted that policies under Assumption 3 are different from the sliding window policies defined in [5]. Indeed, our scheduling policy allows the transmitter to transmit data units outside the window. Assumption 4 ensures that the transmitter does not waste resources on the frames which have been decoded.

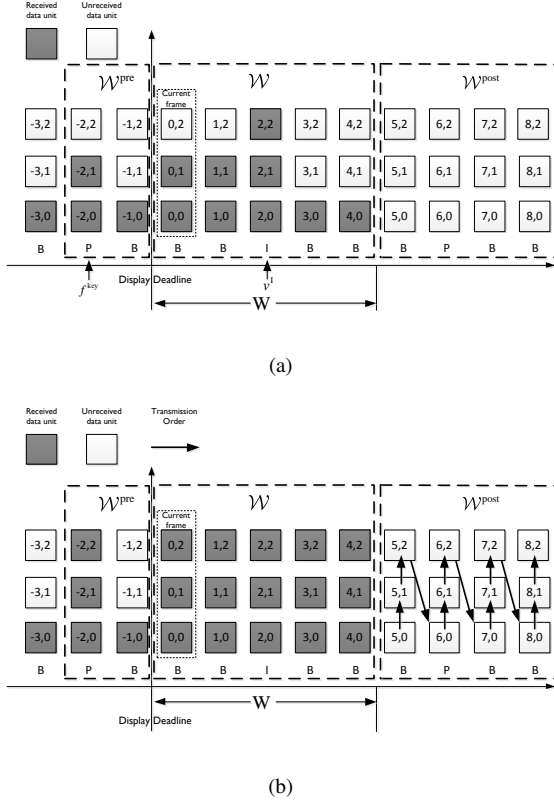


Fig. 5. (a): An illustration of the receiver buffer state when $F^{\text{intra}} = 8$, $F^{\text{GOP}} = 4$, $L = 2$, and $W = 5$. $v^I = 2$, $\mathbf{v}^{\text{pre}} = (2, 1)$, $\mathbf{v}^W = (2, 2, 3, 1, 1)$, $\mathbf{v}^{\text{post}} = (0, 0, 0)$. Note that because some data units in \mathcal{W} have not been received, the data units in $\mathcal{W}^{\text{post}}$ are not sent. (b): The transmission order when the data in \mathcal{W} and the associated predictors in \mathcal{W}^{pre} has been received.

Remark The window size W provides a tradeoff between complexity and optimality. The larger the window, the less constrained the control policy but the higher complexity¹. We note that, although the frames in the current GOP are played out sequentially, they are decoded together. According to Assumption 4, if $W \geq F^{\text{GOP}}$, the frames in \mathcal{W} have all been decoded and the scheduler cannot schedule any data from \mathcal{W} . Therefore, we set $W \geq F^{\text{GOP}}$.

We define the overall buffer state space \mathcal{V} via four sets \mathcal{V}^I , \mathcal{V}^{pre} , \mathcal{V}^W , and $\mathcal{V}^{\text{post}}$, where $\mathcal{V} = \mathcal{V}^I \times \mathcal{V}^{\text{pre}} \times \mathcal{V}^W \times \mathcal{V}^{\text{post}}$. The

set \mathcal{V}^I records the types and playout deadlines of the frames in the buffer. The sets \mathcal{V}^{pre} , \mathcal{V}^W , and $\mathcal{V}^{\text{post}}$ describe the states of the frames in \mathcal{W}^{pre} , \mathcal{W} and $\mathcal{W}^{\text{post}}$, respectively.

- \mathcal{V}^I : We define v^I as the frame index of the active I frame with the earliest playout deadline. Since the prediction structure is assumed to be the same for all intra periods, v^I determines the types and playout deadlines of all the frames in the receiver buffer.
- \mathcal{V}^{pre} : If the current frame is a B frame, the state space \mathcal{V}^{pre} is defined as a vector $\mathbf{v}^{\text{pre}} = (b_{f^{\text{key}}}^{\text{pre}}, \dots, b_{-1}^{\text{pre}})$, where b_f^{pre} is the number of the received quality layers in the f^{th} frame and f^{key} is the frame index of the last expired key picture. If the current frame is a key picture, $\mathcal{W}^{\text{pre}} = \emptyset$ and we define $\mathbf{v}^{\text{pre}} = -1$.
- \mathcal{V}^W : Similar to \mathcal{V}^{pre} , we define the buffer state space for \mathcal{W} as a vector $\mathbf{v}^W = (b_f^W, \dots, b_{-1}^W)$, where b_f^W is the number of the received quality layers in the f^{th} frame.
- $\mathcal{V}^{\text{post}}$: The set $\mathcal{W}^{\text{post}}$ contains infinite number of frames. Therefore, recording the number of data units received for each frame is impossible. We note that, when Assumption 3 is enforced, the number of data units received in $\mathcal{W}^{\text{post}}$ must be non-increasing in the frame index. Hence, we only need record the total number of received data units for each layer. We define the buffer state space of $\mathcal{W}^{\text{post}}$ as a $1+L$ -dimensional vector $\mathbf{v}^{\text{post}} = (b_0^{\text{post}}, b_1^{\text{post}}, \dots, b_L^{\text{post}})$, where b_ℓ^{post} is the number of the received data units in the ℓ^{th} layer of $\mathcal{W}^{\text{post}}$. Because the receiver buffer size is assumed to be large, i.e., essentially infinite, b_ℓ^{post} is unbounded. Thus $\mathcal{V}^{\text{post}} = \mathbb{N}^{1+L}$, where $\mathbb{N} = \{0, 1, \dots, \infty\}$.

With the above definition, buffer state $\mathbf{v} = (v^I, \mathbf{v}^{\text{pre}}, \mathbf{v}^W, \mathbf{v}^{\text{post}})$ contains all the information that is relevant to the quality of frames in the receiver buffer.

In [17] and [18], it is shown that a first-order finite state Markov chain (FSMC) can be used to describe the first-order channel state transition probabilities for Rayleigh fading channels. First-order FSMC models have also been validated in [19] and [20] by wireless channel measurements in urban areas. In our MDP-based model, we employ a first-order FSMC to describe the dynamics of the channel state.

We denote by x the transmission rate of the transmitter, i.e., the number of bits transmitted in a time slot ΔT . We denote by y the packet error rate of the channel. We define the channel state as $\mathbf{c} = (x, y)$. The channel state space is $\mathcal{C} = \{\mathbf{c}^1, \dots, \mathbf{c}^{|\mathcal{C}|}\}$, where $\mathbf{c}^i = (x^i, y^i)$ is the i^{th} channel state. The state transition matrix \mathbf{P}^c is a $|\mathcal{C}| \times |\mathcal{C}|$ matrix with entry $\mathbf{P}_{i,j}^c = \mathbb{P}(\mathbf{c}^j | \mathbf{c}^i)$ being the transition probability from state \mathbf{c}^i to \mathbf{c}^j .

The system state space \mathcal{S} is defined as the product of the buffer state space \mathcal{V} and the channel state space \mathcal{C} . For each state $\mathbf{s} \in \mathcal{S}$, we define a feasible control set $\mathcal{U}_{\mathbf{s}}$ that contains all the scheduling actions (see Equation (1)) complying with all the three assumptions. The state \mathbf{s} contains all the information about the receiver buffer and the channel. The transmitter must decide which action in $\mathcal{U}_{\mathbf{s}}$ to take in

¹In our simulations, we find that setting $W = 9$ is sufficient.

order to minimize the distortion. We define the scheduling policy $\mu(\cdot)$ as the mapping from the system state \mathbf{s} to an action in $\mathcal{U}_{\mathbf{s}}$. Under given scheduling policy μ , the system state transit as a controlled Markov chain. The state transition probability $\mathbb{P}_{\mu}(\cdot|\cdot)$ is determined by the scheduling policy μ (see Appendix A for detail). In the following sections, we show how to optimize the scheduling policy $\mu(\cdot)$.

B. Optimization Objective

Since the channel condition is modeled as a random process, we denote by $(C_t, V_t, S_t)_{t \in \mathbb{N}}$ the random processes modeling channel, buffer and system state, respectively. Accordingly, we denote $S = \lim_{t \rightarrow +\infty} S_t$. We define a function $d(\mathbf{s})$ of state \mathbf{s} as the estimated distortion of the frame that is played out at state \mathbf{s} .² Our aim is to find an optimal policy $\mu^*(\cdot)$ that minimizes the expectation of distortion, i.e.,

$$J_{\mu} = \mathbb{E}_{\mu}[d(S)], \quad (2)$$

where $\mathbb{E}_{\mu}[\cdot]$ is the expectation over the stationary distribution of the controlled Markov chain under policy μ .

We now introduce the definition of $d(\mathbf{s})$. If the displayed frame is a key picture (I frame or P frame), we estimate its distortion using the rate-distortion model in Section II-C as

$$d(\mathbf{s}) = \begin{cases} d^I(z(\mathbf{s})) & : \text{ for I frames} \\ d^P(z(\mathbf{s})) & : \text{ for P frames,} \end{cases} \quad (3)$$

where $z(\mathbf{s})$ denotes the amount of received data for the displayed frame at state \mathbf{s} . If the displayed frame is a B frame, which is encoded using all the $1 + L$ layers of its reference frames as predictor, the distortion cannot be directly estimated using the rate-distortion model in Section II-C because the loss in the enhancement layers of its reference frames causes encoder-decoder predictor mismatch, which is also known as drift in SVC [1]. We employ the model proposed in [21] to take into account the distortion due to drift. Let $\{\widehat{v}_i^{\text{ref}}, i \in 1, 2\}$ denote the predictor for the B frame at the encoder, i.e., the pixel value of the i^{th} reference frame with all the $1 + L$ layers. Let $\{\widehat{v}_i^{\text{ref}}, i \in 1, 2\}$ be the the predictor for the B frame at the decoder, i.e., the pixel value of the i^{th} reference frame with all the received quality layers. The drift of the reference frame is thus $\epsilon_i^{\text{dft}} = \widehat{v}_i^{\text{ref}} - v_i^{\text{ref}}$. In [21], it is shown that the MSE of a type-B^T frames can be estimated as:

$$\tilde{d}(\mathbf{s}) = d^{B^T}(z(\mathbf{s})) + \frac{1}{4}\mathbb{E}[(\epsilon_1^{\text{dft}})^2] + \frac{1}{4}\mathbb{E}[(\epsilon_2^{\text{dft}})^2] + \frac{1}{2}\mathbb{E}[\epsilon_1^{\text{dft}}\epsilon_2^{\text{dft}}], \quad (4)$$

where $d^{B^T}(z(\mathbf{s}))$ is the rate-distortion function defined in Section II-C and the other terms on the right hand side are the distortions due to drift. Since $\mathbb{E}[(\epsilon_1^{\text{dft}} - \epsilon_2^{\text{dft}})^2] = \mathbb{E}[(\epsilon_1^{\text{dft}})^2] + \mathbb{E}[(\epsilon_2^{\text{dft}})^2] - 2\mathbb{E}[\epsilon_1^{\text{dft}}\epsilon_2^{\text{dft}}] \geq 0$, we have $\mathbb{E}[\epsilon_1^{\text{dft}}\epsilon_2^{\text{dft}}] \leq \frac{1}{2}\mathbb{E}[(\epsilon_1^{\text{dft}})^2] + \frac{1}{2}\mathbb{E}[(\epsilon_2^{\text{dft}})^2]$. Thus, $\tilde{d}(\mathbf{s})$ is upper bounded by $d^{B^T}(z(\mathbf{s})) + \frac{1}{2}\mathbb{E}[(\epsilon_1^{\text{dft}})^2] + \frac{1}{2}\mathbb{E}[(\epsilon_2^{\text{dft}})^2]$. We use this upper bound as a proxy of the B frame's distortion in our MDP model. The function $d(\mathbf{s})$ is defined as

$$d(\mathbf{s}) = d^{B^T}(z(\mathbf{s})) + \frac{1}{2}\mathbb{E}[(\epsilon_1^{\text{dft}})^2] + \frac{1}{2}\mathbb{E}[(\epsilon_2^{\text{dft}})^2]. \quad (5)$$

²Since in each slot, a frame is played out before the scheduling actions are taken. Therefore, $d(\mathbf{s})$ is not a function of the actions taken at the state \mathbf{s} .

The term $\mathbb{E}[(\epsilon_1^{\text{dft}})^2]$ is estimate from the distortion of the reference frame as follows. Let v_i^{ref} be the original pixel value of the reference frame before encoding. The decoding error of the reference frame is thus $\widehat{\epsilon}_i^{\text{ref}} = \widehat{v}_i^{\text{ref}} - v_i^{\text{ref}} = (\widehat{v}_i^{\text{ref}} - v_i^{\text{ref}}) + (\widehat{v}_i^{\text{ref}} - v_i^{\text{ref}})$, where $\widehat{v}_i^{\text{ref}} - v_i^{\text{ref}} = \epsilon_i^{\text{dft}}$ is the distortion due to drift and $\widehat{v}_i^{\text{ref}} - v_i^{\text{ref}}$ is the distortion due to encoding. Assuming $\widehat{v}_i^{\text{ref}} - v_i^{\text{ref}}$ and $\widehat{v}_i^{\text{ref}} - v_i^{\text{ref}}$ are uncorrelated³, we have $\mathbb{E}[(\widehat{\epsilon}_i^{\text{ref}})^2] = \mathbb{E}[(\epsilon_i^{\text{dft}})^2] + \mathbb{E}[(\widehat{v}_i^{\text{ref}} - v_i^{\text{ref}})^2]$. Since the B frame is predicted by the L^{th} enhancement layer of the reference frame, we have $\mathbb{E}[(\widehat{v}_i^{\text{ref}} - v_i^{\text{ref}})^2] = d_L$. Denoting by $d_i^{\text{ref}}(\mathbf{s}) = \mathbb{E}[(\epsilon_i^{\text{ref}})^2]$ the distortion of the reference frame, we have

$$\mathbb{E}[(\epsilon_i^{\text{dft}})^2] = d_i^{\text{ref}}(\mathbf{s}) - d_L. \quad (6)$$

Substituting (6) into (5), we have

$$d(\mathbf{s}) = d^{B^T}(z(\mathbf{s})) + \frac{1}{2}[d_1^{\text{ref}}(\mathbf{s}) + d_2^{\text{ref}}(\mathbf{s})] - d_L. \quad (7)$$

The distortion of the reference frame $d_i^{\text{ref}}(\mathbf{s})$ can be recursively estimated using (3) and (7). Because the prediction structure is acyclic, the recursion terminates when the reference frame is a key picture and (3) applies.

It should be noted that, in (7), the distortion is overestimated using an upper bound of (4). As will be shown by the simulation results in Section.III-F. This overestimation does not sacrifice the quality of the decoded videos.

C. Finite State Problem Formulation

Since the state space $\mathcal{V}^{\text{post}}$ is infinite, the state space \mathcal{S} is also infinite. Optimizing the scheduling policy over this infinite-state space is intractable. We define a set \mathcal{W}^{buf} as the data units in window \mathcal{W} and their associated predictors in \mathcal{W}^{pre} . With Assumption 3, the scheduling policy is actually fixed when are all the data in \mathcal{W}^{buf} is received. We only need to determine the optimal scheduling policy for states where some of the video data in \mathcal{W}^{buf} has not been received, which is a finite state set. The system state, however, still evolves in the infinite state space \mathcal{S} . In the following, we show how to simplify this infinite state space problem to a finite-state problem.

We define the set of states where some of the video data in \mathcal{W}^{buf} has not been received as follows:

$$\mathcal{S}_{\mathcal{W}} = \{\mathbf{s} | \mathbf{s} \in \mathcal{S}, \mathcal{W}^{\text{buf}} \not\subseteq \mathcal{O}(\mathbf{s})\}, \quad (8)$$

where $\mathcal{O}(\mathbf{s})$ is the set of buffered video data units when the state is \mathbf{s} . We define another subset of \mathcal{S} as the complement of $\mathcal{S}_{\mathcal{W}}$:

$$\mathcal{S}_{\overline{\mathcal{W}}} = \{\mathbf{s} | \mathbf{s} \in \mathcal{S}, \mathcal{W}^{\text{buf}} \subseteq \mathcal{O}(\mathbf{s})\}. \quad (9)$$

For all the states in $\mathcal{S}_{\overline{\mathcal{W}}}$, all the video data units in \mathcal{W}^{buf} has been received.

Given a policy $\mu(\cdot)$, the system state evolves as a controlled Markov chain in set $\mathcal{S}_{\mathcal{W}} \cup \mathcal{S}_{\overline{\mathcal{W}}}$. Because the transmission rate is finite, the number of states in $\mathcal{S}_{\overline{\mathcal{W}}}$ which can be reached

³This assumption is empirically true. We calculated the correlation coefficient of $\widehat{v}_i^{\text{ref}} - v_i^{\text{ref}}$ and $\widehat{v}_i^{\text{ref}} - v_i^{\text{ref}}$ using the frames of test sequence ‘‘foreman’’, ‘‘Paris’’, and ‘‘bus’’. The average correlation coefficient is 0.05.

from \mathcal{S}_W in one step is also finite. We formally define this set of states as follows

$$\mathcal{S}_\Delta = \{s'|s' \in \mathcal{S}_{\bar{W}}; \exists s \in \mathcal{S}_W, s.t., \mathbb{P}_\mu(s'|s) > 0\}, \quad (10)$$

where $\mathbb{P}_\mu(s'|s)$ is the state transition probability under policy μ (for the expression for $\mathbb{P}_\mu(s'|s)$, see Appendix. A). Thus to move from \mathcal{S}_W into the set $\mathcal{S}_{\bar{W}}$, the system state first hits a state in \mathcal{S}_Δ and then stays in $\mathcal{S}_{\bar{W}}$ for some time. During this period, the decoded video distortion is always d_L , because all the layers in \mathcal{W}^{buf} are available. The evolution of the system when it moves into set $\mathcal{S}_{\bar{W}}$ affect the performance of the system. In general, the longer it stays in $\mathcal{S}_{\bar{W}}$, the better the performance is. Although the scheduling policy in $\mathcal{S}_{\bar{W}}$ is fixed as described in Assumption 3, the policy in \mathcal{S}_W determines how frequently the system state will hit $\mathcal{S}_{\bar{W}}$ and thus critically impacts the system performance.

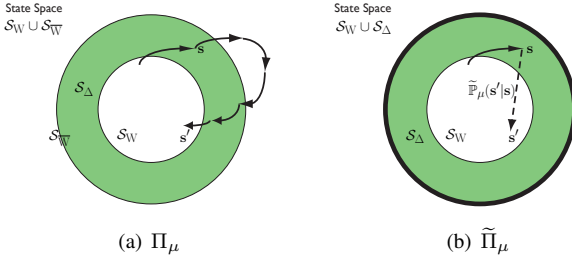


Fig. 6. The dynamics of the system Π_μ and the corresponding simplified system $\tilde{\Pi}_\mu$.

In the following, we denote the system under a given policy μ as system Π_μ . Let $t_\mu(s)$ be the expected time spent by Π_μ in $\mathcal{S}_{\bar{W}}$ after it enters $\mathcal{S}_{\bar{W}}$ at state $s \in \mathcal{S}_\Delta$. Let $\tilde{\mathbb{P}}_\mu(s'|s)$ denote the probability that Π_μ jumps back to \mathcal{S}_W at state $s' \in \mathcal{S}_W$ after it enters $\mathcal{S}_{\bar{W}}$ at state s . To find the optimal policy, we define a finite-state system $\tilde{\Pi}_\mu$ as follows

Definition 1: A system $\tilde{\Pi}_\mu$ is called the simplified system of the original system Π_μ if it has the following dynamics:

- 1) The system is a controlled semi-Markov process over state space $\tilde{\mathcal{S}} = \mathcal{S}_W \cup \mathcal{S}_\Delta$. In any state $s \in \tilde{\mathcal{S}}$, the distortion is $d(s)$ as in (3) and (7). In any state in \mathcal{S}_W , the system evolves according to the policy μ . The system state transition probability is $\mathbb{P}_\mu(\cdot|\cdot)$.
- 2) When the system jumps to a state $s \in \mathcal{S}_\Delta$, it spends $t_\mu(s)$ slots in s with distortion d_L for each slot. The system then transitions to a state $s' \in \mathcal{S}_W$ with probability $\tilde{\mathbb{P}}_\mu(s'|s)$ (see Fig. 6).

It should be noted that $\tilde{\Pi}_\mu$ is not coupled with the original system Π_μ . It just shares some properties with the original system. The following theorem relates the distortion under $\tilde{\Pi}_\mu$ and that of Π_μ .

Theorem 1: If the jump chain of the original system Π_μ is positive recurrent, then the time-average video distortion of Π_μ is the same as the simplified system $\tilde{\Pi}_\mu$.

Proof Sketch: If the jump chain is positive recurrent, the jump from \mathcal{S}_W to \mathcal{S}_Δ can partition the Markov process into i.i.d segments. We only need to optimize the policy μ to minimize the average distortion in each segment. Every

segment consists of two consecutive subsegments. During the first subsegment, $s \in \mathcal{S}_{\bar{W}}$. In the other subsegment, $s \in \mathcal{S}_W$. Because every state in $\mathcal{S}_{\bar{W}}$ has the same distortion d_L , we can abstract the first subsegment as a single state with transition probability $\tilde{\mathbb{P}}_\mu(\cdot)$. This simplified system provides the same average distortion as the original system. For a detailed proof, see the technical report [22]. ■

Remark The positive recurrent condition for the jump chain means that the average throughput of the channel is neither too large nor too small relative to the average data rate of the video. If the average throughput of the channel is very large, the receiver buffer can always buffer enough frames and dynamic scheduling is unnecessary. If the average channel throughput is too small, the channel cannot support the video stream and dynamic scheduling cannot help either.

As indicated by Theorem 1, given any policy μ , the video distortion of Π_μ is the same as $\tilde{\Pi}_\mu$. Thus, we can optimize our policy with respect to $\tilde{\Pi}_\mu$, which has a finite-state space, and a standard policy optimization algorithm can be applied.

Before we can apply an MDP algorithm to optimize the policy, we need to compute $t_\mu(s)$ and $\tilde{\mathbb{P}}_\mu(s'|s)$ for every state $s \in \mathcal{S}_\Delta$ and $s' \in \mathcal{S}_W$. Both $t_\mu(s)$ and $\mathbb{P}_\mu(s'|s)$ only involve dynamics of the system in $\mathcal{S}_{\bar{W}}$. Details on how to compute $t_\mu(s)$ and $\tilde{\mathbb{P}}_\mu(s'|s)$ are found in [22].

D. Determining Optimal Policy via Value Iteration

Given $t_\mu(\cdot)$ and $\tilde{\mathbb{P}}_\mu(\cdot|\cdot)$, the optimal policy for an MDP can be determined for the simplified system $\tilde{\Pi}_\mu$, which is also the optimal policy of Π_μ . Let s^{ini} be any state in $\tilde{\mathcal{S}} = \mathcal{S}_W \cup \mathcal{S}_\Delta$. The hitting time to state s^{ini} can partition the process into i.i.d cycles. Optimizing the policy $\mu(\cdot)$ in the cycles minimizes the time-average video distortion of the system. Similar to the derivation in [23, p. 441], this is equivalent to an average-cost minimization problem with stage-cost $(d(s) - \lambda)\eta(s)$, where λ is the expected average-cost of each cycle, i.e., the average distortion. The function $\eta(s)$ is defined as

$$\eta(s) = \begin{cases} 1 & : s \in \mathcal{S}_W \\ t_\mu(s) & : s \in \mathcal{S}_\Delta, \end{cases}$$

Note that $d(s)$ is the cost of spending one slot on state s and λ is the expected cost per slot. Therefore, $d(s) - \lambda$ is the extra-cost of spending one slot on state s . Since $\eta(s)$ is the average time spent on state s , $(d(s) - \lambda)\eta(s)$ is the total extra-cost of visiting state s . Let us denote by $h(s)$ the average cost-to-go in each cycle when the system starts at state s . Then we have the following Bellman's equation array:

$$h(s) = \begin{cases} (d(s) - \lambda)\eta(s) + \sum_{s' \in \mathcal{S}_W \cup \mathcal{S}_\Delta} \mathbb{P}_\mu(s'|s)h(s'), & \text{if } s \in \mathcal{S}_W \\ (d(s) - \lambda)\eta(s) + \sum_{s' \in \mathcal{S}_W \cup \mathcal{S}_\Delta} \tilde{\mathbb{P}}_\mu(s'|s)h(s'), & \text{if } s \in \mathcal{S}_\Delta \end{cases} \quad (11)$$

where $h(s^{\text{ini}}) = 0$. To find the optimal policy, the standard value iteration algorithm can be applied [23, p. 430].

On the one hand, the assumptions on scheduling policy result in the finite state MDP-based formulation. On the other

hand, the assumptions may render the derived scheduling policy sub-optimal. To verify the performance of the scheduling policy derived from the MDP formulation is actually close to optimal, we prove a performance upper bound in the next section.

E. Performance Upper Bound

As discussed in Section. II-C, $\{d^k(z_t), k \in \mathcal{K}\}$ are the rate quality models of type- k frames when all the predictors have also been received. Since $d^k(z_t)$ does not incorporate the distortion due to drift, the time-average distortion of the transmitted video is at least $\frac{1}{n} \sum_{t=1}^n \sum_{k \in \mathcal{K}} d^k(z_t) \mathbb{1}_t^k$, where n is the number of frames in the video sequence and $\mathbb{1}_t^k$ is the indicator that the t^{th} frame is a type- k frame. Let r_t be the amount of data that is received in the t^{th} slot, a distortion lower bound of any scheduler is given by the following offline optimization problem:

$$\begin{aligned} & \underset{z_{1:n}}{\text{minimize}} \quad \frac{1}{n} \sum_{k \in \mathcal{K}} \sum_{t=1}^n d^k(z_t) \mathbb{1}_t^k \\ & \text{s.t.} \quad \frac{1}{t} \sum_{i=1}^t z_i \leq \frac{1}{t} \sum_{i=1}^t r_i, \quad \forall t \in \{1, 2, \dots, n\}, \end{aligned} \quad (12)$$

where the constraint $\frac{1}{t} \sum_{i=1}^t z_i \leq \frac{1}{t} \sum_{i=1}^t r_i$ guarantees that the received data for the frames displayed before time t does not exceed the cumulative throughput prior to time t . We can further relax the constraints in (12) by only keeping the last one, i.e., when $t = n$. The relaxed optimization problem is then given by

$$\begin{aligned} & \underset{z_{1:n}}{\text{minimize}} \quad \frac{1}{n} \sum_{k \in \mathcal{K}} \sum_{t=1}^n d^k(z_t) \mathbb{1}_t^k \\ & \text{s.t.} \quad \frac{1}{n} \sum_{t=1}^n z_t \leq \frac{1}{n} \sum_{t=1}^n r_t. \end{aligned} \quad (13)$$

Let $\widehat{d^k}(z_t)$ be the convex envelope of $d^k(z_t)$ (see Fig. 4). Since, $d^k(z_t)$ are lower bounded by $\widehat{d^k}(z_t)$, we can bound problem (13) by:

$$\begin{aligned} & \underset{z_{1:n}}{\text{minimize}} \quad \frac{1}{n} \sum_{k \in \mathcal{K}} \sum_{t=1}^n \widehat{d^k}(z_t) \mathbb{1}_t^k \\ & \text{s.t.} \quad \frac{1}{n} \sum_{t=1}^n z_t \leq \frac{1}{n} \sum_{t=1}^n r_t. \end{aligned} \quad (14)$$

Let $n^k = \sum_{t=1}^n \mathbb{1}_t^k$ denotes the number of type- k frames. Since the functions $\widehat{d^k}(z_t)$ are convex, by Jensen's inequality, we have

$$\frac{1}{n^k} \sum_{t=1}^n \widehat{d^k}(z_t) \mathbb{1}_t^k \geq \widehat{d^k} \left(\frac{1}{n^k} \sum_{t=1}^n z_t \mathbb{1}_t^k \right)$$

Problem (14) can then be bounded by:

$$\begin{aligned} & \underset{z_{1:n}}{\text{minimize}} \quad \sum_{k \in \mathcal{K}} \frac{n^k}{n} \widehat{d^k} \left(\frac{1}{n^k} \sum_{t=1}^n z_t \mathbb{1}_t^k \right) \\ & \text{s.t.} \quad \sum_{k \in \mathcal{K}} \frac{n^k}{n} \left(\frac{1}{n^k} \sum_{t=1}^n z_t \mathbb{1}_t^k \right) \leq \frac{1}{n} \sum_{t=1}^n r_t. \end{aligned} \quad (15)$$

If the video is reasonably long, e.g. several minutes, the frame number n will be very large. If we let $n \rightarrow \infty$ and assume the channel throughput r_t is ergodic, $\frac{1}{n} \sum_{t=1}^n r_t$ will converge to the ergodic capacity $r^{\text{avg}} = \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{t=1}^n r_t$. Furthermore, let F^k denote the number of type- k frames in a intra period. We have $\frac{n^k}{n} \rightarrow \frac{F^k}{F^{\text{intra}}}$. Similarly, for stationary policies⁴, the limits $z^k = \lim_{n \rightarrow \infty} \frac{1}{n^k} \sum_{t=1}^n z_t \mathbb{1}_t^k$ exist. We have $\lim_{n \rightarrow \infty} \left[\frac{n^k}{n} \left(\frac{1}{n^k} \sum_{t=1}^n z_t \mathbb{1}_t^k \right) \right] = \frac{F^k}{F^{\text{intra}}} z^k$. Thus, we have shown the following theorem:

Theorem 2: For ergodic wireless throughput and stationary adaptive scheduling policies the following optimization gives an upper bound on performance (lower bound of distortion):

$$\begin{aligned} & \underset{z^k, k \in \mathcal{K}}{\text{minimize}} \quad \sum_{k \in \mathcal{K}} \frac{F^k}{F^{\text{intra}}} \widehat{d^k}(z^k) \\ & \text{s.t.} \quad \sum_{k \in \mathcal{K}} \frac{F^k}{F^{\text{intra}}} z^k \leq r^{\text{avg}}. \end{aligned} \quad (16)$$

Since the rate-distortion function $\widehat{d^k}(\cdot)$ is assumed to be convex, the above optimization problem is convex and easily solved. In Section III-F, this performance bound will be employed as a benchmark to evaluate the performance of our MDP-based scheduling policy.

F. Performance Evaluation of the MDP-based Scheduling Policy

In this section, we evaluate the performance of the policy obtained from our MDP-based formulation. The algorithm was evaluated on test sequences “foreman”, “bus”, “flower”, “mobile” and “Paris” [24]. These video sequences were encoded using H.264/SVC reference software JSVM [25] with a base layer and a CGS enhancement layers. The intra-period and IDR period were set to $F^{\text{intra}} = 16$. The GOP length was fixed at $F^{\text{GOP}} = 4$. The quantization parameter (QP) of the base layer, denoted by QP^{base} , were chosen such that the data rate of the base layer is lower than the average channel throughput. The QP of the CGS enhancement layer is set as $\text{QP}^{\text{base}} - 10$. We employ this configuration to make sure that the channel is at least good enough to support the base layer. Otherwise, any scheduling policy cannot provide acceptable video quality. The CGS is split into two MGS layers. The first MGS layer contains 6 of the 16 transform coefficients of the CGS layer. The other 10 coefficient belongs to the second MGS layer. The QPs and rate-distortion model parameters of the encoded video sequences are shown in table II. Parallel to [9] and [10], we employ the FSMC channel model proposed in [17] to model the dynamics of Rayleigh fading channels. The SNR at the receiver is partitioned into 4 regions using the algorithm proposed in [17]. In our simulations, we set the average SNR to $\Lambda^{\text{avg}} = 10\text{dB}$. For each sequence, 200 transmissions were sent over the simulated channel. A startup delay constraint was fixed to 200ms, i.e., video playback began 6 frames after the transmission began. After each transmission, a trace file

⁴A policy is called stationary if it is a function of state \mathbf{s} and the function is invariant with respect to time t .

that recorded the packet loss in each time slot was generated. We used the bitstream extractor of JSVM to remove those dropped packets. The extracted bitstreams were decoded using the JSVM decoder with frame copy error concealment. For more details about the FSMC channel model, see Appendix B.

The performance of the MDP-based scheduling algorithm was tested over the simulated Markov channel models with different Doppler frequencies ($f^d = 5\text{Hz}$ and 3Hz , respectively). The simulation results are summarized in Table III and Table IV. The visual quality is measured via the MS-SSIM index which correlates well with human objective judgments [26]. The time-averaged MS-SSIM value is further converted to Difference Mean Opinion Score (DMOS) using the following mapping

$$q^{\text{dmos}} = 13.3442 \log(1 - q^{\text{ssim}}) + 3.6226(1 - q^{\text{ssim}}) + 77.0117, \quad (17)$$

where q^{ssim} denotes the time averaged quality measured in MS-SSIM and q^{dmos} is the corresponding DMOS value. Equation (17) is obtained by logistic regression using the MS-SSIM indices and MOS values of the images in the LIVE database [27]. DMOS ranges from 0 to 100. Value 0 means perfect visual quality and value 100 means bad visual quality. Roughly speaking, value 50 means fair quality. It can be seen from Table III and Table IV that the DMOS value of the MDP-based scheduling policy is worse than the performance bound by at most 2, which is visually insignificant. Given that the bound given by Theorem 2 is an upper bound (i.e. a lower bound of DMOS value), the MDP-based scheduling policy is indeed near-optimal.

TABLE III

THE PERFORMANCE OF THE NEAR-OPTIMAL POLICY IN SSIM-PREDICTED DMOS. $f^d = 5$.

	Paris	mobile	flower	bus	foreman
MDP Policy	26.9020	38.9033	34.5826	41.8721	32.2426
Upper bound	25.6017	38.0842	34.0626	41.4600	31.6807

TABLE IV

THE PERFORMANCE OF THE NEAR-OPTIMAL POLICY IN SSIM-PREDICTED DMOS. $f^d = 3$.

	Paris	mobile	flower	bus	foreman
MDP Policy	27.1376	39.0452	35.7828	42.0808	32.4431
Upper bound	25.2314	37.9431	33.7052	41.2611	31.4852

IV. NEAR-OPTIMAL HEURISTIC ON-LINE SCHEDULING ALGORITHM

Although the MDP-based formulation makes it possible to compute a good scheduling policy using value iteration algorithm, off-line computation of such policies requires *a priori* knowledge of the channel dynamics. This motivates us to design a simple on-line scheduling policy that delivers similar performance as the MDP-based policy that only requires little *a priori* knowledge about the channel dynamics.

A good online video scheduling algorithm should explicitly take advantage of the channel dynamics and schedule data from different quality layers as a function of the receiver buffer state. There are three fundamental questions in designing such a scheduler: 1) How should one incorporate limited knowledge of channel dynamics in adaptive scheduling? 2) How should one determine the number of enhancement layers to schedule? 3) How should one allocate appropriate transmission rate among current and future intra periods? In the following, we will show how to address these fundamental problems by reasonably simplifying the MDP-based scheduling algorithm.

A. Channel Model Simplification

In a practical wireless communication environment, accurate channel dynamics models such as the state transition probability \mathbf{P}^c are not generally available. Some basic characteristics for the channel dynamics can, however, be easily used. At any slot t , the instantaneous channel throughput r_t can be estimated using receiver channel state information as

$$\hat{r}_t = x_t(1 - y_t),$$

where (x_t, y_t) is the channel state at t (see Section. III-A). The ergodic channel throughput r^{avg} can be estimated by averaging \hat{r}_t over time. If we model r_t as the realization of a random process $\{R_t, t \in \mathbb{N}\}$, the temporal correlation coefficient $\rho = \frac{\text{cov}(R_t, R_{t+1})}{\sigma(R_t)\sigma(R_{t+1})}$ can also be estimated from \hat{r}_t . Further it is reasonable to assume the channel throughput R_t will typically regress to the mean r^{avg} . This inspires us to use a simple autoregressive model to capture the dynamics of the channel. A first order autoregressive model (AR(1)) for R_t is given as,

$$R_t - \phi R_{t-1} = c + N_t, \quad (18)$$

where N_t is an i.i.d random variable with zero mean value. From (18), parameter c and ϕ can be estimated as $\phi = \rho$ and $c = r^{\text{avg}}(1 - \rho)$ [28][p. 115]. Thus, we have

$$R_t - \rho R_{t-1} = r^{\text{avg}}(1 - \rho) + N_t. \quad (19)$$

Using this autoregressive model, the amount of data that will be delivered in the next ζ slots by the channel can be estimated as

$$g(\hat{r}_t) = \mathbb{E} \left[\sum_{a=0}^{\zeta-1} R_{t+a} \middle| R_t = \hat{r}_t \right] = \sum_{a=0}^{\zeta-1} [\hat{r}_t \rho^a + r^{\text{avg}}(1 - \rho^a)]. \quad (20)$$

To obtain an accurate estimate in the near future, we set the length of the window ζ into the future that will be considered to be the relaxation time⁵ of the channel, i.e. , $\zeta = \lceil -(\ln \rho)^{-1} \rceil$. In the following, we use this to determine which quality layers to schedule.

⁵The relaxation time is defined as the temporal distance at which the temporal correlation coefficient is reduced to $\frac{1}{e}$

TABLE II
THE ENCODING PARAMETERS AND RATE-DISTORTION MODEL PARAMETERS OF THE TESTED SEQUENCES.

sequences	layer 0 (base layer)				Layer 1				Layer 2			
	QP	$\omega_0^I, \omega_0^P, \omega_0^{B^1}, \omega_0^{B^2}$ /Byte	d_0 /MSE		$\omega_1^I, \omega_1^P, \omega_1^{B^1}, \omega_1^{B^2}$ /Byte	d_1 /MSE			$\omega_2^I, \omega_2^P, \omega_2^{B^1}, \omega_2^{B^2}$ /Byte	d_2 /MSE		
foreman	30	6712, 2499, 928, 520	16.27		8302, 8293, 3373, 2775	5.491			5844, 5773, 2177, 1893	4.124		
bus	38	5920, 2417, 889, 568	100.8		7837, 8003, 3390, 2925	41.35			4636, 4412, 1577, 1339	21.65		
flower	40	8261, 2076, 548, 324	172.1		6786, 6900, 1951, 1611	96.66			6633, 6610, 2008, 1545	30.85		
mobile	40	9648, 1556, 510, 262	186.0		9090, 9193, 2541, 2171	89.90			7627, 6894, 1973, 1701	37.35		
Paris	32	12353, 2640, 865, 463	32.33		9850, 9457, 2103, 1571	18.59			8091, 7987, 2024, 1555	5.420		

B. Layer Selection

Given the current channel state, receiver buffer state, and estimated available capacity for a window ζ into the future, the goal is to determine which layers to schedule. We will focus on determining the number of enhancement layers which should be scheduled. We denote by $L^{\text{sch}}(s_t)$ the number of layers to be scheduled if the state is s_t . Once $L^{\text{sch}}(s_t)$ is determined, the online scheduling algorithm only schedules data units from the first $L^{\text{sch}}(s_t)$ layers.

The layer selection scheme for our proposed on-line algorithm is motivated by that of the MDP-based policy. Using $g(\hat{r}_t)$ defined in (20), we can estimate the amount of data which can be delivered in the next ζ slots. Let $\Gamma(\ell, s_t)$ be the amount of data which is not currently available at the playback buffer at time t , and belongs to the first ℓ layers of the next ζ frames that have not been decoded. The quantities $g(\hat{r}_t)$ and $\Gamma(\ell, s_t)$ summarize the channel and buffer states for the next ζ slots. Note that $\Gamma(\ell - 1, s_t) \leq g(\hat{r}_t) < \Gamma(\ell, s_t)$ means that we can probably transmit all the data up to the ℓ^{th} layer in the next ζ slots. Intuitively, we can simply choose $L^{\text{sch}}(s_t) = \ell - 1$ when $\Gamma(\ell - 1, s_t) \leq g(\hat{r}_t) < \Gamma(\ell, s_t)$. As discussed next, this layer selection scheme can be motivated by the near-optimal scheduling policies computed for the MDP-based model.

Note that $\hat{r}_t = x_t(1 - y_t)$ is determined by state s_t , thus $g(\hat{r}_t)$ can also be written as function of s_t , i.e., $g(s_t)$. Suppose we partition the state space into subsets $\mathcal{P}^\ell = \{s \in \mathcal{S} : \Gamma(\ell - 1, s) \leq g(s) < \Gamma(\ell, s)\}, \ell \in \{1, \dots, L + 2\}$ and calculate the fraction of states in \mathcal{P}^ℓ where the MDP-based policy only schedules the first $\ell - 1$ layers⁶. As shown in Fig. 7, for 71% of the states of \mathcal{P}^1 and \mathcal{P}^2 , the MDP-based policy only schedules the first layer. For about 65% of the states of \mathcal{P}^3 , the MDP-based policy only schedules the first 2 layers. Finally the MDP-based policy will schedule all the layers on 81% of the states in \mathcal{P}^4 . These observations justify our intuition regarding layer selection. In our proposed on-line scheduling algorithm, we will simply choose $L^{\text{sch}}(s_t) = \ell - 1$ if $\Gamma(\ell - 1, s_t) \leq g(\hat{r}_t) < \Gamma(\ell, s_t)$. In other words, our heuristic algorithm determines $L^{\text{sch}}(s_t)$ by roughly estimating the number of layers which can be transmitted.

C. Resource Allocation Between Current and Future Intra Periods

In each transmission slot, about \hat{r}_t bits of video data are delivered to the receiver. In the following, we refer to \hat{r}_t as the budget for slot t . Once $L^{\text{sch}}(s_t)$ is determined, we still

⁶We define $\Gamma(L + 2, s_t) = +\infty$

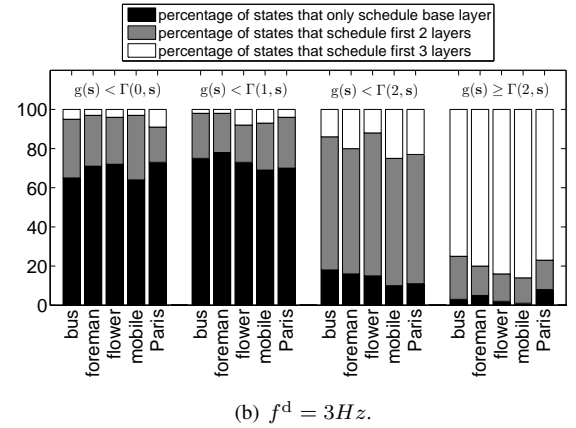
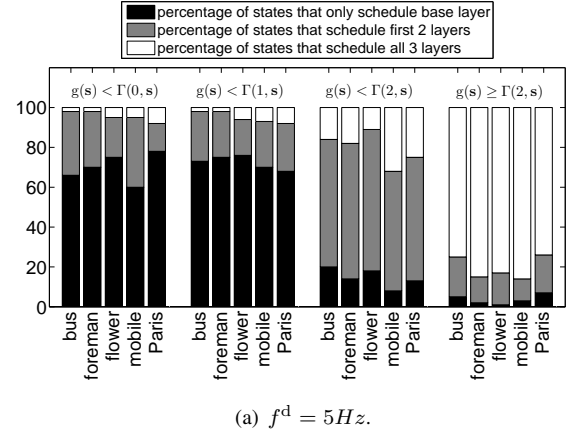


Fig. 7. Given different relationship between $g(s)$ and $\Gamma(\ell, s)$, the proportions of states corresponding to different $L^{\text{sch}}(s)$ are shown in different colors. Results are obtained under Rayleigh fading channels with different Doppler shifts (5Hz in (a) and 3Hz in (b)) and are calculated on 5 different video sequences (“bus”, “foreman”, “flower”, “mobile”, “Paris”).

need to determine how to allocate this budget among current and future intra periods. Sometimes it is necessary to transmit data associated with next I frame before the data units in the current intra period. For example, when the next I frame is approaching its display deadline and its base layer has not yet been received, if we focus on transmitting the frames in the current intra period sequentially, this increases the risk that the next I frame can not be decoded before its deadline. This in turn would cause severe decoding failures throughout the next intra period.

Let \mathcal{I} be the data units in the undecoded I frame that has the earliest display deadline. We denote by $\Psi^{\text{cur}}(\ell, s_t)$ the amount

of unreceived data in the first ℓ^{th} layer of current intra period at state \mathbf{s}_t . We denote by $\Psi^I(\ell, \mathbf{s}_t)$ the amount of unreceived data in the first ℓ^{th} layer of \mathcal{I} at state \mathbf{s}_t . We propose the following heuristic for allocating the bit budget between current intra period and \mathcal{I} . In each transmission slot, the scheduling algorithm allocates up to $\Omega_t = \frac{\Psi^I(\mathbf{L}^{\text{sch}}(\mathbf{s}_t), \mathbf{s}_t)}{\Psi^{\text{cur}}(\mathbf{L}^{\text{sch}}(\mathbf{s}_t), \mathbf{s}_t) + \Psi^I(\mathbf{L}^{\text{sch}}(\mathbf{s}_t), \mathbf{s}_t)}$ of the transmission bit budget to \mathcal{I} . In other words, the number of bits allocated to \mathcal{I} is $\min(\Omega_t \times \hat{r}_t, \Psi^I(\mathbf{L}^{\text{sch}}(\mathbf{s}_t), \mathbf{s}_t))$.

Here Ω_t gives the relative importance of the next I frame and current intra period. If $\Psi^I(\mathbf{L}^{\text{sch}}(\mathbf{s}_t), \mathbf{s}_t) = 0$, then $\Omega_t = 0\%$. It is not necessary to transmit any data for the next I frame. If $\Psi^{\text{pre}}(\mathbf{L}^{\text{sch}}(\mathbf{s}_t), \mathbf{s}_t) = 0$, then $\Omega_t = 100\%$. We only focus on transmitting the future intra periods.

The online scheduling algorithm is summarized in Algorithm 1.

Algorithm 1 On-line adaptive scheduling algorithm

Input: $\mathbf{s}_t, r^{\text{avg}}, x_t, y_t$, and ρ

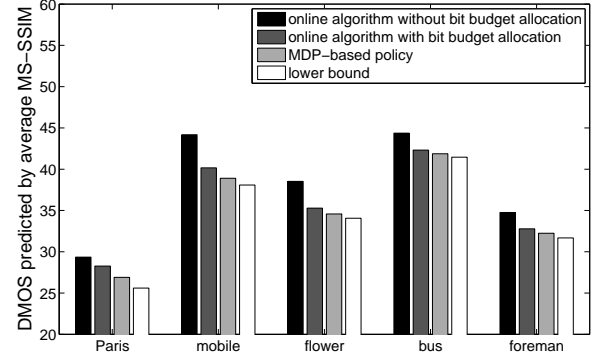
```

1:  $\zeta = \lceil -(\ln \rho)^{-1} \rceil; \hat{r}_t = x_t(1 - y_t)$ 
2: loop  $t$ 
3:    $g(\hat{r}_t) \leftarrow \sum_{a=0}^{\zeta-1} [\hat{r}_t \rho^a + r^{\text{avg}}(1 - \rho^a)]$ 
4:   for  $\ell = 1 \rightarrow L + 1$  do
5:     Compute  $\Gamma(\ell, \mathbf{s}_t)$ 
6:     if  $g(\hat{r}_t) < \Gamma(\ell, \mathbf{s}_t)$  then
7:       break
8:     end if
9:   end for
10:  if  $\ell=1$  then
11:     $\mathbf{L}^{\text{sch}}(\mathbf{s}_t) \leftarrow 1$ 
12:  else
13:     $\mathbf{L}^{\text{sch}}(\mathbf{s}_t) \leftarrow \ell - 1$ 
14:  end if
15:  Compute  $\Psi^{\text{cur}}(\mathbf{L}^{\text{sch}}, \mathbf{s}_t)$  and  $\Psi^I(\mathbf{L}^{\text{sch}}, \mathbf{s}_t)$ 
16:   $\Omega_t \leftarrow \frac{\Psi^I(\mathbf{L}^{\text{sch}}, \mathbf{s}_t)}{\Psi^{\text{cur}}(\mathbf{L}^{\text{sch}}, \mathbf{s}_t) + \Psi^I(\mathbf{L}^{\text{sch}}, \mathbf{s}_t)}$ 
17:  Schedule  $\min(\Omega_t \times \hat{r}_t, \Psi^I(\mathbf{L}^{\text{sch}}, \mathbf{s}_t))$  bits from  $\mathcal{I}$ .  $\triangleright$ 
    Scheduling data
18:  Schedule  $\hat{r}_t - \min(\Omega_t \times \hat{r}_t, \Psi^I(\mathbf{L}^{\text{sch}}, \mathbf{s}_t))$  bits from
    other active frames.
19: end loop
```

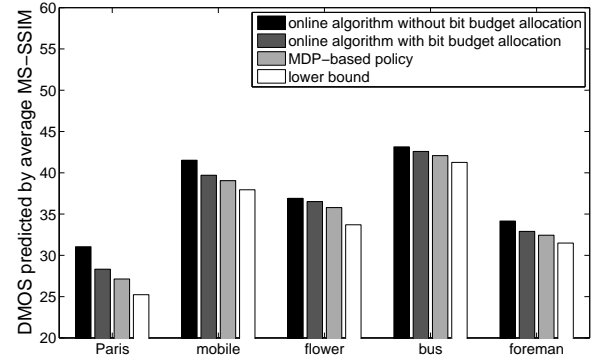
D. Performance Evaluation of the On-line Scheduling Algorithm

The performance of the on-line scheduling algorithm was tested over the simulated Markov channel models with different Doppler frequencies ($f^d = 5\text{Hz}$ and 3Hz , respectively). This setting is the same as the simulation setting in Section III-F. The results are summarized in Fig. 8. As can be seen, the performance of the proposed online-scheduling algorithm is almost as good as the MDP-based scheduling algorithm. Moreover the online scheduling algorithm's performance is close to the bound given by Theorem 2. We conclude it is a near-optimal scheduling algorithm.

We have also tested the performance of the online algorithm without bit budget allocation between current and future intra periods. As can be seen, the performance is worse than the



(a) $f^d = 5\text{Hz}$.



(b) $f^d = 3\text{Hz}$.

Fig. 8. Performance comparison of different scheduling algorithms. Video quality is measured in DMOS which is predicted by MS-SSIM using equation (17).

MDP-based scheduling policy and the performance bound. This motivates the necessity of allocating bit between current and future intra periods.

V. CONCLUSIONS

We have developed adaptive scheduling algorithms for stored scalable video transmission in wireless channels. By modeling the wireless channel as a Markov chain, an MDP model is proposed in which policies that minimize the distortion of decoded videos can be computed. By simplifying the scheduling algorithm obtained from the MDP formulation, we propose an online scheduling algorithm which only requires limited knowledge of channel dynamics. Simulation results demonstrate the near-optimality of the proposed online scheduling policy versus a proposed bound on performance.

APPENDIX A TRANSITION PROBABILITY

Notations: Let $\mathbf{1}$ be the unit vector of all-ones and $\mathbf{0}$ be the zero vector. $\max\{\mathbf{a}, \mathbf{b}\}$ is the componentwise maximum of vector \mathbf{a} and \mathbf{b} . $\mathbb{1}(\cdot)$ is the indicator function.

Let $\mathbf{s}_t = (\mathbf{c}_t, \mathbf{v}_t)$ and $\mathcal{U}_{\mathbf{s}_t}$ be the system state and the corresponding feasible control set at slot t , where $\mathbf{c}_t = (x_t, y_t)$ and $\mathbf{v}_t = (v_t^I, \mathbf{v}_t^{\text{pre}}, \mathbf{v}_t^W, \mathbf{v}_t^{\text{post}})$. At the beginning

of each slot, one frame is decoded and played out. Let $\mathbf{v}_t^+ = (v_t^{I+}, \mathbf{v}_t^{\text{pre}+}, \mathbf{v}_t^{W+}, \mathbf{v}_t^{\text{post}+})$ denote the buffer state right after the first frame is displayed. For \mathbf{v}_t^{I+} , we have

$$\mathbf{v}_t^{I+} = \begin{cases} F^{\text{intra}} - 1 & \text{if } v_t^I = 0, \\ v_t^I - 1 & \text{if } v_t^I \neq 0, \end{cases} \quad (21)$$

The first frame in \mathcal{W} is moved into \mathcal{W}^{pre} , thus we have

$$\mathbf{v}_t^{\text{pre}+} = (b_{f_{\text{key}}}^{\text{pre}}, \dots, b_{-1}^{\text{pre}}, b_0^W). \quad (22)$$

The first frame in $\mathcal{W}^{\text{post}}$ moves into \mathcal{W} . Thus, we have

$$\mathbf{v}_t^{W+} = \left(b_1^W, \dots, b_{W-1}^W, \sum_{\ell=0}^L \mathbb{1}(b_\ell^{\text{post}} \geq 1) \right), \quad (23)$$

For the set $\mathcal{W}^{\text{post}}$, once the current frame is played out, we have

$$\mathbf{v}_t^{\text{post}+} = \max \{ \mathbf{v}_t^{\text{post}} - \mathbf{1}, \mathbf{0} \} \quad (24)$$

After the first frame is displayed, the transmitter begins to sequentially transmit the collection of video data units indicated by the action $\mathcal{U}_t = \mu(\mathbf{s}_t) = \{(f_1, \ell_1), \dots, (f_{|\mathcal{U}_t|}, \ell_{|\mathcal{U}_t|})\}$. Let $\Delta\mathcal{U}_t = \{(f_1, \ell_1), \dots, (f_{n_t}, \ell_{n_t})\}$ denote the completely received data units by the end of the slot, where n_t is the number of received data units. Among the data units in $\Delta\mathcal{U}_t$, let $\Delta\mathbf{v}_t^{\text{pre}}$ and $\Delta\mathbf{v}_t^W$ be the number of newly received data units for each frame in set $\mathcal{W}^{\text{pre}+}$ and \mathcal{W}^{W+} , respectively. At the beginning of the $(t+1)^{\text{th}}$ slot, we have the following state transition relationship

$$\mathbf{v}_{t+1}^{\text{pre}} = \mathbf{v}_t^{\text{pre}+} + \Delta\mathbf{v}_t^{\text{pre}}, \quad (25)$$

$$\mathbf{v}_{t+1}^W = \mathbf{v}_t^{W+} + \Delta\mathbf{v}_t^W. \quad (26)$$

Similarly, we denote by $\Delta\mathbf{v}_t^{\text{post}} = (\Delta b_0^{\text{post}}, \dots, \Delta b_L^{\text{post}})$ the number of newly received data units for each layer in frame set $\mathcal{W}^{\text{post}+}$. The state transition relationship of $\mathcal{W}^{\text{post}}$ is

$$\mathbf{v}_{t+1}^{\text{post}} = \mathbf{v}_t^{\text{post}+} + \Delta\mathbf{v}_t^{\text{post}} \quad (27)$$

The amount of video data in $\Delta\mathcal{U}_t$, denoted by $\Phi(\mathbf{v}_t, \Delta\mathcal{U}_t)$, can be estimated according to buffer state \mathbf{v}_t^I and the rate-quality model introduced in Section II-C. Specifically, for each data unit in $\Delta\mathcal{U}_t$, we first determine the frame type according to v_t^I and then estimate the amount of data by the rate-quality model. The set $\Delta\mathcal{U}_t$ records the completely transmitted data units up to $(f_{n_t}, \ell_{n_t})^{\text{th}}$ data unit. However, data unit $(f_{n_t+1}, \ell_{n_t+1})$ is only partially received. Denoting the amount of data in unit $(f_{n_t+1}, \ell_{n_t+1})$ by $\tilde{\Phi}(\mathbf{v}_t^I, \Delta\mathcal{U}_t)$, the amount of received data is at least $\Phi(\mathbf{v}_t^I, \Delta\mathcal{U}_t)$ and at most $\Phi(\mathbf{v}_t^I, \Delta\mathcal{U}_t) + \tilde{\Phi}(\mathbf{v}_t^I, \Delta\mathcal{U}_t)$. Assuming the physical layer packet length is L^{PHY} , there is $N = \lceil \frac{x_t}{L^{\text{PHY}}} \rceil$ packet transmissions during a time slot. The number of successfully transmitted packets is at least $N_l = \lceil \frac{\Phi(\mathbf{v}_t^I, \Delta\mathcal{U}_t)}{L^{\text{PHY}}} \rceil$ and is less than $N_h = \lceil \frac{\Phi(\mathbf{v}_t^I, \Delta\mathcal{U}_t) + \tilde{\Phi}(\mathbf{v}_t^I, \Delta\mathcal{U}_t)}{L^{\text{PHY}}} \rceil$. As assumed in Section II-D, the channel state is constant over each slot. Thus, the packet losses are independent within each slot. The number of successful packet transmissions in a slot is distributed binomially. Hence, the state transition probability from $\mathbf{s}_t = (\mathbf{c}_t, \mathbf{v}_t)$ to $\mathbf{s}_{t+1} = (\mathbf{c}_{t+1}, \mathbf{v}_{t+1})$ is

$$\mathbb{P}_\mu(\mathbf{s}_{t+1}|\mathbf{s}_t) = \left[\sum_{n_t=N_l}^{N_h-1} \binom{N}{n_t} y_t^{N-n_t} (1-y_t)^{n_t} \right] \mathbb{P}(\mathbf{c}_{t+1}|\mathbf{c}_t), \quad (28)$$

where the first multiplicative term is the transition probability of the receiver buffer state from \mathbf{v}_t to \mathbf{v}_{t+1} and the second term is the transition probability of the channel state from \mathbf{c}_t to \mathbf{c}_{t+1} .

APPENDIX B SIMULATION SETTINGS

We employ the FSMC channel model proposed in [17] to model the dynamics of Rayleigh fading channels. The SNR at the receiver is partitioned into $|\mathcal{C}|$ regions using the algorithm proposed in [17]. Let Λ_i be the partition thresholds, where $\Lambda_0 = -\infty$ and $\Lambda_{|\mathcal{C}|} = \infty$. Let $\tilde{\Lambda}_k$ be the representative SNR in the k^{th} region. For Rayleigh fading channels, we have

$$\tilde{\Lambda}_k = \frac{\int_{\Lambda_{k-1}}^{\Lambda_k} \lambda p(\lambda) d\lambda}{\int_{\Lambda_{k-1}}^{\Lambda_k} p(\lambda) d\lambda}, \quad (29)$$

where $p(\lambda) = \frac{1}{\Lambda^{\text{avg}}} \exp(-\frac{\lambda}{\Lambda^{\text{avg}}})$ is the probability distribution function of the received instantaneous SNR of Rayleigh fading channels with average SNR Λ^{avg} . According to [17], the state transition probability \mathbf{P}^c is computed as

$$\mathbf{P}_{i,j}^c = \begin{cases} \frac{\mathcal{K}(\Lambda_j)\Delta T}{\pi_i} & \text{if } j = i + 1, \\ \frac{\mathcal{K}(\Lambda_i)\Delta T}{\pi_i} & \text{if } j = i - 1, \\ 1 - \frac{\mathcal{K}(\Lambda_j)\Delta T}{\pi_i} - \frac{\mathcal{K}(\Lambda_i)\Delta T}{\pi_i} & \text{if } j = i, \\ 0 & \text{otherwise,} \end{cases}$$

where $\pi_i = \int_{\Lambda_{i-1}}^{\Lambda_i} p(\lambda) d\lambda$. $\mathcal{K}(\Lambda_i) = \sqrt{\frac{2\pi\Lambda_i}{\Lambda^{\text{avg}}}} f^d \exp(-\frac{\Lambda_i}{\Lambda^{\text{avg}}})$ is the level crossing rate of threshold Λ_i where f^d is the Doppler frequency. The coherence time is estimated via $t_{\text{cor}} = 0.423/f^d$. In our simulations, we set $|\mathcal{C}| = 4$ and $\Lambda^{\text{avg}} = 10\text{dB}$.

We assume that BPSK, QPSK and 8PSK are used for modulation. The symbol error rate p_k^s in the k^{th} SNR region is $p_k^s = 2Q(\sqrt{2\tilde{\Lambda}_k} \sin \frac{\pi}{2M})$, where $M = 1, 2, 3$ for BPSK, QPSK and 8PSK, respectively. Each packet contains 2048 symbols. Thus, the packet length $L^{\text{PHY}} = 2048 \times M$, where $M = 1, 2$ and 3 for BPSK, QPSK and 8PSK, respectively. The transmission time for each packet is $\Delta t = 1.5\text{ms}$. The transmission data rate is given by $x_k = \frac{\Delta T}{\Delta t} L^{\text{PHY}}$. The packet error rate is given by $y_k = 1 - (1 - p_k^s)^{2048}$. The modulation scheme for k^{th} channel states is chosen such that the throughput $x_k(1 - y_k)$ is maximized.

REFERENCES

- [1] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1103–1120, Sept. 2007.
- [2] W. Li, "Overview of fine granularity scalability in MPEG-4 video standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 301–317, Mar. 2001.
- [3] M. Podolsky, S. McCanne, and M. Vetterli, "Soft ARQ for layered streaming media," *Technical Report, Computer Science Division, University of California, Berkeley*, vol. UCB/CSD-98-1024, 1998.
- [4] P. de Cuetos and K. W. Ross, "Optimal streaming of layered video: joint scheduling and error concealment," in *Proceedings of the International Conference on Multimedia*, 2003, pp. 55–64.
- [5] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. Multimedia*, vol. 8, no. 2, pp. 390–404, Apr. 2006.

- [6] J. Chakareski, P. A. Chou, and B. Aazhang, "Computing rate-distortion optimized policies for streaming media to wireless clients," in *Proceedings of Data Compression Conference*, 2002, pp. 53–62.
- [7] N. Changuel, N. Mastronarde, M. Van der Schaar, B. Sayadi, and M. Kieffer, "End-to-end stochastic scheduling of scalable video overtime-varying channels," in *Proceedings of the international conference on Multimedia*, 2010, pp. 731–734.
- [8] F. Fu and M. van der Schaar, "A new systematic framework for autonomous cross-layer optimization," *IEEE Trans. Veh. Technol.*, vol. 58, no. 4, pp. 1887–1903, May. 2009.
- [9] Y. Zhang, F. Fu, and M. van der Schaar, "On-line learning and optimization for wireless video transmission," *IEEE Trans. Signal Process.*, vol. 58, no. 6, pp. 3108–3124, Jun. 2010.
- [10] F. Fu and M. van der Schaar, "A systematic framework for dynamically optimizing multi-user wireless video transmission," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 3, pp. 308–320, Apr. 2010.
- [11] —, "Structural solutions for dynamic scheduling in wireless multimedia transmission," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 5, pp. 727–739, may 2012.
- [12] C. Chen, R. W. Heath, A. C. Bovik, and G. de Veciana, "Adaptive policies for real-time video transmission: a Markov decision process framework," in *18th IEEE International Conference on Image Processing*, Sept. 2011.
- [13] D. Love, J. Heath, R.W., W. Santipach, and M. Honig, "What is the value of limited feedback for MIMO channels?" *IEEE Communications Magazine*, vol. 42, no. 10, pp. 54 – 59, Oct. 2004.
- [14] D. Love, R. Heath, V. Lau, D. Gesbert, B. Rao, and M. Andrews, "An overview of limited feedback in wireless communication systems," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 8, pp. 1341–1365, Oct. 2008.
- [15] 3GPP, "Further advancements for E-UTRA physical layer aspects," 3rd Generation Partnership Project (3GPP), TR 36.814, 2010.
- [16] N. Changuel, B. Sayadi, and M. Kieffer, "Online learning for qoe-based video streaming to mobile receivers," in *Proceedings of the Global Communications Conference*, 2012.
- [17] Q. Zhang and S. A. Kassam, "Finite-state Markov model for Rayleigh fading channels," *IEEE Trans. Commun.*, vol. 47, no. 11, pp. 1688–1692, Nov. 1999.
- [18] H. S. Wang and P.-C. Chang, "On verifying the first-order Markovian assumption for a Rayleigh fading channel model," *IEEE Trans. Veh. Technol.*, vol. 45, no. 2, pp. 353–357, May. 1996.
- [19] H.-P. Lin and M.-J. Tseng, "Two-layer multistate Markov model for modeling a 1.8 GHz narrow-band wireless propagation channel in urban Taipei city," *IEEE Trans. Veh. Technol.*, vol. 54, no. 2, pp. 435–446, Mar. 2005.
- [20] T. Su, H. Ling, and W. J. Vogel, "Markov modeling of slow fading in wireless mobile channels at 1.9 GHz," *IEEE Trans. Antennas Propag.*, vol. 46, no. 6, pp. 947–948, Jun. 1998.
- [21] J. Sun, W. Gao, D. Zhao, and W. Li, "On rate-distortion modeling and extraction of H.264/SVC fine-granular scalable video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 3, pp. 323–336, Mar. 2009.
- [22] Technical Report [On line]. Available: <https://webSPACE.utexas.edu/cc39488/pdf/report.pdf>.
- [23] D. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd ed. Athena Scientific, 2005, vol. 2.
- [24] Test sequences [On line]. Available: <http://trace.eas.asu.edu/yuv/>.
- [25] J. Reichel, S. Schwarz, and M. Wien, "Joint scalable video model 11 (JSVM 11)," *Joint Video Team, Doc. JVT-X202*, Jul. 2007.
- [26] K. Seshadrinathan, R. Soundararajan, A. C. Bovik, and L. K. Cormack, "Study of subjective and objective quality assessment of video," *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1427–1441, Jun. 2010.
- [27] LIVE image quality assessment database [On line]. Available: <http://live.ece.utexas.edu/research/quality/subjective.htm>.
- [28] S. M. Pandit and S. M. Wu, *Time-Series and System Analysis with Application*. J. Wiley & Sons, 1983.
- [29] M. Neuts, *Matrix-Geometric Solutions in Stochastic Models: An Algorithm Approach*. The Johns Hopkins University Press, 1981.