Scheduling Algorithms for Broadcasting Media with Multiple Distortion Measures

Carri W. Chan, *Student Member, IEEE,* Nick Bambos, *Member, IEEE,* Susie Wee, *Member, IEEE,* and John Apostolopoulos, *Fellow, IEEE*

***Abstract*—The growing popularity of multimedia streaming applications brings a growth in diversity of media clients (laptops, PDAs, cellphones). Effectively serving this heterogeneous group of users is highly desirable. Scalable media codecs such as H.264/MPEG-4 SVC help make this adaptation possible. To account for the various capabilities and requests of each user, such as varying spatial or temporal resolutions, Multiple Distortion Measures (MDM) are considered [1]. Rather than consider a homogeneity in users, the MDM framework considers multiple different distortion values for each media packet for each user type. We consider the scenario of simultaneously broadcasting a video stream to multiple users over wireless links. The objective is to design a scheduling algorithm which achieves the highest aggregate Quality-of-Service, measured by distortion and delay, over all different user types. We cast the problem as a stochastic shortest path problem and use Dynamic Programming to find the optimal policy. For statistically static channels, the optimal policy is shown to be of threshold type. For time-varying channels, a quasi-static policy is introduced. Experimental results show that our policy reduces distortion by up to a factor of 2 over conventional approaches which do not consider MDM.**

***Index Terms*—Multiple distortion measures, wireless scheduling, dynamic programming, multimedia streaming.**

I. Introduction

In current media streaming systems, media is often simultaneously streamed to users with various display capabilities over different network conditions. Scalable media, such as JPEG2000 [2] for images and H.264/MPEG-4 SVC [3] for video, allows content providers to transmit and adapt media for different types of receivers by simply discarding packets. Rate-Distortion optimized scheduling has attracted a significant amount of attention (see [4] and related references) even in the case of scalable media [5]. By prioritizing the different packets in an intelligent manner, the needs of all clients can be addressed to supply the best media content possible.

Scalable media can be quickly and easily adapted to different user types. How to evaluate performance of a scalable media system is examined in [1], [6]. To account for the varying display capabilities of each user, we consider media packets with Multiple Distortion Measures [1]. Multiple Distortion Measures capture the varying display capabilities of each user. Suppose we simultaneously stream an image or video to users with large displays, such as laptop monitors, as well as to users with small displays, such as cellphone screens. The amount of distortion incurred with the loss of a packet will depend on the type of user. For instance, the loss of a packet which contains high frequency information, such as texture in a sweater or grass in a field may result in high distortion incurred for a user with a large display. However, the loss of this packet may be negligible to a different user because the high frequency information cannot be displayed on the smaller display.

By considering the viewing capabilities of each user when making a scheduling decision, up to 4dB gains can be achieved in Multiple Distortion Measure aware embedded schedules [1]. Embedded schedules are schedules which incrementally add packets so that all packets in the schedule at rate R_1 are also included in the schedule at rate $R_2 > R_1$. Embedded schedules are useful because they make rate reduction possible by simply truncating the bitstream. It is most surprising that distortion values associated with each packet vary greatly depending on the type of consumer. A packet that is very important to a high resolution user can be virtually useless to a low resolution user. The disparities between packet values of each user makes scheduling packets with Multiple Distortion Measures an interesting problem.

In this paper, we examine a different streaming media scenario. We consider the case of a single wireless transmitter broadcasting the same content to multiple receivers. Each receiver may have a different distortion measure due to his display capabilities. In each time slot, the server must determine whether to transmit the Head of Line (HOL) packet or drop it. Dropping the packet will lead to incurred distortion at each user who has not yet received the packet and the amount of distortion will depend on the user type. Transmitting the packet may only benefit a subset of the users, depending on who receives it successfully, and it blocks transmissions of packets waiting in the queue. Suppose the HOL packet is transmitted, but only a fraction of the users receive it. A control dilemma arises: should we retransmit the packet, only benefiting those users who have not yet received the packet, or should we drop it, hurting the same users who have not received it, and send a different packet that potentially benefits all users? We examine this scheduling decision.

A significant body of work in the Computer Science com-

Carri W. Chan is with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305 USA. email: cwchan@stanford.edu

Nick Bambos is with the Department of Electrical Engineering and Department of Management Science & Engineering, Stanford University, Stanford, CA 94305 USA. email: bambos@stanford.edu

Susie Wee is with the Experience Software Business, Hewlett-Packard, Cupertino, CA 95014 USA. email: susie.wee@hp.com

John Apostolopoulos is with the Multimedia Communications and Networking Lab, Hewlett-Packard Laboratories, Palo Alto, CA 94304 USA. email: john_apostolopoulos@hp.com

Manuscript received June 24, 2008; revised November 29, 2008; accepted April 29, 2009

munity has examined broadcast scheduling [7], [8], and [9]. This work focuses on generating near-optimal approximation algorithms for throughput maximization in wireless broadcast scenarios. One significant distinction between these works and ours is that we consider stochastic channels where transmissions are not guaranteed to be successful. Furthermore, we account for differing prioritization of each packet to each user, whereas these works consider the case where there is no prioritization of packets or users.

Our model is closely related to existing models for certain broadcast scheduling applications; as an example [10] considers a multimedia broadcast problem where many users wish to simultaneously consume the same media stream. As is the case with many multimedia applications, latency is an important Quality of Service aspect to address. Low latency broadcast scheduling has been examined in [11] and [12] among others. Our work differs from these in that we consider the unique properties of media communications where packet/frame importance varies between frames and even across users. This work is similar in flavor to [13] and [14]. In [13], backlog is used as a measure of the time until the transmission session completes and the transmission buffer is emptied. However, backlog is a very coarse measure for delay because two systems with the same backlog size, but with different media importance, may result in very different transmission times and delays. For instance, a backlog of 1 packet can have zero delay if the distortion of that packet is 0 and it is immediately dropped. On the other hand, a backlog of 1 can have very high delay if the distortion of that packet is extremely high and requires complete reception by all users, possibly requiring many retransmissions, before concluding the transmission session. As such, we propose to measure delay by the average number of retransmissions performed per packet. In the case of periodic video traffic, this corresponds to the average delay of each frame. In contrast to [14], we provide a more in depth look into the optimal scheduling policy and further study the tradeoff between the distortion and delay. In this paper, we introduce a knob to tune for a precise target frame rate. We include a new simulation study of these delay results as well as extend upon the previous simulation scenarios.

The rest of the paper is as follows. In Section II we formally introduce the problem formulation for the broadcasting problem we study. In Section III we present an offline algorithm to generate the optimal scheduling policy for a general class of networks. We cast the problem as a stochastic shortest path problem and use Dynamic Programming to find the optimal policy which is stored in a lookup table the transmitter refers to at each scheduling time slot. In Section IV we examine the optimal policy in the case of i.i.d Bernoulli packet losses. We show that the optimal policy is of threshold type and can be calculated online. In Section V we compare our Multiple Distortion Measure aware policy to standard policies which only consider a single distortion measure. Through simulation results for actual H.264/MPEG-4 SVC encoded videos, we show that up to 3dB gains can be achieved by considering Multiple Distortion Measures when making scheduling decisions. Finally, we conclude in Section VI.

II. Problem Formulation and Setup

The problem we study is how to schedule media packets given the diverse needs of individual users, which can be quantified via Multiple Distortion Measures. We consider the wireless broadcast scenario depicted in Fig. 1. A pre-encoded video stream is stored in a transmission buffer. In each time slot, t, the transmitter broadcasts a packet over a single channel frequency or CDMA code. All U users are tuned to the same frequency/code and can potentially receive the broadcasted packet. However, due to varying path loss and fading to each user, the channel quality differs at each receiver. Therefore, the probability of successful reception may be different to each user $u \in \mathcal{U} = \{1, 2, \ldots, U\}$. At the beginning of each time slot, the transmitter must make a decision about the Head of Line (HOL) packet: transmit or drop. That is, packets are transmitted according to a FIFO discipline and in each time slot, the dilemma is whether to serve the HOL packet by retransmitting it or to service the following packet by dropping it. It is possible to consider other service disciplines; however, we will assume a First-In-First-Out discipline in our discussion in order to focus on the distortion versus delay tradeoff. Transmitting or retransmitting the packet makes it possible for users who have not yet received the HOL packet to receive it. However, retransmissions will induce delay for all other packets waiting in the transmission queue. By dropping the HOL packet, the next packet in the queue can be transmitted and delay can be reduced. However, dropping a packet comes at a cost. All users that have not received the packet that is dropped will suffer an increase in media distortion. There is clearly a tradeoff between incurring distortion versus delay. It is this tradeoff which we will examine throughout this paper.

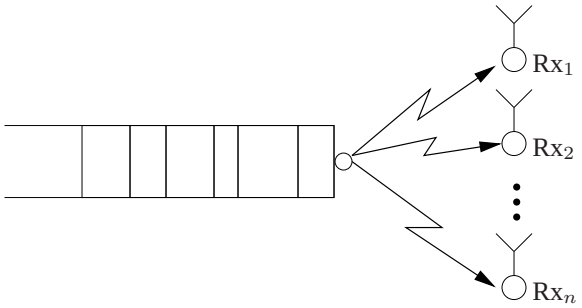


Fig. 1. Broadcasting media packets to multiple users.

A. Wireless Channels

Wireless channels are generally modeled in one of two ways–either by a time-varying bit-rate or by a time-varying probability of successful transmission. We take the second approach. We assume that in each time slot, t, the quality of the wireless channel to each user, u, is characterized by the probability of successful transmission, $s_u(t)$. $s_u(t)$ can be an arbitrary stochastic process. There has been a substantial amount of work to construct reliable *dynamic* models to describe this variability (for instance, see [15] and associated work). Consequently, it is common to model the success probabilities via a Finite-State Markov Chain with

states $c_u \in \mathcal{C} = \{1, 2, \ldots, C\}$ and transition probabilities from $\mathbf{c} \in \mathcal{C}^U$ to $\bar{\mathbf{c}} \in \mathcal{C}^U$ as $q_{\mathbf{c}\bar{\mathbf{c}}}$. If $c_u(t)$ is the state of channel u in time slot t then the probability of successful transmission to user u is $s_u(t) = s_u(c_u(t))$. We assume, after each transmission, acknowledgements are transmitted back to the server and this feedback is reliable. Therefore, the server knows which users have and have not yet received the HOL packet.

B. Distortion Costs: Multiple Distortion Measures

The objective of the transmitter is to minimize the total distortion of all receivers. Distortion can come in two forms: media (spatial) distortion and play out delay (temporal distortion). Hence forth, we will refer to distortion as the media distortion which is a measure of the fidelity of the displayed image or video and play out delay as the disruption due to late packets. We will assume that distortion is additive across multiple dropped frames as in [16] and related works. While there are more complex distortion models (see [17], [18]), the additive model is a commonly used model which is relatively simple and accurate, while also being tractable and we believe practical. In Section V we show evidence of the performance of algorithms which use this model. It is also possible to model simple linear dependencies, which can be depicted as a tree structure, as will be seen in Section V. These types of dependencies often suffice in depicting accurate distortion models for many video codecs. In conventional media systems, each packet is assumed to have a single distortion measure. The distortion value associated with each packet is the increase in incurred distortion which would occur with the loss of that packet. Equivalently, the distortion value is the reduction in distortion with the inclusion of that packet.

Typically, distortion is measured as the mean-squared error compared to the original high resolution, high frame rate video sequence. However, this ignores the case of different user types who measure distortion in various ways–hence, the need for Multiple Distortion Measures [1]. The latter arise due to the increase in diversity of multimedia consumers. Intuitively, a user with a low resolution display will have very different requirements than a user with a high resolution display.

We assume M packets are to be transmitted in some predetermined order to each receiver. Each packet may correspond to part or the whole of a single video frame. Packet $m \in \{1, 2, \ldots, M\}$ is the mth packet to be transmitted. We denote by $d_u^m \geq 0$ the amount of distortion incurred by user u if he does not receive packet m. The distortion value is precomputed and stored in the header of packet m as in [19] and is available information to the transmitter. In conventional systems, $d_u^m = d_{u'}^m$ for all $u, u' \in \mathcal{U}$, but because of Multiple Distortion Measures $d_u^m \neq d_{u'}^m$ for some $u \neq u'$. Note that a packet is only useful to a receiver the first time it is received, i.e., distortion is not reduced further with the second reception of the same packet. Suppose user u does not receive packets in the set $\mathcal{K}$. Then his total incurred distortion is: $d_u = d_u^0 + \sum_{k \in \mathcal{K}} d_u^k$ where d_u^0 is the base distortion due to lossy compression assuming all packets are received. Then the media quality of user u can be measured by the typical metric of PSNR, where $\text{PSNR}_u = 10 \log_{10}(255^2/d_u)$.

Conventional media systems today do not account for Multiple Distortion Measures, which leads to the question: Where do MDMs come from and how do we calculate them? MDMs introduce multiple benchmarks against which to evaluate performance in order to enable more applicable performance metrics. A block diagram of how to generate Multiple Distortion Measures can be seen in Fig. 2 for a more detailed description see [1]. We define by $\mathcal{T}_u(X)$ a *transformation* operator of media content, X, for user type u. A transformation converts media content X into a modified, benchmark version which user type u will view and consume the content. For example, this transformation could be spatial downsampling to convert our original benchmark image, X, into a low resolution benchmark image, $\mathcal{T}_u(X)$, if user type u wishes to view the image on a low resolution display. The transformation could also be a temporal downsampling or framerate conversion operation, such as frame dropping in the simplest case, to reduce the frame rate for video. Therefore, $\mathcal{T}_u(X)$ is the reference media against which performance evaluation is measured for user u. Define $\mathcal{T}_I$ as the identity transformation such that $\mathcal{T}_I(X) = X$. There will be multiple transformation operators–one corresponding to each user type.

These multiple benchmark images (one for each transformation) are now used to calculate distortion values of reconstructed images–hence, Multiple Distortion Measures (MDM). Let's define $D_{\mathcal{T}_u}(\hat{X})$ as the distortion of reconstructed image $\hat{X}$ compared to the benchmark image $\mathcal{T}_u(X)$. Note that this is a function of X and $\hat{X}$ as well as the transform $\mathcal{T}_u$, $D_{\mathcal{T}_u}(\hat{X}) = f(\hat{X}, \mathcal{T}_u(X))$. The conventional approach in media transmission systems is to assume the receiver wishes to consume the data in the original format of encoding, so performance evaluation is done by calculating distortion of the reconstructed image, $\hat{X}$, compared to the original, i.e. $D = D_{\mathcal{T}_I}(\hat{X}) = f(\hat{X}, X)$ as in Fig. 2. However, this is still done even if the image, $\hat{Y}$, is displayed and reconstructed differently such as on a cellphone screen. It is difficult to make a comparison between $\hat{Y}$ and X since the reconstructed image and benchmark image have different resolutions. This is sometimes bypassed by up-sampling the low resolution image. Instead of calculating distortion of $\hat{Y}$ compared to the original benchmark X, we propose to calculate distortion compared to the transformed benchmark image, $\mathcal{T}_u(X)$. This provides a more applicable performance evaluation. In the scenario of Fig. 2 with a desktop and a cellphone user, there would be two distortion measures for each packet: one for the high resolution user, $D_{\mathcal{T}_I}(\cdot)$, and one for the low resolution user, $D_{\mathcal{T}_L}(\cdot)$.

C. Delay Costs

Another form of distortion is the disruption caused by delayed packets. We assume that the video sequence has a target frame rate. This corresponds to a target period $R \geq 1$ at which frames will be played out. We assume that the period is normalized to the length of a time slot so that users wish to play a new frame every R time slots. Suppose packet m is retransmitted τ times. If $\tau < R$, more retransmissions are possible while still allowing the receivers to play out the video sequence at the target frame rate. However, if $\tau > R$, the

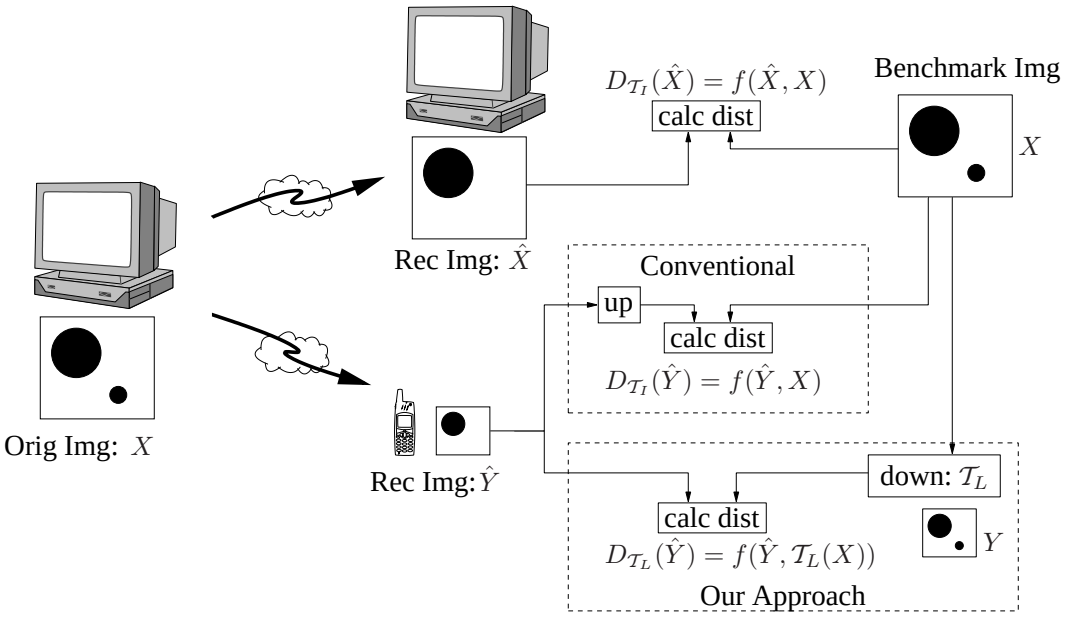


Fig. 2. Diagram of Multiple Distortion Measures. The conventional approach assumes all users evaluate performance compared to the original benchmark image, $\mathcal{T}_I(X) = X$. This is often done by upsampling a low resolution version of the received image, $\hat{Y}$, to the original image resolution. Multiple Distortion Measures calculate distortion compared to a *transformed* benchmark image, $\mathcal{T}_L(X)$, which accurately captures the display capabilities of the user in question. In this case, the low resolution benchmark image $Y = \mathcal{T}_L(X)$ is a downsampled version of the original benchmark image.

receivers must deviate from the target frame rate while waiting for the next frame and incur play out delay. We measure this type of disruption by a delay cost, $D(\tau)$. $D(\tau)$ is a non-decreasing function of τ, as more transmissions translates to more delay which should be penalized. However, we make no other assumption about its functional form and leave that under the control of the system designer. One possible function is a step function where $D(\tau) = 0$ for $\tau \leq R$ and $D(\tau) = 1$ for $\tau > R$. This would penalize uniformly for each retransmission that leads to deviation from the target frame rate.

If each packet had a single distortion measure, each user would incur the same amount of distortion with the loss of the same packet. By accounting for multiple distortion measures, the transmitter can intelligently determine whether to (re)transmit the HOL packet or drop it and transmit the next one in order to collectively maximize the viewing quality of all users. Suppose a packet has been received by user A, but not user B. Suppose also that this packet is very important for user A, but not at all for user B. In standard systems, this packet would be identically prioritized for both users and a scheduling algorithm may require that both users receive the packet. However, if the transmitter is aware of the Multiple Distortion Measures, he can drop the HOL packet with little loss of performance and spend more resources on future transmissions. It is this scheduling dilemma which we wish to examine.

The scheduling problem we will address is precisely formulated as follows. Given the Multiple Distortion Measures, the Head-of-Line packet reception information, the channel success probabilities, and channel dynamics, the transmitter must determine whether to transmit the HOL packet, or drop it and transmit the next packet. Our goal is to find an efficient algorithm to resolve this control dilemma in a manner that maximizes the viewing quality for all receivers.

III. General Optimal Offline Algorithm

In this section, we model the scheduling dilemma of transmitting or dropping the Head-of-line (HOL) packet as a stochastic shortest path problem. We use Dynamic Programming to find the optimal solution and store the scheduling protocol in a lookup table [20]. In each time slot, the transmitter accesses the system state based on packet reception information, channel statistics, and packet distortion values, then references this table and determines whether it is optimal to transmit or drop the HOL packet.

All users need/want to experience low play out delay and high media quality. Unfortunately, these are competing objectives and there is a tradeoff between delay and media distortion. Requiring high media quality can result in high delays, whereas low delay can result in low media quality. We weigh the delay cost $D(\tau)$ by $\alpha \geq 0$, which varies the importance of distortion cost versus delay costs along this tradeoff. If $\alpha = 0$, customers are only concerned with distortion and in order to ensure successful reception of all packets by all users, the number of retransmissions by the transmitter, and subsequently the delay, can be arbitrarily high. As $\alpha \to \infty$, users are only concerned with costs associated with deviating from the target frame rate, and many packets may be dropped. Implicitly, α is the tradeoff factor between distortion costs and play out delay.

We can now cast the control problem as a stochastic shortest path problem. The state to be tracked in each time slot is

$(\mathbf{v}, \tau, m, \mathbf{c})$. $\mathbf{v}$ is a vector of indicator variables, v_u, where:

$$v_u = \begin{cases} 1, & \text{if user } u \text{ has not yet received packet } m; \\ 0, & \text{if user } u \text{ has already received packet } m. \end{cases} \quad (1)$$

τ is the number of previous transmissions of the HOL packet, m, and $\mathbf{c}$ is the vector of the current channel states so that $\mathbf{s}(\mathbf{c}) = \{s_1(c_1), s_2(c_2), \ldots, s_U(c_U)\}$ is a vector of the success probabilities to each user $u \in \mathcal{U}$. Define $x_u(c_u)$ as a Bernoulli random variable which represents the successful transmission to user u given channel state c_u. Therefore $x_u(c_u) \sim$ Bernoulli$(s_u(c_u))$, ($x_u(c_u) = 1$ if the transmission is successful, and 0 otherwise). For the rest of the discussion, we suppress the dependence on c_u of $x_u = x_u(c_u)$ and understand that the success of a transmission is dependent on the channel state. We can then define $p(\mathbf{v}, \hat{\mathbf{v}}, \mathbf{c}) = p(\mathbf{v}, \hat{\mathbf{v}})$ as the probability of transitioning from $\mathbf{v} \rightarrow \hat{\mathbf{v}}$ given the channel state $\mathbf{c}$, i.e. $\hat{v}_u = v_u(1 - x_u)$. Let $J^*(\mathbf{v}, \tau, m, \mathbf{c})$ be the minimum expected cost to go associated with initial state $(\mathbf{v}, \tau, m, \mathbf{c})$. Then $J^*(\mathbf{v}, \tau, m, \mathbf{c})$ satisfies Bellman's equation which relates the optimal cost in the current state to the expected future costs:

$$\begin{aligned} &J^*(\mathbf{v}, \tau, m, \mathbf{c}) \\ &= \min\{\alpha D(\tau) + E_{(\hat{\mathbf{v}}, \bar{\mathbf{c}}|\mathbf{v}, \mathbf{c})}[J^*(\hat{\mathbf{v}}, \tau + 1, m, \bar{\mathbf{c}})], \\ &\qquad \sum_u v_u d_u^m + E_{(\bar{\mathbf{v}}, \bar{\mathbf{c}}|\mathbf{1}, \mathbf{c})}[J^*(\bar{\mathbf{v}}, 1, m + 1, \bar{\mathbf{c}})]\} \\ &= \min\{\alpha D(\tau) + \sum_{q_{\mathbf{c}\bar{\mathbf{c}}}} \sum_{p(\mathbf{v}, \hat{\mathbf{v}})} q_{\mathbf{c}\bar{\mathbf{c}}} p(\mathbf{v}, \hat{\mathbf{v}}) J^*(\hat{\mathbf{v}}, \tau + 1, m, \bar{\mathbf{c}}), \\ &\qquad \sum_u v_u d_u^m + \sum_{q_{\mathbf{c}\bar{\mathbf{c}}}} \sum_{p(\mathbf{1}, \bar{\mathbf{v}})} q_{\mathbf{c}\bar{\mathbf{c}}} p(\mathbf{1}, \bar{\mathbf{v}}) J^*(\bar{\mathbf{v}}, 1, m + 1, \bar{\mathbf{c}})\} \end{aligned} \quad (2)$$

The first term in the minimization corresponds to the decision to transmit the HOL packet. An instantaneous cost, $\alpha D(\tau)$, due to (re)transmission is incurred. The expected future cost is given by $J^*(\cdot)$ with the system state updated. $\mathbf{v} \rightarrow \hat{\mathbf{v}}$ is updated based on the (un)successful transmissions to each user. The decision to transmit the HOL packet results in the number of transmission attempts to increment by one. Finally, the channel state is updated $\mathbf{c} \rightarrow \bar{\mathbf{c}}$ (with probability $q_{\mathbf{c}\bar{\mathbf{c}}}$).

The second term in the minimization corresponds to dropping the HOL packet and transmitting the next packet in the queue, $m + 1$. By dropping the packet, distortion is incurred by each user who has not yet received packet m, for all u such that $v_u = 1$. At the beginning of this time slot, before the transmission of packet $m + 1$ occurs, $\mathbf{v} = \mathbf{1}$, because no users have yet received packet $m + 1$. Now $\mathbf{1} \rightarrow \bar{\mathbf{v}}$ and $\mathbf{c} \rightarrow \bar{\mathbf{c}}$ in a similar manner as before. Notice that each user's distortion is weighed uniformly. However, as we will later see, this means that sometimes one user's Quality-of-Service (QoS) is sacrificed for the global good. It is possible to avoid this and introduce "fairness" to the system by weighing each user's distortion by β_u so that the distortion contribution for each user for the loss of packet m is $\beta_u d_u^m$. This does not significantly alter our discussion and we omit the details for the sake of space.

The optimal decision is to select the action (transmit/drop) in order to minimize the total costs. The tradeoff here is to either transmit the HOL packet and pay an immediate delay cost while potentially reducing the distortion costs versus dropping the HOL packet and pay the immediate distortion costs while avoiding the delay cost from retransmitting the HOL packet.

There are M total packets to be transmitted. Once all the packets are transmitted, no more distortion or delay can be incurred; hence, our terminal state cost is $J^*(\mathbf{v}, \tau, M+1, \mathbf{c}) = 0$. Solving the DP recursion in (2) will result in the minimal cost and the transmission policy for each HOL packet corresponds to the action which minimizes the cost in each state. The solution can be found using the *value iteration* method [20].

Proposition 1: There exists a stationary optimal control solution to (2) which is obtainable via value iteration.

Proof: The Bellman's recursion terminates when there are no more packets in the transmission queue left to transmit. i.e., $m = M + 1$. There is no cost for being in this state and the optimal policy will never leave this state once it reaches it. Any policy which does not empty the buffer in finite time will incur infinite cost due to the retransmission costs, $D(\tau)$. There exists a policy which will empty the transmission buffer and cause the Bellman's recursion to terminate in finite time. (i.e., we can drop *all* HOL packets and terminate in M time slots.) Therefore, there should exist a stationary optimal policy which is obtainable via value iteration [20]. ■

Let M be the number of packets to transmit, U be the number of users to transmit to, C be the number of channel states per user, and R_{max} be the maximum allowable retransmissions. The state space for this problem is:

$$\mathcal{S} = \Big[(\mathbf{v}, \tau, m, \mathbf{c}) : \mathbf{v} \in \{0, 1\}^U, \tau \in \{0, 1, \ldots, R_{max}\}, \\ m \in \{1, \ldots, M + 1\}, \mathbf{c} \in \{0, 1, \ldots, C\}^U\Big]$$

The size of the state space for this problem is exponential in the size of some problem parameters–$|\mathcal{S}| = (M + 1)2^U C^U R_{max}$. Without a restriction on the number of retransmissions, the state space is countably infinite. Ignoring the vast computation required to calculate the optimal policy, the memory requirements, itself, to store a lookup table of the optimal policy would be intractable. Clearly, the size of the state space precludes the use of this approach in real world systems.

IV. A Threshold Policy for Statistically Static Channels

Due to the potential intractability of the general DP formulation in the previous section, we leverage a key case in order to develop practical algorithms for real world implementation. We model the wireless channel to each user by i.i.d. packet losses, equivalently, by a single state Markov Chain. The success probabilities differ for each user, but are fixed across subsequent time slots. This assumption is justified in the case of slowly varying channels. In this case, the optimal control is of threshold type and can be computed online. Based on this key case, which yields tractable solutions, we later investigate how to apply these results to develop algorithms

which mimic the optimal policy in the more general case of dynamic channels. We begin by appropriately modifying the DP recursion in (2).

Because the success probabilities are fixed, we no longer have to track the state of the channel. We now denote by s_u the probability of successful transmission to user u in *all* time slots. The DP recursion in (2) can be rewritten as:

$$\begin{aligned} &J^*(\mathbf{v},\tau,m) \\ &\quad = \min\{\alpha D(\tau) + E_{(\hat{\mathbf{v}}|\mathbf{v})}[J^*(\hat{\mathbf{v}},\tau+1,m)], \\ &\qquad \sum_u v_u d_u^m + E_{(\bar{\mathbf{v}}|\mathbf{1})}[J^*(\bar{\mathbf{v}},1,m+1)]\} \\ &\quad = \min\{\alpha D(\tau) + \sum_{p(\mathbf{v},\hat{\mathbf{v}})} p(\mathbf{v},\hat{\mathbf{v}}) J^*(\hat{\mathbf{v}},\tau+1,m), \\ &\qquad \sum_u v_u d_u^m + \sum_{p(\mathbf{1},\bar{\mathbf{v}})} p(\mathbf{1},\bar{\mathbf{v}}) J^*(\bar{\mathbf{v}},1,m+1)\} \end{aligned} \tag{3}$$

Again, the first term in the minimization corresponds to the cost of transmitting the HOL packet and the second term corresponds to the cost of dropping the HOL packet and transmitting the next packet.

Suppose a packet is retransmitted, but no new users successfully receive it, so that $\hat{\mathbf{v}} = \mathbf{v}$. Intuitively, the cost in state $(\mathbf{v},\tau+1,m)$ should result in larger costs than $(\mathbf{v},\tau,m)$ as the delay is increased, but the distortion is not decreased. The following proposition establishes this claim.

Proposition 2 (Monotonicity): $J^*(\mathbf{v},\tau,m)$ is increasing in τ, for each fixed $\mathbf{v}$ and m.

Proof: Let's consider 2 systems, one starting in state $(\mathbf{v},\tau,m)$ and the other starting in state $(\mathbf{v},\tau+1,m)$. Consider a coupling of the systems so that they see that same sample paths or realizations of packet transmissions, i.e. the same successful and unsuccessful transmissions. Let $\pi^*_{\tau+1}$ denote the optimal policy used by the system starting with $\tau+1$. Let $J^{\tau+1}(\mathbf{v},\tau,m)$ denote the cost of the system starting in state $(\mathbf{v},\tau,m)$ which tracks the evolution of the $(\mathbf{v},\tau+1,m)$ system and uses policy $\tilde{\pi} = \pi^*_{\tau+1}$ instead of the optimal policy, π^*_τ for this state. More formally, let $S(\mathbf{v})$ be the random evolution of $\mathbf{v}$ given a transmission of the HOL packet. Then,

$$\tilde{\pi}(\mathbf{v},\tau,m) = \pi^*(\mathbf{v},\tau+1,m)$$

It is easy to see that if $\tilde{\pi}(\mathbf{v},\tau,m) = \texttt{Transmit}$, then the $(\mathbf{v},\tau,m)$ system evolves to $(S(\mathbf{v}),\tau+1,m)$ and the $(\mathbf{v},\tau+1,m)$ system evolves to $(S(\mathbf{v}),\tau+2,m)$. Then,

$$\tilde{\pi}(S(\mathbf{v}),\tau+1,m) = \pi^*(S(\mathbf{v}),\tau+2,m)$$

Alternatively, if $\tilde{\pi}(\mathbf{v},\tau,m) = \texttt{Drop}$, then both the $(\mathbf{v},\tau,m)$ and $(\mathbf{v},\tau+1,m)$ systems evolve to $(S(\mathbf{1}),1,m+1)$. And for all subsequent time slots $\tilde{\pi} = \pi^*$.

Therefore, in each time slot, each system employs the same transmit or drop decision given by $\pi^*_{\tau+1}$. Once the decision is to drop the HOL packet, both systems will be in the same state, $(S(\mathbf{v}),1,m+1)$ and, hence, incur identical costs in all future states due to the coupling. Prior to dropping, the system starting in state $(\mathbf{v},\tau+1,m)$ incurs larger costs in each time slot because $D(\tau) \le D(\tau+1)$. Let $J^{\tilde{\pi}}(\mathbf{v},\tau,m)$ denote the expected cost of policy $\tilde{\pi}$ starting in state $(\mathbf{v},\tau,m)$. Let Z be the random stopping time such that $\tilde{\pi}(\mathbf{v}',\tau+Z,m) = \texttt{Drop}$ Then,

$$\begin{aligned} J^*(\mathbf{v},\tau,m) &\le J^{\tilde{\pi}}(\mathbf{v},\tau,m) \\ &= E_Z[\sum_{t=0}^{Z} \alpha D(\tau+t) + \sum_u v'_u d_u^m] \\ &\quad + E[J^{\tilde{\pi}}(S(\mathbf{1}),1,m+1)] \\ &= E_Z[\sum_{t=0}^{Z} \alpha D(\tau+t) + \sum_u v'_u d_u^m] \\ &\quad + E[J^*(S(\mathbf{1}),1,m+1)] \\ &\le E_Z[\sum_{t=0}^{Z} \alpha D(\tau+1+t) + \sum_u v'_u d_u^m] \\ &\quad + E[J^*(S(\mathbf{1}),1,m+1)] \\ &= J^*(\mathbf{v},\tau+1,m) \end{aligned} \tag{4}$$

where the first inequality follows from the optimality of $J^*(\cdot)$. The second equality follows from the definition of the $\tilde{\pi}$ policy. The second inequality follows from the non-decreasing property of $D(\tau)$. And the last equality follows from the definition of the optimal policy. This concludes the proof. ■

Define $\tilde{d}_u^m$ as the effective distortion of packet m to user u given the current system state. This is the distortion incurred by user u if packet m is dropped, given the current state. $\tilde{d}_u^m = 0$ if the packet has already been received, i.e., $v_u = 0$, and $\tilde{d}_u^m = d_u^m$ if the packet has not yet been received, i.e., $v_u = 1$. We can now prove the optimal transmission/drop policy is of threshold type. This policy is analogous to the one in [13] which, instead of comparing transmission versus distortion costs, compares backlog versus distortion costs.

Theorem 1 (Optimal Threshold Policy): The optimal policy for independent i.i.d. packet losses is of threshold type. That is, for each fixed $\mathbf{v}$ and m, there exists some $\tau_{max} \in [1,2,\ldots,\infty)$ such that (re)transmitting the Head of Line packet if $\tau \le \tau_{max}$ and dropping it otherwise is an optimal policy; τ_{max} depends on $\mathbf{v}$ and m.

Proof: If the optimal policy is to transmit in state $(\mathbf{v},\tau+1,m)$, then it is also optimal to transmit it in state $(\mathbf{v},\tau,m)$. This is because, in state $(\mathbf{v},\tau,m)$ the cost to transmit is less than the dropping cost; indeed, note that

$$\begin{aligned} \alpha D(\tau) &+ E_{(\hat{\mathbf{v}}|\mathbf{v})}[J^*(\hat{\mathbf{v}},\tau+1,m)] \\ &\le \alpha D(\tau) + E_{(\hat{\mathbf{v}}|\mathbf{v})}[J^*(\hat{\mathbf{v}},\tau+2,m)] \\ &\le \alpha D(\tau+1) + E_{(\hat{\mathbf{v}}|\mathbf{v})}[J^*(\hat{\mathbf{v}},\tau+2,m)] \\ &\le \sum_u v_u d_u^m + E_{(\hat{\mathbf{v}}|\mathbf{1})}[J^*(\hat{\mathbf{v}},1,m+1)] \end{aligned} \tag{5}$$

The first inequality is due to the monotonicity result in Proposition 2. The second inequality is due to the non-decreasing property of $D(\cdot)$. The last inequality is because it is optimal to transmit in state $(\mathbf{v},\tau+1,m)$. Similarly, if we drop in state $(\mathbf{v},\tau,m)$, we also drop in state $(\mathbf{v},\tau+1,m)$. Therefore, there exists τ_{max}, such that if $\tau \le \tau_{max}$ the optimal policy is to (re)transmit packet m, otherwise, the optimal policy is to drop and transmit packet $m+1$. This τ_{max} is a function of $\tilde{d}^m$ and will vary for each Head of Line packet. ■

We have just established that the optimal transmission

policy for i.i.d. Bernoulli packet drops is of threshold type. We now turn our attention to determining the value of τ_{max} which defines this policy.

Theorem 2 (Value of τ_{max}): In the optimal threshold policy under i.i.d. Bernoulli packet losses, τ_{max} is defined as $\tau_{max} = \max\left\{\tau \middle| \alpha D(\tau) \leq \sum_u s_u \tilde{d}_u^m\right\}$.

Proof: Suppose $\tau = \tau_{max}$, then it is optimal to transmit in the current time slot and, regardless of the transmission outcome, it is optimal to drop the HOL packet in the next time slot. In the current time slot, $\tau = \tau_{max}$, the retransmission and future costs are incurred. The future costs comprise of the dropping cost in the next time slot and the expected total cost after m is removed. With probability $\bar{s_u} = 1 - s_u$, the HOL packet is not received by user u during the retransmission, and distortion cost, $\tilde{d}_u^m = v_u d_u^m$ is incurred for dropping packet m. Then,

$$\begin{aligned} J^*(\mathbf{v}, \tau_{max}, m) &= \alpha D(\tau_{max}) \\ &\quad + E_{(\hat{\mathbf{v}}|\mathbf{v})}[J^*(\hat{\mathbf{v}}, \tau_{max}+1, m)] \\ &= \alpha D(\tau_{max}) + \sum_u \bar{s_u} v_u d_u^m \\ &\quad + E_{(\bar{\mathbf{v}}|\mathbf{1})}[J^*(\bar{\mathbf{v}}, 1, m+1)] \qquad (6) \\ &= \alpha D(\tau_{max}) + \sum_u \bar{s_u} \bar{d}_u^m \\ &\quad + E_{(\bar{\mathbf{v}}|\mathbf{1})}[J^*(\bar{\mathbf{v}}, 1, m+1)] \\ &\leq \sum_u \bar{d}_u^m + E_{(\bar{\mathbf{v}}|\mathbf{1})}[J^*(\bar{\mathbf{v}}, 1, m+1)] \end{aligned}$$

The first equality is by definition. The second equality is because it is optimal to drop at $\tau = \tau_{max} + 1$. The third equality is by definition of $\tilde{d}_u^m$. The final inequality is because it is optimal to transmit at $\tau = \tau_{max}$. This simplifies to $\alpha D(\tau_{max}) \leq \sum_u s_u \tilde{d}_u^m$. Therefore, for all τ that satisfy $\alpha D(\tau) \leq \sum_u s_u \tilde{d}_u^m$, it is optimal to transmit the HOL packet. Otherwise, it is optimal to drop it. ■

We have just shown the optimal policy is of threshold type. Theorem 1 and 2 define the optimal transmission policy for statistically static channels where packet losses are i.i.d. over time and independent across users. Given the channel success probabilities, s_u, the optimal transmission policy can be summarized as:

Policy 1 (Optimal Static Policy): If $s_u(t) = s_u, \forall t$

1) *Initialize*: $m = 1$, $\mathbf{v} = \mathbf{1}$.
2) Given distortion values $\tilde{d}_u^m = v_u d_u^m$:
   (i) **Transmit** the HOL packet if $\alpha D(\tau) \leq \sum_{u=1}^{U} s_u \tilde{d}_u^m$.
   (ii) Otherwise, **drop** HOL packet. $\mathbf{v} = \mathbf{1}$ and transmit packet $m \leftarrow m + 1$.
3) Update the reception vector $\mathbf{v}$ based on the outcome of Step 2.
4) Repeat steps 2-3 in every time-slot.

Therefore, the optimal transmission policy can be interpreted as to transmit the HOL packet when the expected reduction in distortion, $\sum_u s_u \tilde{d}_u^m$, is greater than the expected cost of retransmission, $\alpha D(\tau)$.

The threshold policy described in Policy 1 has low complexity and optimal performance for scheduling media packets with multiple distortion measures and a target frame rate over static channels. The target frame rate can be used to select an appropriate function $D(\tau)$. These are valuable properties for practical implementations.

A. A Heuristic for Quasi-Static Channels

In the case of slowly varying channels compared to the length of the transmission horizon, it is reasonable to assume the channel is static. However, due to varying path loss, fading, and mobility, wireless channels are typically dynamic. For statistically varying channels, we leverage the optimal policy for static channels, Policy 1, to develop a well-justified "quasi-static" heuristic. We assume that in each time slot, the transmitter has accurate information about the current channel success probabilities, which may change over time. We then use Policy 1 to determine whether to transmit or drop the Head of Line packet. However, because the channel success probabilities may change in each subsequent time slot, they are continuously updated to reflect the current success probabilities. If the channels are static over the horizon of transmission of the entire video sequence, this quasi-static policy is optimal and coincides with Policy 1. For slowly varying channels, which are "quasi-static", this policy will achieve near optimal performance. More dynamic channels will likely lead to some loss in performance. The quasi-static transmission policy is as follows:

Policy 2 (Quasi-Static Policy): For time-varying channels, $s_u(t) = s_u(c_u)$:

1) *Initialize*: $m = 1$, $\mathbf{v} = \mathbf{1}$.
2) Given the current channel conditions, $\mathbf{c}$, and distortion values $\tilde{d}_u^m = v_u d_u^m$:
   (i) Transmit the HOL packet if $\alpha D(\tau) \leq \sum_{u=1}^{U} s_u(c_u) \tilde{d}_u^m$.
   (ii) Otherwise, drop HOL packet. $\mathbf{v} = \mathbf{1}$ and transmit packet $m \leftarrow m + 1$.
3) Update the reception vector $\mathbf{v}$ based on the outcome of Step 2.
4) Update the current channel conditions, $\mathbf{c}$.
5) Repeat steps 2-4 in every time-slot.

This algorithm has the same simple form as in the case of i.i.d Bernoulli packet losses. It is easy to implement and independent of packet transmission order. By updating the channel success probabilities in each time slot, this algorithm can adapt with slowly varying channels.

In the following section, we will show, via experimental results, that this heuristic has near optimal performance. Recent work on this type of "quasi-static" scheduling has shown very good results in practice, [21] and [22]. It has also been shown that heuristics based on this approach may have near optimal performance, with finite bounds on the loss of optimality [23].

B. Measure of Delay

In this section, we discuss the relationship between play out delay and distortion. In some scenarios, achieving a certain target play out rate is the highest priority. We examine how to achieve this rate with the minimum incurred media distortion.

The distortion costs are easily measured by comparing the received media to the original. Here we look deeper into the delay costs and the tradeoff with distortion costs. More precisely, how does the delay cost $D(\tau)$ actually correspond to delay?

From Policy 1, the optimal policy is to transmit as long as $\tau \leq D^{-1}\left(\frac{1}{\alpha}\sum_u s_u \tilde{d}_u^m\right)$. Now, suppose the target frame rate is 1 and that $D(\tau) = \tau$. Let τ_m be the expected number of transmissions for packet m. Then the average expected number of transmissions per frame, i.e. the average expected deviation from the target frame rate, is:

$$\begin{aligned} E[\tau] &= \frac{1}{M}\sum_m \tau_m \\ &\leq \frac{1}{M}\sum_m \left[D^{-1}\left(\frac{1}{\alpha}\sum_u s_u \tilde{d}_u^m\right)\right] \\ &= \frac{1}{\alpha M}\sum_m \sum_u s_u \tilde{d}_u^m \end{aligned} \quad (7)$$

The last equality follow from the assumption that $D(\tau) = \tau$. So, if we wanted to guarantee a certain average transmission rate, we could appropriately scale α to achieve that target. For other invertible functions, $D(\tau)$, an appropriate α can be determined for the target transmission rate.

In order to determine the appropriate α for the target transmission rate, we need to know what the expected transmission rate is as a function of α: $E[\tau] = f(\alpha)$. For simplicity, we consider a scenario with 2 users. The case for more users will follow similarly.

For each packet, there are 4 scenarios: $(v_1, v_2) = \{(1,1),(1,0),(0,1),(0,0)\}$. Clearly, it is optimal to drop the HOL packet if $v_1 = v_2 = 0$. Therefore there are 3 thresholds, τ^1_{max},τ^2_{max}, and τ^{12}_{max} corresponding to the maximum number of retransmissions given user 1, 2, or both 1 and 2 have not received the HOL packet. We will drop the max with the knowledge that these τs correspond to thresholds. Without loss of generality, assume that $\tau^1 \geq \tau^2$, equivalently, $d_1^m \geq d_2^m$.

Now for each HOL packet, we can calculate the distribution of the number of retransmissions, τ.

$$\begin{aligned} P(\tau > t) &= \begin{cases} P(v_1 = 1 \cup v_2 = 1), & \tau^2 \leq t \\ P(v_1 = 1), & \tau^1 \geq t > \tau^2 \\ P(v_1 = 1 \cap v_2 = 1), & \tau^{12} \geq t > \tau^1 \\ 0, & t > \tau^{12} \end{cases} \\ &= \begin{cases} \bar{s}_1^{t-1} + \bar{s}_2^{t-1} - (\bar{s}_1\bar{s}_2)^{t-1}, & \tau^2 \leq t \\ \bar{s}_1^{t-1}, & \tau^1 \geq t > \tau^2 \\ \bar{s}_1^{t-1}\bar{s}_2^{t-1}, & \tau^{12} \geq t > \tau^1 \\ 0, & t > \tau^{12} \end{cases} \end{aligned} \quad (8)$$

So that

$$\begin{aligned} E[\tau] &= f(\alpha) = \sum_{t=1}^{\infty} P(\tau \geq t) \\ &= 1 + \sum_{t=2}^{\tau^2}\left[\bar{s}_1^{t-1} + \bar{s}_2^{t-1} - (\bar{s}_1\bar{s}_2)^{t-1}\right] \\ &\quad + \sum_{t=\tau^2+1\wedge 2}^{\tau^1} \bar{s}_1^{t-1} + \sum_{t=\tau^1+1\wedge 2}^{\tau^{12}} \bar{s}_1^{t-1}\bar{s}_2^{t-1} \\ &= 1 + \frac{\bar{s}_1^2 - \bar{s}_1^{\tau^1+1}}{s_1\bar{s}_1} + \frac{\bar{s}_2^2 - \bar{s}_2^{\tau^2+1}}{s_2\bar{s}_2} \\ &\quad + \frac{(\bar{s}_1\bar{s}_2)^{\tau^1+1} + (\bar{s}_1\bar{s}_2)^{\tau^2+1}}{\bar{s}_1\bar{s}_2(1-\bar{s}_1\bar{s}_2)} \\ &\quad - \frac{(\bar{s}_1\bar{s}_2)^2 + (\bar{s}_1\bar{s}_2)^{\tau^{12}+1}}{\bar{s}_1\bar{s}_2(1-\bar{s}_1\bar{s}_2)} \end{aligned} \quad (9)$$

Then given a target frame rate, R, one can determine the α which satisfies $R = f(\alpha)$. In summary, through this approach for selecting α we have the ability to operate the system in a manner which would provide the desired average target frame rate at the clients.

V. Performance Evaluation

In this section, we present experimental results which highlight the performance of Policy 1 and 2. We present our results for 200 CIF frames of the standard Soccer test sequence encoded using H.264/MPEG-4 SVC JSVM 8.6. Each media packet corresponds to a single, whole frame. We encode with a single leading I-frame and an I or P frame every 8^{th} frame with B frames in between. The GOP structure defines how the video sequence is encoded and is depicted in Fig. 3. We assume there are two user types, a low frame rate user and a high frame rate user, and there are 2 users of each type for a total of 4 users. The high frame rate users wish to consume the video at the original 60 frames per second. The low frame rate users wish to consume the video at 30 frames per second in order to save battery power. Therefore, every other B frame of the original video sequence can be discarded without any incurred distortion to the low frame rate user, but with some incurred distortion to the high rate user. Lost frames are reconstructed using frame copy error concealment techniques. If no frames are lost, the maximum achievable PSNR is 40.31dB for the low frame rate users and 39.74dB for the high. We calculate the distortion values of frames by dropping one at a time and calculating the resulting mean-squared error. In order to account for the precedence constraints, we calculate the distortion of a packet m as the distortion incurred with the loss of packet m and all of its children frames. For instance, the distortion of B_2 is that distortion incurred with the loss of frames B_1, B_2, and B_3. Our algorithm relies on these distortion values for each packet and the additive model presented in Section II-B. However, for all results the quoted distortion is the actual distortion incurred by dropping all the lost packets, decoding, and calculating the resulting distortion. In order to focus the distortion versus delay tradeoff under MDM, these experimental results only

utilize the temporal scalability of the video; however, it is certainly possible to consider SNR and spatial scalability as well.

We compare our algorithm to 3 different benchmark policies. If lossless transmission is required, a persistent policy, which transmits each packet until all users successfully receive it must be used. We refer to this policy as the *Persistent* policy. Unfortunately, this policy can lead to arbitrarily long delays if there is a single poor channel that causes a bottleneck, preventing further transmission of waiting frames. We also consider a periodic policy where each packet is allotted the same number of retransmissions. This policy ignores the reception acknowledgements and blindly transmits packets at a fixed, periodic rate. We refer to this repetition coding policy as the *Periodic* policy. A more intelligent transmitter could employ an optimization framework, such as the one discussed in Section IV. Conventional approaches assume a single distortion metric, so for this policy we assume that the scheduler believes all users are high frame rate viewers, i.e., $d_u^m = d_{u'}^m$, $\forall u, u'$. We refer to this policy as the *SDM (Single Distortion Measure)* policy. We refer to Policy 1 and 2 from Section IV which incorporates multiple distortion measures as the *MDM (Multiple Distortion Measure)* policy. For static channels we use Policy 1; for dynamic channels we use Policy 2.

We present performance results in terms of the standard metric for media quality, PSNR$= 10\log(\frac{255^2}{D})$, where D is distortion in mean-squared error over all frames. Note that our algorithm is optimized to minimize distortion which has a non-linear transformation into PSNR. In Fig. 4, we see the performance, in terms of average PSNR of all users, versus the average number of transmissions per packet. For the simulations, we assume the probability of successful transmission is static over subsequent time slots. We examine the case of dynamic success probabilities later. As an illustrative example, we assume the probability of success to the high frame rate users is $.9$ and $.85$. For the low rate users is $.5$ and $.55$. We look at other values of s_L later. We also assume a linear delay cost $D(\tau) = \tau$.

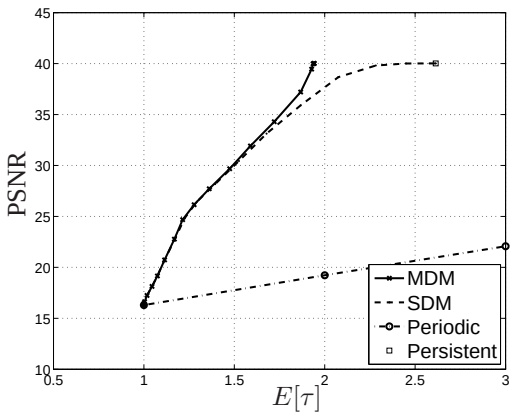


Fig. 4. Static Channels: Transmitting to four users with different channel qualities. Average PSNR versus average number of transmissions.

MDM outperforms the other benchmarks by over 3dB. *Periodic* performs very poorly, performing over 20dB worse. Clearly, utilizing feedback information is very important. *Persistent* achieves the highest PSNR, but because of the bottleneck to the low resolution user, the average number of retransmissions is very high. *MDM* achieves the same PSNR as *Persistent*, but with, on average, 3 fewer retransmission attempts every 4 frames. The effect due to the poor channel to the low frame rate users is most noticeable in Fig. 5, which plots the PSNR performance for the same experiment but for the high and low frame rate viewers separately. The channel quality is so great to the high frame rate users, that after a single retransmission it is highly likely that both have received the packet. However, the low success probability to the low frame rate users causes a bottleneck and requires multiple retransmissions before the PSNR improves. *SDM* and *MDM* can overcome the blocking effect caused by the poor channel–incurring some distortion, but reducing the average number of retransmissions. However, because *MDM* is Multiple Distortion Measure aware, the performance of the low frame rate user is improved. Instead of retransmitting frames which do not help the low frame rate users, *MDM* can drop them without incurring distortion while *SDM* and *Persistent* will not. As such, *MDM* can achieve the same PSNR with fewer retransmissions per frame. Also, with the same average number of retransmissions, *MDM*, achieves gains up to 3dB in average PSNR .

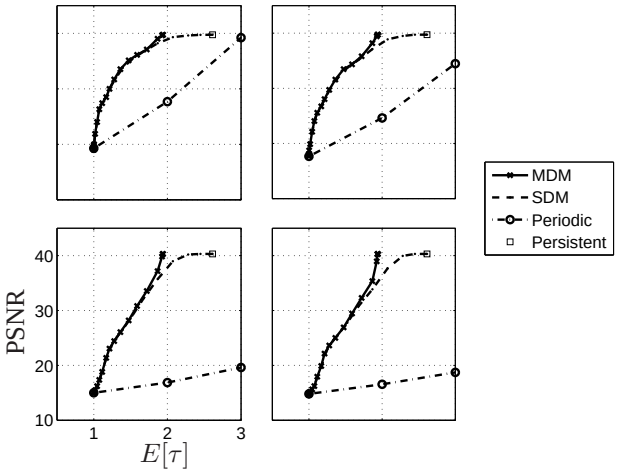


Fig. 5. Static Channels: Transmitting to four users with different channel qualities. Average PSNR versus average number of transmissions for high rate users (top) and low rate users (bottom).

The poor channel quality to the low frame rate users can cause delays. However, only half of the video frames are important to this user type because he is viewing at half the original frame rate. As such, a Multiple Distortion Measure-aware transmitter could drop the packets which are not useful to the low frame rate users to help avoid the bottleneck they causes. In Fig. 6, we examine the average number of transmissions necessary to achieve minimum distortion. As we increase the probability of success to the low rate user with the worse success probability, the bottleneck effect is reduced and the *MDM* and *SDM* policies perform similarly. Because one of the low rate users still has a low probability of success

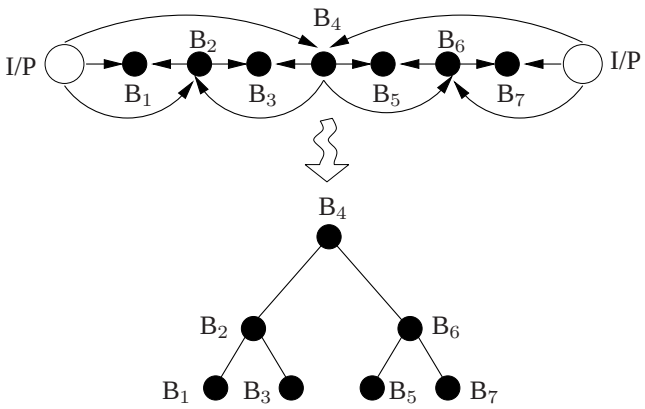


Fig. 3. For hierarchical B frames, the hollow circles correspond to anchor I or P frames. The dependency structure can be mapped to a simple tree structure where each frame's parent is the most junior parent. The most junior parent corresponds to the lowest parent in the precedence tree. For instance, B_3 has 2 parents: B_2 and B_4. However, B_4 is a parent to B_2 so the precedence constraint of B_4 on B_3 is captured by a single precedence constraint of B_2 on B_3. We assume that all I and P frames are successfully transmitted and received, and therefore the dependencies on I and P frames are not listed.

(.55) some bottleneck effect still remains. Not surprisingly, the *SDM* policy requires the same number of retransmissions as *Persistent* to achieve the highest viewing quality because it is not aware that some frames are useless to the low frame rate user and transmits them until all users have received each frame. *MDM* can reduce the number of retransmissions by nearly a factor of 2 by realizing some packets do not benefit the low frame rate user.

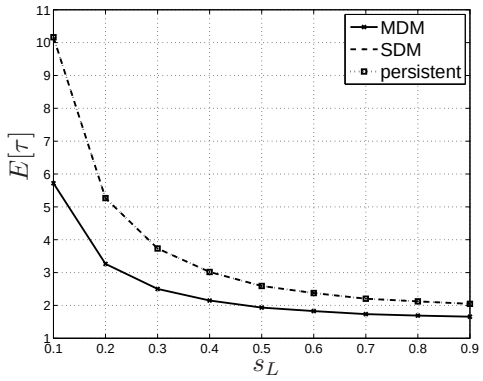


Fig. 6. Static Channels: Expected number of transmissions to achieve highest PSNR quality versus channel quality to low rate user.

is transmitted until it is received by both users. Because of the quality of the channel, the number of transmissions is less than the target frame rate. Once all users receive the necessary frame, the HOL packet is dropped, even if the target frame rate is higher. Clearly, by delaying the play out at the receiver, the target frame rate can be achieved exactly if the number of retransmissions is lower than this rate. Alternatively, the transmitter can idle during some time slots in order to achieve the desired rate at the receiver.

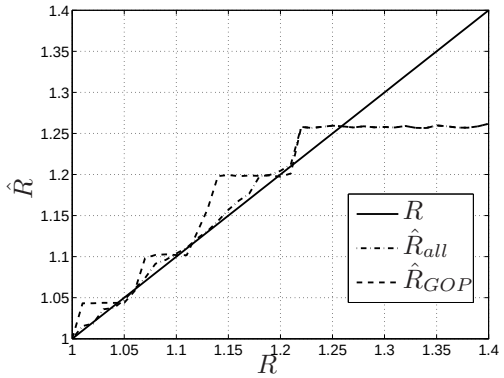


Fig. 7. Select α to achieve a target frame rate R. $\hat{R}_{all}$ selects α based on *all* frames while $\hat{R}_{GOP}$ selects α every 8 frames.

A. Delay

In Section IV-B, we presented a method to determine α to achieve a target frame rate, R, given the channel success probabilities and distortion values of the frames. We again consider the scenario of static channels. This time with one high frame rate user with channel success probability .95 and one low frame rate user with channel success probability .7. We search for the α which satisfies Eqn. 9 for our target rate R. The empirical rate $\hat{R}$ is plotted versus the target rate of R in Fig. 7. The empirical frame rate is quite close to the desired frame rate. Because the channels are so good, the expected frame rate saturates at the expected number of transmissions (1.252) until all packets are received. For large R, each packet

It may be impractical to assume the scheduler has access to all frames at once in order to determine $\alpha(R)$. If we modify each α for each frame, the different distortion values of each frame would be ignored and the algorithm would reduce to the periodic strategy which we have seen perform quite poorly. For real-time applications, we propose determining an $\alpha(R)$ for each Group-of-Pictures (GOP), so that knowledge of *all* frames is not required, but that there is enough diversity in the distortion values of the frames to be able to tradeoff transmission for a low distortion frame to allow high retransmissions for a high distortion frame. The empirical rate $\hat{R}_{GOP}$ is plotted versus the target rate of R in Fig. 7. We see that by modifying α for each GOP, the frame rate deviates slightly from target, although minimally. The frequent update of α is also beneficial

for the case of time-varying channels.

B. Dynamic (non-Quasi-Static) Channels

Thus far, we have examined statistically static channels. Due to user mobility and other physical phenomenon, it is often the case that the channel quality is varying. We model the channels as 2-state Markov Chains. To stress and assess our quasi-static heuristic policy, we examine the following challenging channel. The probability of success to the high rate user is .9 or .6. The probability of success to the low rate user is .9 or .1. We assume that the transition probabilities are .8. Therefore, the channel is varying quickly and we expect this to stress the performance of the quasi-static algorithm. We use the quasi-static heuristic presented in Section IV-A. We compare the performance of this heuristic to the optimal policy which we find using Dynamic Programming. In order to ensure that the state space is sufficiently small that Dynamic Programming techniques are possible, we introduce a sufficiently large maximum possible number of retransmissions, $R_{max} = 50$, and consider only 2 users–a low and high rate user. One may expect the quickly varying channel to cause the heuristic policy of Policy 2 to fail, but this is not the case. Fig. 8 shows the average PSNR versus the average number of retransmissions. We can see that the heuristic policy is within 1dB of optimal and outperforms the conventional *SDM* approach by over 5dB.

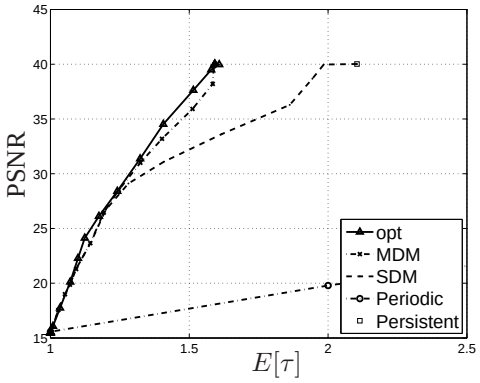


Fig. 8. Dynamic Channels: Transmitting to two users with channel qualities modeled as 2-state Markov Chains. Average PSNR versus average number of transmissions.

VI. Summary

In this paper we have examined the problem of broadcasting multimedia packets with Multiple Distortion Measures. With the growth in diversity of mobile multimedia users, MDM can satisfy the diverse needs of each user. Using Dynamic Programming techniques, a simple, optimal transmission policy for broadcasting packets with Multiple Distortion Measures over statistically static channels was presented. This policy was shown to be of threshold type, so that the decision to transmit or drop a packet can be calculated online. We gained insight into the fundamental tradeoff between delay and distortion. We presented an algorithm which minimizes aggregate distortion given a target frame rate. Experimental results showed that this policy outperforms current media systems which do not consider MDM. Leveraging the optimal policy for statistically static channels, a quasi-static heuristic for general channel dynamics was proposed. This heuristic was experimentally shown to have near optimal performance. In this work, we have shown that accounting for MDM can significantly improve QoS of multimedia users. Furthermore, we have shown that simple, high performing algorithms can be developed within this framework.

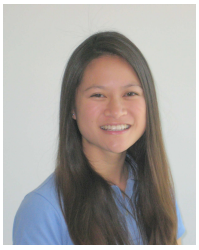
Acknowledgment

The authors would like to thank the anonymous referees for useful comments that improved the exposition of this work.

References

[1] C. W. Chan, S. Wee, and J. Apostolopoulos, "Multiple distortion measures for packetized scalable media," *IEEE Trans. Multimedia*, vol. 10, pp. 1671–1686, Dec. 2008.
[2] *15444 JPEG-2000 Image Coding System, Part 1: Core coding system*.
[3] Joint Video Team of ISO/IEC MPEG & ITU-T VCEG, *AHG Report on Spatial Scalability Resampling, Document JVT-Q007*, Oct. 2005.
[4] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. Multimedia*, vol. 8, pp. 390–404, Apr. 2006.
[5] Z. Miao and A. Ortega, "Optimal scheduling for streaming of scalable media," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, vol. 2, (Pacific Grove, CA, USA), pp. 1357–1362, Nov. 2000.
[6] Y. Wang, M. van der Schaar, S.-F. Chang, and A. C. Loui, "Classification-based multidimensional adaptation prediction for scalable video coding using subjective quality evaluation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, pp. 1270–1279, Oct. 2005.
[7] N. Bansal, D. Coppersmith, and M. Sviridenko, "Improved approximation algorithms for broadcast scheduling," in *Proc. SODA*, pp. 344–353, 2006.
[8] A. Bar-Noy, S. Guha, Y. Katz, J. Naor, B. Schieber, and H. Shachnai, "Throughput maximization of real-time scheduling with batching," in *Proc. SODA*, pp. 742–751, 2002.
[9] J.-H. Kim and K.-Y. Chwa, "Scheduling broadcasts with deadlines," *Theoretical Computer Science*, vol. 325, pp. 479–488, 2004.
[10] M. Etoh and T. Yoshimura, "Wireless video applications in 3g and beyond," *IEEE Wireless Commun. Mag.*, vol. 12, pp. 66–72, Aug. 2005.
[11] C. J. Su and L. Tassiulas, "Broadcast scheduling for information distribution," in *Proc. IEEE INFOCOM*, vol. 1, pp. 109–117, Apr. 1997.
[12] M. Sharif and B. Hassibi, "Delay guarantee versus throughput in broadcast fading channels," in *Proc. IEEE ISIT*, p. 248, July 2004.
[13] C. W. Chan, N. Bambos, S. Wee, and J. Apostolopoulos, "Optimal scheduling of media packets with multiple distortion measures," in *Proc. IEEE ICME*, pp. 955–958, July 2007.
[14] C. W. Chan, N. Bambos, S. Wee, and J. Apostolopoulos, "Wireless video broadcasting to diverse users," in *Proc. IEEE ICC*, pp. 377–382, May 2008.
[15] Q. Zhang and S. A. Kassam, "Finite-state markov model for rayleigh fading channels," *IEEE Trans. Commun.*, vol. 47, pp. 1688–1692, Nov. 1999.
[16] K. Stuhlmüller, N. Färber, M. Link, and B. Girod, "Analysis of video transmission over lossy channels," vol. 18, pp. 1012–1032, June 2000.
[17] H. Yang and K. Rose, "Advances in recursive per-pixel end-to-end distortion estimation for robust video coding in h.264/avc," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, pp. 845–856, July 2007.
[18] Y. Liang, J. Apostolopoulos, and B. Girod, "Analysis of packet loss for compressed video: Effect of burst losses and correlation between error frames," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, July 2008.
[19] J. Apostolopoulos, "Secure media streaming & secure adaptation for non-scalable video," in *Proc. IEEE ICIP*, vol. 3, pp. 1763–1766, Oct. 2004.
[20] D. Bertsekas, *Dynamic Programming and Optimal Control*, vol. 1 & 2. Athena Scientific, 2nd ed., 2000.
[21] A. Dua and N. Bambos, "Downlink wireless packet scheduling with deadlines," *IEEE Trans. Mobile Comput.*, vol. 6, pp. 1410–1425, Dec. 2007.
[22] J. Huang, R. Berry, and M. Honig, "Wireless scheduling with hybrid arqs," *IEEE Trans. Wireless Commun.*, vol. 4, pp. 2801–2810, Nov. 2005.

[23] C. W. Chan and V. F. Farias, “Stochastic depletion problems: Effective myopic policies for a class of dynamic optimization problems,” *Mathematics of Operations Research*. to appear.



Carri W. Chan Carri W. Chan is currently a Ph.D. candidate in the Electrical Engineering Department at Stanford University. She received her B.S. degree in Electrical Engineering from the Massachusetts Institute of Technology in 2004 and her M.S. degree in Electrical Engineering from Stanford University in 2006. She is a recipient of the STMicroelectronics Stanford Graduate Fellowship. Her research interests include stochastic modeling and optimization of service and computing systems. She has looked at scheduling problems for packetized multimedia streaming over wireless networks. Recently, her research interests have included scheduling problems in healthcare operations.

Nick Bambos received his Ph.D. in electrical engineering and computer science from U.C. Berkeley in 1989, after graduating in Electrical Engineering from the National Technical University of Athens, Greece in 1984. He served on the Electrical Engineering faculty of UCLA from 1990 to 1995 and joined Stanford University in 1996, where he is now a professor in the Electrical Engineering department and the Management Science & Engineering department. His current research interests are in performance engineering of communication networks and computing systems, including queuing and scheduling issues in wireless an wireline networks, as well as ergodic random processes, queuing theory and adaptive control of stochastic processing networks.

Susie Wee is currently the Vice President of the HP Experience Software Business. She has been with HP for over 11 years, previously as the Director of the Mobile and Media Systems Lab in HP Labs. She has worked closely with many HP businesses and she has led international research collaborations with NTT DoCoMo in Japan, Infocomm Institute of Research in Singapore, and universities such as MIT. Susie’s research interests are in video processing, image security and multimedia streaming. Computerworld named her as one of the ”Top Innovators” under age 40 in 2007 and she was named one of the ”Top Innovators” under age 35 by MIT Technology Review in 2003. Susie was recently awarded the Technical Excellence Award from INCITS for her work on creating the international standard on JPEG2000 Image Security (JPSEC). She holds over 25 granted patents and was a consulting assistant professor at Stanford University co-teaching a graduate class on digital video processing. Susie was formerly an associate editor of the IEEE Transactions on Image Processing and the IEEE Transactions on Circuits, Systems, and Video Technology. Susie received her Bachelor of Science, Master of Science and Ph.D. degrees in Electrical Engineering and Computer Science from MIT.

John Apostolopoulos (S’91, M’97, SM’06, F’08) received the B.S., M.S., and Ph.D. degrees in EECS from MIT. He joined Hewlett-Packard Laboratories in 1997, where he is currently the Lab Director for the Multimedia Communications and Networking Lab, and a Distinguished Technologist. He also teaches and conducts joint research at Stanford University, where he is a Consulting Associate Professor of EE, and also regularly lectures at MIT. In graduate school, he worked on the U.S. Digital TV standard and received an Emmy Award Certificate for his contributions. He received a best student paper award for part of his Ph.D. thesis, the Young Investigator Award (best paper award) at VCIP 2001 for his work on multiple description video coding and path diversity, was named ”one of the world’s top 100 young (under 35) innovators in science and technology” (TR100) by Technology Review in 2003, and was co-author for the best paper award at ICME 2006 on authentication for streaming media. His work on media transcoding in the middle of a network while preserving end-to-end security (secure transcoding) was recently adopted by the JPEG-2000 Security (JPSEC) standard. He currently serves as chair of the IEEE IMDSP and member of MMSP technical committees, and recently was general co-chair of VCIP’06 and technical co-chair for ICIP’07. His research interests include improving the reliability, fidelity, scalability, and security of media communication over wired and wireless packet networks.