A Quantitative Framework for guaranteeing QoE of Video delivery over Wireless

Hemant Kowshik*, Partha Dutta*, Malolan Chetlur* and Shivkumar Kalyanaraman*
*IBM India Research Lab, Bangalore
Email: {hkowshik, parthdut, mchetlur, shivkumar-k}@in.ibm.com

***Abstract*—The rapid growth of mobile video traffic is placing a severe strain on mobile network operators. In this paper, we study the problem of efficient video delivery over the cellular downlink. The key objective is to maximize the Quality of Experience (QoE) of the user, which cannot be entirely captured by traditional QoS metrics like packet loss rate, delay and throughput. In this paper, we consider application level metrics such as the buffering ratio and low bit rate ratio, that are known to closely correlate with user QoE. Our goal is to develop a framework that can guarantee a pre-specified QoE for each user, provided it is feasible under the current channel conditions. We present a two-tiered solution with a standard basestation scheduler that works on a per-packet basis and a Video Management System (VMS) that works at the granularity of thousands of video frames.**

The video management system uses knowledge of the video playout curves and the average future channel states to develop a scheduling policy that is feasibility optimal, i.e., it can satisfy any set of QoE guarantees that can be satisfied by any other scheduling algorithm. We describe algorithms that maintain a guaranteed buffering ratio, or a guaranteed low bit rate ratio, or both. The algorithms are simple and leverage recent results on real-time scheduling in wireless networks. We evaluate the performance of our algorithms using real video traces and a standard channel model. The VMS ensures that the per-user QoE guarantees are maintained, as compared with a standard PF scheduler that is oblivious to application level QoE requirements.

I. Introduction

With the advent of 4G cellular networks and the increasing adoption of smartphones, mobile traffic is growing at an unprecedented rate. According to a recent industry report, mobile traffic is expected to increase 18 fold in the period from 2011 to 2016 [1]. With the increasing data rates in 4G cellular networks, a host of video-based applications can be supported on the mobile device. This includes applications like streaming video, video calling and mobile gaming. Not surprisingly, video traffic is a major component of the mobile traffic mix, currently accounting for 52% of mobile traffic, and expected to rise to 70% by the year 2016 [1].

There are primarily two ways for operators to keep up with the growing demand. First, operators could invest in new network infrastructure which involves enormous capital expenditure but may yield limited revenue. Second, operators could optimize use of the wireless spectrum, easing the load on the network and ensuring graceful degradation in overload scenarios. Towards this end, we address the problem of optimizing video delivery to multiple mobile users from a cellular base station, so as to maximize network utilisation and provide guarantees on the Quality of Experience (QoE).

We fix attention on streaming and video-on-demand flows generated by services like Youtube and Netflix. Our objective is to develop a quantitative framework for delivering video with QoE guarantees, and subsequently devise algorithms that optimize the wireless spectrum resource while still providing performance guarantees. The framework must be rich enough to capture different performance metrics, and provide differentiated service guarantees for different classes of users.

In this paper, we seek to provide guarantees on the QoE for each video in a set of multiplexed video requests. The classical approach is to treat the requests as data streams and schedule them in an intelligent manner so as to maximize utilization and ensure fairness among users. A host of basestation scheduling policies have been proposed to optimize quantities like throughput, per-packet delay and minimum guaranteed bitrate. However, these quantities while being insightful for understanding data traffic, fail to capture the key aspects of a video watching experience. The problem of quantifying video experience is multi-faceted and subjective to a certain extent. However, recent work has established some key parameters of interest [2]. For instance, the quality of experience is sensitive to stalls, pixelation and jitter, while being insensitive to losing a small fraction of frames, lowering the bit rate for a short period of time and omitting a few unimportant frames.

The main contribution of this paper is a framework that directly considers metrics of experience, and algorithms that provide long-term QoE guarantees to each user. The key observation we make is that while schedulers work at a per-packet granularity, the experience of video watching is only affected at the granularity of a few seconds. For instance, users are more comfortable with a single stall of 5 seconds than 10 stalls of 500ms each. To address this discrepancy between packet scheduling and video experience timescales, we present a two-tiered solution consisting of a standard basestation scheduler that works at a MAC frame-level granularity (timescale of 2-5 ms) and a Video Management System(VMS) that works at the granularity of *superframes* (of the order of a few thousand frames = a few seconds). Given the average bitrates over the next superframe by the basestation scheduler, the VMS must choose the subset of videos to serve, and their bitrates.

We use a Lyapunov function based approach to derive scheduling policies that maintain per-user QoE guarantees in the face of arbitrary channel variation and arbitrary arrivals and departures of requests. The proposed policies are feasibility

optimal, i.e., the admissible region of these policy dominates the admissible region of any other policy. We consider a standard video delivery system, where the video client stalls the video when the buffer is empty. Further, we consider the scenario where the video server may deliver videos of different resolutions depending on the network condition. We consider two key QoE metrics, namely the Buffering Ratio defined to be the fraction of time the user spends in a stalled state, and the Low Bitrate Ratio which is the fraction of time the user is served a lower resolution video. For these metrics, the scheduling problem reduces to solving a fractional knapsack problem where the greedy algorithm is optimal. Further, we show how the framework can be extended to provide a joint guarantee on the buffering ratio and low bit rate ratio.

Next, we present how this scheduling framework can be extended to a more general utility framework where each video user specifies a certain utility for delivery of the next frame. Then, the task of the scheduler is to schedule video streams so that each user's long-term average utility satisfies a certain guarantee. Finally, we conduct a simulation study to evaluate the performance of our algorithms against a standard basestation scheduler. The simulations show that in the presence of our algorithms, each user adheres to it's QoE guarantee, irrespective of the bitrate of the video. This is in contrast with the weighted Proportional Fair (PF) scheduler which shows widely varying QoE depending on the bitrate of the video and channel conditions.

II. Related Work

The variable bit rate of compressed videos makes it difficult to provide performance guarantees on their delivery. Multiple smoothing schemes have been proposed in the literature to address this problem [3], [4], [5], [6], [7]. In this work, we require that the bit rate of a video is maintained to be a constant over the duration of a superframe. This requirement can be ensured using one of the existing smoothing schemes, for instance, the buffering based scheme proposed in [7] that minimizes the variance of the bit rate of a video for a given client buffer size.

The classical solution to the problem of multiplexing several data streams with QoS requirements over the wireless channel is MAC-layer scheduling. Several scheduling algorithms have been proposed for wireless networks[8] [9], and in particular cellular networks [10]. These algorithms seek to optimize QoS parameters like delay, jitter, throughput and packet-loss rate [11] [12]. In this paper we use a Lyapunov function based approach introduced in [13]. More recently, this approach has been used to analyze delay behaviour [14], and to devise scheduling policies for real-time traffic [15]. In contrast with the prior scheduling work, we are interested in guaranteeing QoE rather than optimizing classical QoS parameters.

Managing the delivery of video streams over wireless channel has been studied in multiple papers [16], [17], [18], [19]. The authors in [16], [17] study the probability distribution of the number of buffering events (or stalls) in the playout of a single video delivered over a wireless channel. Scheduling schemes for delivery of multiple videos over wireless has been studied in [19] to reduce and fairly distribute the number of buffering events, and in [18] to reduce energy consumption. However, unlike our work, the methods proposed in these papers do not provide any QoE guarantee, and implementing these methods requires modification to the base station scheduling scheme.

Multiple techniques have been designed to limit the impact of deteriorating network conditions on the QoE of video delivery. Prioritizing video packets based on their importance [20] and scalable video coding [21], [22], [23] are two such approaches. Our framework differs from earlier work on prioritizing video packets in two ways: first, prioritization is done at the level of video streams rather than individual packets, and second, the time-granularity of serving packets of a video is at the superframe level rather than single video frames. This enables us to intervene at the time scale of experience, while avoiding changes to the base station scheduler.

III. Description of Video Delivery Framework

We consider the single cell scenario where we have a single basestation serving multiple video-on-demand requests seeking immediate playout. The video requests arrive and depart in arbitrary fashion. Each video is variable bit rate (VBR) and the wireless channel has variable throughput due to fading and shadowing effects. Given the channel quality and the state of playout for each user, the objective of the basestation is to decide which video requests to serve, and at what bitrates.

The classical approach to ensure quality of service (QoS) among competing flows is to enqueue video requests and devise a priority-based algorithm which serves queues in priority order. The assigned priority typically depends on channel quality and queuelength. These algorithms guarantee throughput optimality, i.e., they ensure optimal utilization of the wireless channel, while still maintaining QoS guarantees. Weighted Fair Queueing (WFQ) and Modified Largest Weighted Delay First (M-LWDF) are well known examples. However, from the point of view of optimizing QoE, there are several reasons why the basestation scheduler does not suffice.

(i) Basestation schedulers are application unaware, and hence unable to actively monitor the QoE of the end-user. Further, since the scheduler does not know the current video bitrate, or the state of playout of the video, there is very little opportunity to intelligently prioritize flows and provision bandwidth. From a practical standpoint, modifying the basestation scheduler may not be acceptable to mobile network operators.

(ii) Channel-aware schedulers work at a per-packet level and prioritize requests with longer queuelengths and better channels, so as to achieve throughput optimality. Thus, there is a discrepancy between the timescale at which the scheduler works (typically a few milliseconds) and the timescale at which user experience is evaluated (typically a few seconds).

(iii) The QoS guarantees only apply if the set of video

requests fall within the capacity region. At peak periods however, the basestation may be underprovisioned and the application designer has no handle on how the QoE degrades. In this scenario, we know that stalls will be inevitable, and we would ideally like them to be introduced in a controlled manner.

(iv) In the case of video delivery, a certain minimum throughput is required for acceptable performance. Any bitrate below this will result in stalling and degraded QoE. This sharp threshold behavior distinguishes video traffic from file transfers.

Thus, we need to reinvent the scheduling framework for video delivery, so as to make it possible for application designers to provide QoE guarantees. The proposed architecture is presented in Figure 1.

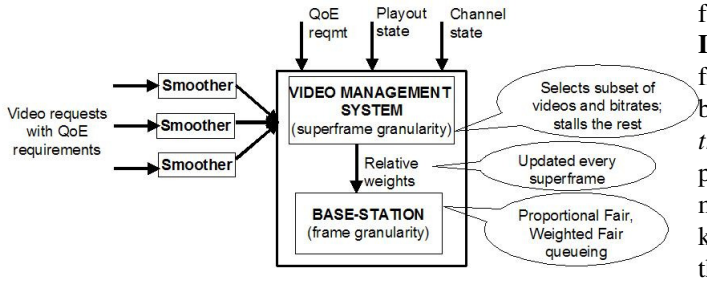


Fig. 1. Two-tiered Architecture

Each of the incoming video requests is sent to a smoother which reduces the short timescale variability in the bitrate of the video. Thus, the bitrate of the smoothed video is assumed to vary much more slowly, staying constant over the course of a few thousand frames, which we define to be a *superframe*. We present a two-tiered scheduling framework. At the lower tier, we have a standard basestation scheduler, which promises certain bitrates to each user, given the current channel conditions. At the higher tier, there is a video management system (VMS) which works at the timescale of a superframe. At the beginning of each superframe, the VMS chooses a subset of video requests to serve actively and their respective rates. These video requests will be played out during the current superframe while the other requests are either buffering at a lower rate, or completely throttled. These videos will appear stalled to the user. [1]

Under this framework, the problem of interest is: given the channel condition, required bitrate and the history of stalls for each video request, which subset of videos must be served actively during the current superframe, so as to guarantee prespecified QoE metrics for each user.

[1]We have not modeled the behavior of the video player, which could employ sophisticated buffering strategies. In our treatment, we focus on the problem of video delivery at the base-station. Hence, we suppose that a request that is not being actively served is stalled.

A. QoE metric

Several metrics have been proposed in the literature to measure the quality of experience of streaming video. Our effort will be to create a framework that can handle multiple QoE metrics. There are two key metrics that we study, while indicating how the framework can be generalized to other possible metrics.

Buffering Ratio (BR): The *buffering ratio* is defined to be the fraction of the video session when the user is stalled. A recent study [2] indicates that the buffering ratio is the parameter that is most highly correlated with dissatisfied users terminating their video session. Further, the study suggests that users prefer a single stall of reasonable length in comparison to several short stalls with the same total length. Putting these two observations together, we suppose that the stall length is at least one superframe (typically 2 to 5 seconds), and we seek to guarantee that the long term fraction of stalled superframes for user i is less than α_i.

Low Bitrate Ratio (LBR): Another measure of QoE derives from the fact that videos are not always served at a fixed bitrate. Depending on the available bandwidth, the server can *transcode* the video frames to a lower or higher bitrate. This presents another degree of freedom in designing the QoE metric. We consider a metric, called the low bit rate ratio, that keeps track of the long-term fraction of superframes served at the lower bit rate. Thus, we seek to guarantee that the low bit rate ratio is less than β_i.

B. Debt-based Feasibility Optimal Policy

The idea of providing long term deterministic guarantees in the face of stochastic channel variation has been explored extensively in the literature on queuing theory. We adapt some results on providing real-time QoS guarantees [14],[15] to develop optimal policies which satisfy the required QoE guarantees.

For ease of exposition, we begin with the buffering ratio metric, and later show how the framework can be easily extended to include a rich class of metrics, including the low bit rate ratio defined above. As indicated before, we suppose that time is divided into superframes, where each superframe is the length of a few thousand frames. It is understod that the video management system only changes policy at the edge of a superframe. We begin by introducing some notation

Consider a set of N users in the system each of who can initiate and terminate a video at the edge of a superframe. Let S_k be the set of users with active video requests in superframe k. Let us denote the bitrate of video i during superframe k to be $b_i(k)$. Since the video has been smoothed beforehand, we suppose that the bitrate of the video stays constant during a superframe. We suppose that the set of active video requests S_k, and their bitrates $b_i(k)$ evolve according to a stationary irreducible Markov chain with finite state. We also suppose that the set of active video requests at time t and their available bitrates are independent of the channel states at time t. However, the selected bitrates of videos will depend on the channel state.

At the beginning of superframe k, we suppose that we have lookahead channel state information, which can be translated into an maximum feasible bitrate $B_i(k)$ for user i in superframe k. Thus, if video i must be served, the base-station needs to allocate $\frac{b_i(k)}{B_i(k)}$ fraction of slots in superframe k to user i. Let $x_i(k)$ be an indicator variable that is 1 if video i is served in superframe k, and 0 otherwise. In our basic formulation, we do not allow videos to be served partially. However, in Section III-D, we show how serving video fractions is a straightforward extension.

QoE requirement: Define $T_i(k) := \sum_{l=1}^{k} \mathbf{1}\{i \in S_l\}$ where $\mathbf{1}$ is the indicator function. For simplicity, we will suppose that $T_i(k) \to \infty$ as $k \to \infty$, for all i. Our objective is to guarantee that

$$\lim_{k\to\infty} \frac{1}{T_i(k)} \sum_{l=1}^{k} (x_i(l)\mathbf{1}\{i \in S_l\}) \geq (1-\alpha_i) \quad \text{for each } i$$

In order to monitor the QoE requirement, we define the performance deficit for user i at the end of superframe k to be

$$\delta_i(k) = (1-\alpha_i)T_i(k) - \sum_{l=1}^{k} (x_i(l)\mathbf{1}\{i \in S_l\}),$$

which is the number of superframes where user i should have been served minus the number of superframes where user i was actually served. The performance deficit for user i evolves according to the following equation

$$\delta_i(k+1) = \begin{cases} \delta_i(k) + (1-\alpha_i) - x_i(k) \text{ if } i \in S_k, \\ \delta_i(k) \text{ otherwise.} \end{cases}$$

For simplicity, let us define $z_i := 1-\alpha_i$. We say that a vector $[z_i]$ is feasible if there exists a scheduling policy that ensures that the QoE requirement for each user is met. Further, since the whole system can be modeled as a controlled Markov chain, any feasible vector $[z_i]$ can in fact be fulfilled by a stationary randomized policy that is allowed to depend on the channel states and the history of stalls. A feasible vector $[z_i]$ is said to be strictly feasible if $[z_i/\theta]$ is also feasible for some $\theta < 1$.

Definition 1 (Feasibility Optimality cf. [15]): A scheduling policy is said to be *feasibility optimal*, if it can fulfil any QoE vector $[z_i]$ that is strictly feasible.

In this paper, we do not suppose that we know the pattern of arrivals and departures of video requests and hence, we do not consider the problem of determining the feasible region. However, we present a feasibility optimal scheduling policy using a Lyapunov function based approach, along the lines of Theorem 3 in [15]. We present the proof here for completeness. To begin with, we recall the following standard result from Lyapunov stability theory.

Theorem 1: Let $L(k)$ be a non-negative Lyapunov function. Suppose there exists some constant $B > 0$, and a non-negative function $f(k)$ adapted to the past history of the system such that:

$$E[L(k+1) - L(k)| \text{ history up to time k}] \leq B - \epsilon f(k),$$

for all k, then $\limsup_{k\to\infty} \sum_{l=1}^{k} E\{f(l)\} \leq B/\epsilon$.□

Theorem 2: Consider a scheduling policy that solves the following optimization problem at the beginning of the k^{th} superframe.

$$\text{Max.} \sum_{i\in S_k} \mathbf{E}\left[\delta_i^+(k-1)x_i(k)|[b_i(k)],[B_i(k)],S_k,[\delta_i(k-1)]\right] \tag{1}$$

The above policy is feasibility optimal.

Proof: Let us begin by defining a Lyapunov function

$$L(k) = \frac{1}{2}\sum_{i=1}^{N} \delta_i^2(k\ -\ 1).$$

The drift of the Lyapunov function $\Delta L(k)$ is defined to be

$$\begin{aligned}
& \mathbf{E}\left[L(k+1) - L(k)|S_k,[\delta_i(k-1)]\right] \\
= \ & \frac{1}{2}\mathbf{E}\left[\sum_{i=1}^{N} \delta_i^2(k) - \delta_i^2(k-1)|S_k,[\delta_i(k-1)]\right] \\
= \ & \sum_{i\in S_k} \frac{1}{2}\Bigg(\mathbf{E}\left[\frac{1}{2}\left(\delta_i(k-1)+1-\alpha_i-x_i(k)\right)^2 \right. \\
& \qquad \left. -\delta_i^2(k-1)\right)|S_k,[\delta_i(k-1)]\Big] \\
= \ & \sum_{i\in S_k} \frac{1}{2}\left(1-\alpha_i-x_i(k)\right)^2 \\
& + \sum_{i\in S_k} \mathbf{E}\left[\delta_i(k-1)\left(1-\alpha_i-x_i(k)\right)|S_k,[\delta_i(k-1)]\right] \\
\leq \ & \frac{N}{2} + \sum_{i\in S_k} \mathbf{E}\left[\delta_i(k-1)\left(1-\alpha_i-x_i(k)\right)|S_k,[\delta_i(k-1)]\right]
\end{aligned} \tag{2}$$

Let us define $z_i := 1-\alpha_i$. Then, the set of feasible vectors $[z_i]$ is a convex set. Indeed, given two points in the feasible region, one can achieve any convex combination by timesharing. Let $(z_1, z_2, \ldots, z_N)$ be an interior point in the feasible region. Let ϵ be small enough so that $[z_i'] \equiv (z_1+\epsilon, z_2+\epsilon, \ldots, z_N+\epsilon)$ is also inside the feasible region. Thus there exists a stationary randomized policy, say $\mathcal{P}'$, that satisfies users with QoE requirements specified by $[z_i']$. Let $x_i'(k)$ be the decrease in performance deficit for user i under $\mathcal{P}'$. Thus, we have, for any $i \in S_k$,

$$\begin{aligned}
& \mathbf{E}\left[x_i'(k)|S_k,[\delta_i(k-1)]\right] \\
= \ & \mathbf{E}\left[\mathbf{E}\left[x_i'(k)|[b_i(k)],[B_i(k)]\right]|S_k,[\delta_i(k-1)]\right] \\
\geq \ & z_i + \epsilon,
\end{aligned} \tag{3}$$

where the inner expectation is over the random variables $[b_i(k)], [B_i(k)]$. Further, if policy $\mathcal{P}$ achieves the maximum in (1), then we have

$$\begin{aligned}
& \sum_{i\in S_k} \mathbf{E}(\delta_i^+(k-1)x_i(k)|[b_i(k)],[B_i(k)],S_k,[\delta_i(k-1)]) \\
\geq & \sum_{i\in S_k} \mathbf{E}(\delta_i^+(k-1)x_i'(k)|[b_i(k)],[B_i(k)],S_k,[\delta_i(k-1)])
\end{aligned}$$

We can restrict attention to policies that only work on clients with $\delta_i(k-1) > 0$, and show that one can still establish

feasibility optimality, despite doing less work. This is due to the fact that the objective function (1) is unaffected by serving flows with negative deficit. From (2), we have the following sequence of upper bounds on $\Delta L(k)$.

$$\begin{aligned} & \frac{N}{2} + \sum_{i \in S_k} \mathbf{E}\left[\delta_i^+(k-1)(1-\alpha_i - x_i(k))|S_k, [\delta_i(k-1)]\right] \\ \leq & \frac{N}{2} + \sum_{i \in S_k} \mathbf{E}\left[\delta_i^+(k-1)(1-\alpha_i - x_i'(k))|S_k, [\delta_i(k-1)]\right] \\ \leq & \frac{N}{2} + \sum_{i \in S_k} \delta_i^+(k-1)\left[(1-\alpha_i) - (1-\alpha_i+\epsilon)\right] \\ = & \frac{N}{2} - \epsilon\left(\sum_{i \in S_k} \delta_i^+(k-1)\right) \end{aligned}$$

where the second inequality follows from (3). Thus, from Theorem 1, we have that

$$\limsup_{k\to\infty} \frac{1}{k}\sum_{l=1}^{k} \mathbf{E}\left[\sum_{i \in S_{l+1}} \delta_i^+(l)\right] \leq \frac{N}{2\epsilon}.$$

Finally, since $|\sum_{i=1}^N \delta_i^+(k) - \sum_{i=1}^N \delta_i^+(k-1)|$ is bounded for all k, we have

$$\frac{1}{k}\mathbf{E}\left[\sum_{i=1}^{N} \delta_i^+(k)\right] \to 0 \text{ as } k \to \infty.$$

Thus, $\frac{\delta_i^+(k)}{k}$ converges to zero in probability. Thus, we have shown that the prescribed strategy of solving (1) at the beginning of each superframe, is feasibility optimal. □

C. Solving the knapsack problem

Since we have assumed that we have accurate lookahead channel state information at the beginning of each superframe, the problem in (1) reduces to a deterministic knapsack problem. The linear objective function becomes the profit function that we seek to maximize, and the capacity constraint on the downlink translates to a knapsack size constraint.

$$\begin{aligned} [KNAPSACK] \quad & \text{Maximize } \textstyle\sum_{i \in S_k} \delta_i^+(k-1)x_i(k) \\ & \text{s.t. } \textstyle\sum_{i \in S_k} \frac{b_i(k)x_i(k)}{B_i(k)} \leq 1 \end{aligned}$$

That is, given a set of items where the i^{th} item has cost $\frac{b_i(k)}{B_i(k)}$ and profit $\delta_i^+(k-1)$, one needs to pick a subset of items to maximize profit for a given total cost.

The knapsack problem, of course, is known to be NP-complete, and has been widely studied in the algorithms literature. The greedy algorithm of picking items in non-increasing order of their profit per unit cost gives a solution whose value is greater than half of the optimal value. Further, the approximation ratio for the greedy algorithm can be improved to $1/(1+\epsilon)$ if we can establish that $\frac{b_i(k)}{B_i(k)}$ is uniformly bounded above by ϵ for all i. There is a vast body of literature on polynomial-time approximation schemes for the knapsack problem [24]. For our problem, we suppose that we use one of these algorithms off the shelf.

D. Efficient use of wireless resources

We must note that in the solution proposed in Section III-B, we do not make the most efficient use of the wireless resources. This is due to the fact that we have considered an on-off framework, where videos are either served or stalled in a superframe. Thus, there might be superframes where the wireless resource is underutilized, even though videos are waiting to be served. This occurs because the basestation cannot completely serve any one of these stalled videos without exceeding capacity. However, this shortcoming can be easily addressed by simply allowing the variables $\{x_i(k)\}$ to take values from the interval $[0,1]$. That is to say, in superframe k, video i can be *partially* served by provisioning a bitrate of $x_i(k)b_i(k)$.

The deficit evolution and the proof of feasibility optimality stays the same. However, in view of superframes being served partially, one needs to be careful about keeping track of the current superframe's bitrate $b_i(k)$. In the case where $x_i(k)$ is restricted to be 0 or 1, the bitrate of video i in superframe k is simply the bitrate of the oldest superframe that has not yet been delivered. In the case where $x_i(k) \in [0,1]$, we adopt the following convention. The current bitrate is set to be the bitrate of the oldest superframe that has not yet been completely delivered. Thus, if video i is only partially served in superframe k, then the bitrate in superframe $k+1$ is set to be $b_i(k+1) = b_i(k)$. Further, one needs to keep track of what fraction of the current superframe remains to be served. Define $X_i(k)$ to be the remaining fraction of the current superframe for user i in superframe k. Thus, if video i is served partially in superframe k, then we set $X_i(k+1) = 1 - x_i(k)$.

Finally, if we suppose that superframes can be served partially, the QoE guarantee depends on how the playout buffer is emptied. For this, we mandate that a superframe is played out only after it is received completely. Thus the long term fraction of served superframes will be equal to the long term buffering ratio at the client side. The following theorem describes a feasibility optimal strategy for the case where $x_i(k) \in [0,1]$.

Theorem 3: Consider a policy which solves the following fractional knapsack problem at the beginning of the kth superframe.

$$\begin{aligned} \text{Maximize} \quad & \sum_{i \in S_k} \delta_i^+(k-1)x_i(k) \\ \text{s.t.} \quad & \sum_{i \in S_k} \frac{x_i(k)b_i(k)}{B_i(k)} \leq 1 \\ & 0 \leq x_i(k) \leq X_i(k) \end{aligned}$$

The above policy is feasibility optimal.

The proof of feasibility optimality follows exactly along the lines of Theorem 1. Further, we know that for the fractional knapsack problem, the optimal strategy is to pick videos in non-increasing order of $\frac{\delta_i^+(k)B_i(k)}{b_i(k)}$, until the constraint is met.

E. Low Bitrate Ratio as a QoE metric

In this section, we show that the proposed framework is flexible enough to admit other QoE metrics. In particular, we consider the scenario where the base-station can choose to serve parts of the video at a lower bitrate, and we consider the LBR metric described in Section III-A. Thus, one can simultaneously monitor two parameters namely the fraction of stalled frames and the fraction of frames served at low bit rate. The strategy of dynamically adapting the bitrate of the video is available as a feature in Adobe's dynamic streaming solutions or ByteMobile's platforms. Alternately, one could use more sophisticated techniques like Scalable Video Coding (SVC) [22] that allow on-the-fly transcoding to retrieve high and low bitrate versions from a single compressed video file.

As in Section III-D, we suppose that superframes can be served partially, and further, that the bitrate at which a video is served can be changed at a frame-level, thus resulting in a fraction of a superframe being served at the lower bitrate. Let $b_i(k)$ and $\tilde{b}_i(k)$ be the high and low bitrates of the two available versions of video i in superframe k. As before, we suppose that we know the maximum feasible bitrate $B_i(k)$ for user i in superframe k. Thus, we need $\frac{b_i(k)}{B_i(k)}$ fraction of slots if video i is served at high bitrate, and $\frac{\tilde{b}_i(k)}{B_i(k)}$ fraction of slots if video i is served at low bitrate. Let $x_i(k)$ be an indicator variable that is 1 if video i is served at either bitrate in superframe k, and 0 otherwise. Further, let $y_i(k)$ be an indicator variable that is 1 if video i is served at high bitrate in superframe k, and 0 otherwise.

QoE requirement: The revised QoE objective is to ensure that the fraction of stalled superframes for video i is less than α_i, and that the fraction of low bitrate superframes is less than β_i, i.e.,

$$\lim_{k\to\infty} \frac{1}{T_i(k)} \sum_{l=1}^{k} x_i(l)\mathbf{1}\{i \in S_l\} \geq (1-\alpha_i),$$
$$\lim_{k\to\infty} \frac{1}{T_i(k)} \sum_{l=1}^{k} y_i(l)\mathbf{1}\{i \in S_l\} \geq (1-\alpha_i-\beta_i)$$

for each i. In order to monitor the QoE requirement, we define two forms of debt for user i at the end of superframe k.

$$\delta_i(k) = (1-\alpha_i-\beta_i)T_i(k) - \sum_{l=1}^{k} y_i(l)\mathbf{1}\{i \in S_l\}$$
$$\tilde{\delta}_i(k) = (1-\alpha_i)T_i(k) - \sum_{l=1}^{k} x_i(l)\mathbf{1}\{i \in S_l\}$$

We now present the feasibility optimal scheduling policy which chooses the subset of videos to serve and their bitrates.

Theorem 4: Consider a scheduling policy that solves the following optimization problem at the beginning of the k^{th} superframe.

$$\begin{aligned}
\text{Maximize} \quad & \sum_{i\in S_k} \left(\delta_i^+(k-1) + \tilde{\delta}_i^+(k-1)\right) y_i(k) \\
& + \sum_{i\in S_k} \tilde{\delta}_i^+(k-1) v_i(k) \\
\text{s.t.} \quad & \sum_{i\in S_k} \frac{\tilde{b}_i(k) y_i(k)}{B_i(k)} + \sum_{i\in S_k} \frac{b_i(k) v_i(k)}{B_i(k)} \leq 1 \\
\text{where} \quad & 0 \leq y_i(k) \leq X_i(k), \quad 0 \leq v_i(k) \leq X_i(k), \\
& y_i(k) + v_i(k) \leq X_i(k),
\end{aligned}$$

where $X_i(k)$ is the remaining fraction of the current superframe for user i. The above policy is feasibility optimal for the frame-level granularity problem. Further, we propose a variation of the greedy algorithm that solves the above optimization problem.

Proof: The Lyapunov function is defined as follows:

$$L(k) = \frac{1}{2} \sum_{i=1}^{N} \left[\delta_i^2(k-1) + \tilde{\delta}_i^2(k-1)\right].$$

Proceeding exactly as in Theorem 2, we can derive

$$\Delta L(k) \leq N + \sum_{i\in S_k} \mathbf{E}\left[\delta_i(k-1)(1-\alpha_i-\beta_i-y_i(k)) + \tilde{\delta}_i(k-1)(1-\alpha_i-x_i(k))|S_k, [\delta_i(k-1)], [\tilde{\delta}(k-1)]\right].$$

As before, we can show that a scheduling policy which solves the following optimization at the beginning of the k^{th} superframe is feasibility optimal.

$$\begin{aligned}
\text{Maximize} \quad & \textstyle\sum_{i\in S_k} \delta_i^+(k-1) y_i(k) + \sum_{i\in S_k} \tilde{\delta}_i^+(k-1) x_i(k) \\
\text{s.t.} \quad & \textstyle\sum_{i\in S_k} \frac{(b_i(k)-\tilde{b}_i(k)) y_i(k)}{B_i(k)} + \frac{\tilde{b}_i(k) x_i(k)}{B_i(k)} \leq 1 \\
& 0 \leq x_i(k) \leq X_i(k), \quad 0 \leq y_i(k) \leq X_i(k)
\end{aligned}$$

We can write $x_i(k) = y_i(k) + v_i(k)$ where $v_i(k)$ is an indicator variable that is 1 if video i was served at low bitrate in superframe k. The optimization problem now becomes

$$\begin{aligned}
\text{Max} \quad & \sum_{i\in S_k} \left(\delta_i^+(k-1) + \tilde{\delta}_i^+(k-1)\right) y_i(k) + \sum_{i\in S_k} \tilde{\delta}_i^+(k-1) v_i(k) \\
\text{s.t.} \quad & \sum_{i\in S_k} \frac{b_i(k) y_i(k)}{B_i(k)} + \sum_{i\in S_k} \frac{\tilde{b}_i(k) v_i(k)}{B_i(k)} \leq 1 \\
& 0 \leq y_i(k) \leq X_i(k), 0 \leq v_i(k) \leq X_i(k), y_i(k) + v_i(k) \leq X_i(k).
\end{aligned}$$

This is a two-dimensional fractional knapsack problem which can be solved using Algorithm 1. □

F. Utility function based optimal policies

The QoE optimization techniques presented above provide QoE guarantees that are met in the long term. For instance, one can guarantee that the fraction of time a user spends stalled is less than α_i, but one has no control over where these stalls occur. For instance, given meta-information regarding the video content, there might be natural breaks in the video where a stall might be more tolerable than others. From the point of

Algorithm 1 Modified Greedy Algorithm for LBR ratio metric

$\theta \leftarrow 0$
Arrange the union of the following two index sequences in decreasing order

$$\left[\frac{B_i(k)\left(\delta_i^+(k-1)+\tilde{\delta}_i^+(k-1)\right)}{b_i(k)}\right] \quad \left[\frac{B_i(k)\tilde{\delta}_i^+(k-1)}{\tilde{b}_i(k)}\right]$$

while $\theta < 1$ **do**
  Select video with highest index
  if Selected video is high bitrate **then**
    Schedule the maximum feasible fraction of this video, say i

$$y_i(k) \leftarrow min\left(\frac{(1-\theta)*B_i(k)}{b_i(k)}, X_i(k)\right)$$

    $\theta \leftarrow \theta + \frac{y_i(k)b_i(k)}{B_i(k)}$
    if Low bitrate version has been scheduled earlier **then**
      $v_i(k) \leftarrow v_i(k) - y_i(k)$
      Delete the index entry corresponding to the high bitrate version of video i
    else
      Delete the index entries corresponding to both high and low bitrate versions of video i
    end if
  else
    Schedule the maximum feasible fraction of this video, say i

$$v_i(k) \leftarrow min\left(\frac{(1-\theta)*B_i(k)}{\tilde{b}_i(k)}, X_i(k)\right)$$

    $\theta \leftarrow \theta + \frac{v_i(k)\tilde{b}_i(k)}{B_i(k)}$
    Update the index and the bitrate of the high bitrate version of video i

$$J^{HBR}(i) \leftarrow \frac{B_i(k)\tilde{\delta}_i(k)}{b_i(k)-\tilde{b}_i(k)}; \quad b_i(k) \leftarrow b_i(k) - \tilde{b}_i(k)$$

    Delete the index entries corresponding to both high and low bitrate versions of video i
  end if
end while

view of improving QoE, it makes sense to incentivize stalling videos at these breaks. Thus, in addition to using the channel quality and the video bitrate, one can also take into account the state of play of each video, and derive a throughput-optimal scheduling policy that provides per-user QoE guarantees.

We propose to do this by asking user i to specify the utility, $U_i(k)$, of being served in the kth superframe. The video management system guarantees user i a long-term fraction $(1-\alpha_i)$ of the requested utility. It is the user's prerogative to adjust the *relative* utility of individual superframes corresponding to their importance. We present a debt-based scheduling policy that is feasibility optimal. To begin, we define the debt of user i at the end of superframe k as follows.

$$\delta_i(k) = (1-\alpha_i)\sum_{l:i\in S_l} U_i(l) - \sum_{l:i\in S_l} U_i(l)x_i(l)$$

Theorem 5: Suppose that the utilities specified by a user is constrained to be positive and bounded, with $0 \le U_i(l) \le C$. Consider the scheduling policy that solves the following optimization problem at the beginning of the kth superframe.

$$\begin{aligned} \text{Maximize} \quad & \textstyle\sum_{i\in S_k} \delta_i^+(k-1)U_i(k)x_i(k) \\ \text{s.t.} \quad & \textstyle\sum_{i\in S_k} \frac{b_i(k)x_i(k)}{B_i(k)} \le 1 \\ \text{where} \quad & 0 \le x_i(l) \le X_i(k) \end{aligned} \tag{4}$$

where $X_i(k)$ is the remaining fraction of the current superframe for user i. The above policy is feasibility optimal.

Proof: The Lyapunov function is chosen to be $L(k) = \frac{1}{2}\sum_{i=1}^{N}\delta_i^2(k-1)$. Proceeding exactly as before, we can derive

$$\begin{aligned} \Delta L(k) &= \sum_{i\in S_k}\frac{1}{2}U_i^2(1-\alpha_i-x_i(k))^2 \\ &+ \sum_{i\in S_k}\mathbf{E}\left[\delta_i(k-1)U_i(k)(1-\alpha_i-x_i(k))|S_k,[\delta_i(k-1)]\right] \\ &\le \frac{NC^2}{2} + \sum_{i\in S_k}\mathbf{E}\left[\delta_i(k-1)U_i(k)(1-\alpha_i-x_i(k))|S_k,[\delta_i(k-1)]\right]. \end{aligned}$$

Following the arguments presented in Theorem 2, we have that a scheduling policy which solves the optimization problem (4) at the beginning of the k^{th} superframe is feasibility optimal. □

Note that there is no advantage for a user to consistently over-quote or under-quote his utility, since the long-term average utility is pre-specified. One candidate scheme for assigning utilities is to set it to be one of two values, $\underline{U}$ or $\overline{U}$, where $\underline{U} \le \overline{U}$. The utility is nominally set to be $\overline{U}$ when the video is playing. However, when a stall occurs, the utility is dropped to $\overline{U}$ for a fixed number of subsequent frames, after which it is returned to the nominal value of $\overline{U}$. This is to encourage stalls to be bunched together, thus reducing the total number of interruptions.

IV. Simulation Study

We have conducted an extensive simulation study in MATLAB. The performance of the two-tiered architecture is compared with a standard base-station scheduler that implements weighted proportional fairness. We consider real video traces and study the perforrmance of deficit-based policies for both the buffering ratio and low bitrate ratio metrics. We begin with a detailed description of different aspects of the simulation framework, and then present the results under a variety of scenarios.

A. Simulation Framework

Video Traces: In our simulation study, we use video traces from the Arizona State University video trace library [25]. We have used single layer MPEG-2 video that is spatially and temporally scalable. The average bitrate of the videos we use varies from 175Kbps to 3.85Mbps. The frame rate of the videos is 30 frames per second. The videos are smoothed with a window size of 100 video frames, making the length of a superframe 3.3 seconds.

Mobility model: Users are positioned in a cell with a nominal cell side of 577m. At the beginning of each superframe, the distance of each user from the basestation is chosen to

be independent and uniformly distributed. This models the mobility of users within the cell

Channel Model: The channel is modeled using the COST Hata 231 path loss model. The path loss is given by

$$\begin{aligned}\text{Path loss(in dB)} &= (44.9 - 6.55\log_{10} h_{bs}) * \log_{10}(\frac{ss_{dist}}{1000}) \\ &+ 45.5 + (35.46 - 1.1h_{ss}) * \log_{10} f_c \\ &- 13.82 * \log_{10} h_b s + 0.7h_{ss} + C\end{aligned}$$

where $h_{bs} = 32m$ is the height of the basestation, ss_{dist} is the distance of the subscriber from the base-station, $h_{ss} = 1.5m$ is the height of the subscriber, $f_c = 2.5GHz$ is the carrier frequency and C is a parameter that is set to be 0dB for suburban and 3dB for urban environments [26].

Apart from the path loss, we also model the effect of shadowing. This shadowing loss (in dB) is modeled as a Gaussian random variable with a standard deviation of 8.9dB (see [26]). While we suppose that the path loss stays constant over the course of the entire superframe, the shadowing loss varies on a frame-by-frame basis. This is to model short-term user mobility.

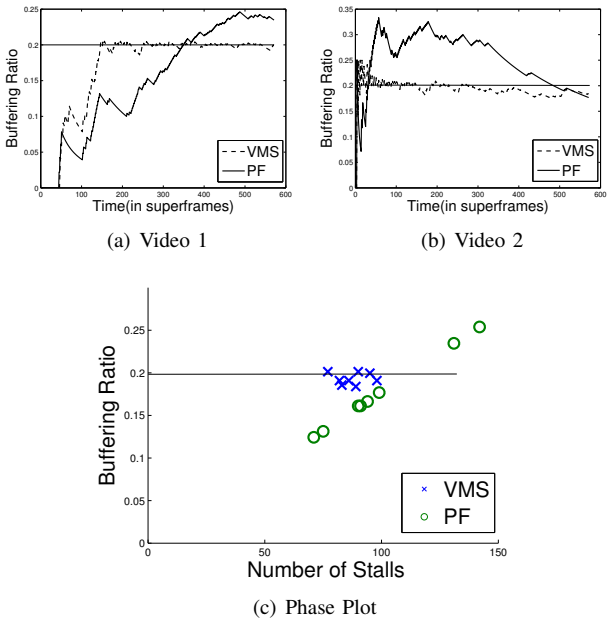
Scheduler The scheduler that we use as a baseline in the performance evaluation is the standard weighted Proportional Fair (PF) scheduler. In each slot, the scheduler picks the flow with the highest value of $\left(weight * \frac{\text{instantaneous rate}}{\text{average rate}}\right)$. Here the average rate is calculated as a moving average over the past 100 Wimax frames. The weight is typically chosen to be proportional to the required bitrate.

B. Experimental Results

1) Buffering ratio metric: Throughout this section, we consider 8 concurrent video flows at a basestation, and compare the performance of the VMS scheduler with the weighted PF scheduler. We begin by evaluating the performance of the VMS for videos of homogeneous bitrates and a uniform buffering ratio requirement of 0.2. Figures 2(a) and 2(b) show the time evolution of the buffering ratio for two candidate videos. We observe that the VMS scheduler maintains tight control over the buffering ratio, ensuring it is never much larger than 0.2. The standard PF scheduler on the other hand shows large variations in the buffering ratio. Figure 2(c) shows a phase plot of the buffering ratio versus number of stalls for all the eight videos. The VMS scheduler performs significantly better in terms of providing QoE guarantees for *each* user.

Next, we consider videos with heterogeneous bitrates and a uniform target buffering ratio of 0.2. Figures 3(a) and 3(b) show the evolution of buffering ratio for a candidate low bitrate video and a high bitrate video respectively. The VMS scheduler performs significantly better in terms of uniformly tracking the target QoE for each user. In contrast, the PF scheduler shows tremendous variations across videos as seen in the phase plot in Figure 3(c).

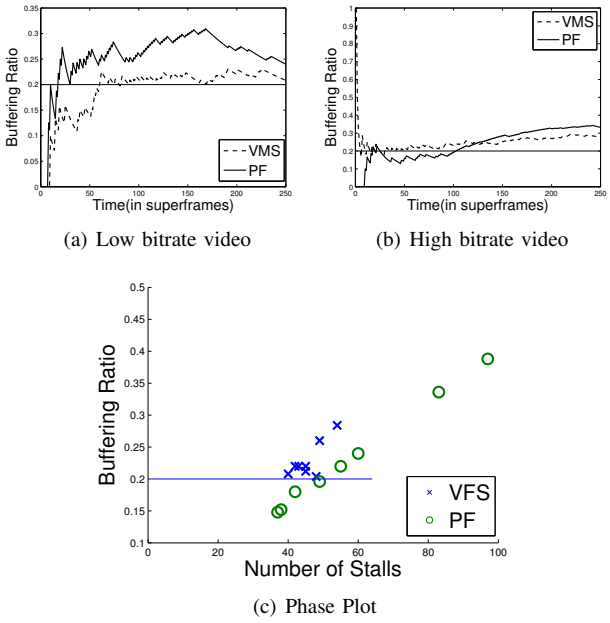
2) Buffering ratio and Low bitrate ratio metrics: We now consider a combination of the buffering ratio and low bitrate ratio metrics. We first consider videos with homogeneous bitrates and a uniform requirement given by $\alpha_i = 0.1, \beta_i = 0.1$.



(a) Video 1 (b) Video 2

(c) Phase Plot

Fig. 2. BR metric, Homogeneous Bitrates



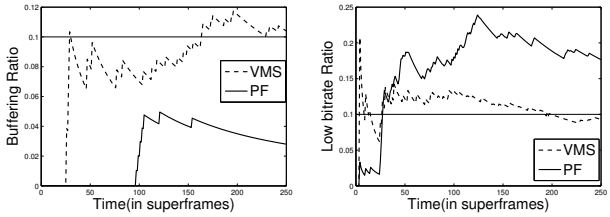
(a) Low bitrate video (b) High bitrate video

(c) Phase Plot

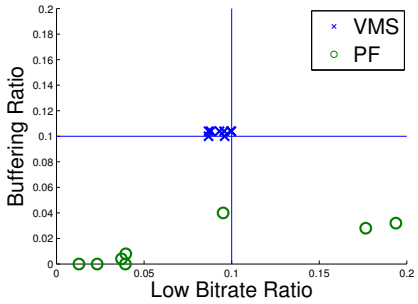
Fig. 3. BR metric, Heterogeneous Bitrates

Figures 4(a) and 4(b) show the evolution of buffering ratio and low bitrate ratio for a candidate video. The VMS scheduler provides good tracking of both QoE metrics while the PF scheduler has little control over the low bitrate ratio. In Figure 4(c), the phase plot of buffering ratio versus the low bitrate ratio shows that the VMS scheduler perfectly maintains the QoE guarantee uniformly for all users. In contrast, the PF scheduler shows lower buffering ratios but highly varying low bitrate ratios.

Finally, we consider videos with heterogeneous bitrates and uniform $\alpha_i = 0.1$ and $\beta_i = 0.15$. Figure 5 shows the phase



(a) Buffering Ratio (b) Low Bitrate Ratio



(c) Phase Plot

Fig. 4. Two QoE metrics, Homogeneous Bitrates

plot of buffering ratio versus low bitrate ratio. The VMS scheduler strictly maintains the low bitrate ratio guarantee while the buffering ratios are maintained as close to 0.1 as possible. In contrast, the PF scheduler shows large variations and does not satisfy either QoE metric uniformly across users.

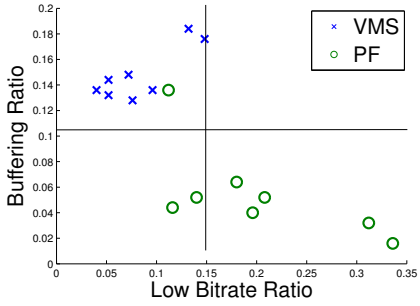


Fig. 5. Two QoE metrics, Heterogeneous Bitrates

V. Concluding Remarks

We have presented a two-tiered architecture for guaranteeing QoE of video delivery in cellular networks. We devised a VMS scheduler that works at the timescale at which user experience is evaluated. We derived feasibility optimal policies that seek to provide long-term QoE guarantees to each user, irrespective of individual bitrates and varying channel conditions. The proposed framework is versatile, admitting a variety of QoE metrics and showing significant improvements in simulations. Several interesting problems remain to be resolved. We need to understand the tradeoffs involved while simultaneously guaranteeing multiple QoE metrics. In addition, we need to study how utility functions may be constructed so as to best quantify user satisfaction.

References

[1] Cisco, "Visual Networking Index: Global mobile data traffic forecast update, 2011-2016."
[2] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, and H. Zhang, "Understanding the impact of video quality on user engagement," in *SIGCOMM 2011*, 2011, pp. 362–373.
[3] S. S. Lam, S. Chow, and D. K. Y. Yau, "An algorithm for lossless smoothing of MPEG video," *ACM SIGCOMM CCR*, vol. 24, no. 4, 1994.
[4] T. Ott, T. V. Lakshman, and A. Tabatabai, "A scheme for smoothing delay-sensitive traffic offered to ATM networks," in *INFOCOM*, 1992.
[5] N. B. Shroff and M. Schwartz, "Video modeling within networks using deterministic smoothing at the source," in *INFOCOM*, 1994.
[6] S. Sen, J. Dey, J. Kurose, J. Stankovic, and D. Towsley, "Streaming CBR transmission of VBR stored video," in *SPIE Symposium on Voice Video and Data Communications*, 1997.
[7] J. D. Salehi, S.-L. Zhang, J. Kurose, and D. Towsley, "Supporting stored video: reducing rate variability and end-to-end resource requirements through optimal smoothing," *ACM Trans. Netw.*, vol. 6, no. 4, 1998.
[8] H. Fattah and C. Leung, "An overview of scheduling algorithms in wireless multimedia networks," *Wireless Communications, IEEE [see also IEEE Personal Communications]*, vol. 9, no. 5, pp. 76–83, 2002.
[9] Y. Cao and V. O. K. Li, "Scheduling algorithms in broad-band wireless networks," *Proceedings of the IEEE*, vol. 89, pp. 76–87, January 2001.
[10] R. Jain, "Scheduling in ieee 802.16e mobile wimax networks: Key issues and a survey," *IEEE Journal on Selected Areas in Communication*, vol. 27, no. 2, 2009.
[11] A. Grilo, M. Macedo, and M. Nunes, "A scheduling algorithm for qos support in ieee802.11 networks," *Wireless Commun.*, vol. 10, no. 3, pp. 36–43, Jun. 2003.
[12] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, and P. Whiting, "Providing quality of service over a shared wireless link," *IEEE Communications Magazine*, vol. 39, no. 2, pp. 150–154, 2001.
[13] L. Tassiulas, R. Tassiulas, and A. Ephremides, "Dynamic server allocation to parallel queues with randomly varying connectivity," *IEEE Transactions on Information Theory*, vol. 39, pp. 466–478, 1993.
[14] M. Neely, "Delay analysis for max weight opportunistic scheduling in wireless systems," in *Proc. 46th Annual Allerton Conf. on Communication, Control, and Computing*, September 2008.
[15] I.-H. Hou and P. R. Kumar, "Scheduling heterogeneous real-time traffic over fading wireless channels," in *Proceedings of the 29th conference on Information communications*, ser. INFOCOM'10, 2010, pp. 2606–2614.
[16] G. Liang and B. Liang, "Effect of delay and buffering on jitter-free streaming over random VBR channels," *IEEE Transactions on Multimedia*, vol. 10, no. 6, 2008.
[17] ——, "Balancing interruption frequency and buffering penalties in VBR video streaming," in *INFOCOM*, 2007.
[18] C.-H. Hsu and M. Hefeeda, "On statistical multiplexing of variable-bit-rate video streams in mobile systems," in *ACM Multimedia*, 2009.
[19] P. Dutta, A. Seetharam, V. Arya, M. Chetlur, S. Kalyanaraman, and J. Kurose, "On managing quality of experience of multiple video streams in wireless networks," *INFOCOM 2012*.
[20] J. Huang, C. Krasic, J. Walpole, and W. chi Feng, "Adaptive live video streaming by priority drop," in *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2003.
[21] H. Radha, M. van der Schaar, and Y. Chen, "The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP," *IEEE Transactions on Multimedia*, vol. 3, no. 1, 2001.
[22] 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 Techn.*, vol. 17, no. 9, 2007.
[23] G.-M. Su and M. Wu, "Efficient bandwidth resource allocation for low-delay multiuser video streaming," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 15, no. 9, 2005.
[24] V. V. Vazirani, *Approximation Algorithms*. Springer, Mar. 2004.
[25] P. Seeling and M. Reisslein, "Video transport evaluation with H.264 video traces," *IEEE Communications Surveys and Tutorials, in print*, 2012, Traces available at `trace.eas.asu.edu`.
[26] "Wimax system evaluation methodology doc v2.1," Jul.