Optimizing Stored Video Delivery For Mobile Networks: The Value of Knowing the Future

Zheng Lu and Gustavo de Veciana
Department of Electrical and Computer Engineering
University of Texas at Austin

***Abstract*—This paper considers the design of cross-layer opportunistic transport for stored video over wireless networks with a slow varying (average) capacity. We focus on two key ideas: (1) scheduling data transmissions when capacity is high; and (2), exploiting knowledge of *future* capacity variations. The latter is possible when users' mobility is known or predictable, e.g., users riding on public transportation or using navigation systems. We consider the design of cross-layer transmission schedules which minimize system utilization (and thus possibly transmit/receive energy) while avoiding, if at all possible, rebuffering/delays, in several scenarios. For the single-user anticipative case where all future capacity variations are known beforehand; we establish the optimal transmission schedule is a Generalized Piecewise Constant Thresholding (GPCT) scheme. For the single-user partially anticipative case where only a finite window of future capacity variations is known, we propose an online Greedy Fixed Horizon Control (GFHC). An upper bound on the competitive ratio of GFHC and GPCT is established showing how performance loss depends on the window size, receiver playback buffer, and capacity variability. Finally we consider the multiuser case where we can exploit both future temporal and multiuser diversity. Our simulations and evaluation based on a measured wireless capacity trace exhibit robust potential gains for our proposed transmission schemes.**

I. Introduction

Video delivery over wireless networks is expected to grow quickly in the next few years. Recent studies (see [1]) show that mobile data traffic will increase 25-fold between 2011 and 2016 and about two-thirds of the traffic will be video including real-time video, video on demand (VoD), video conferencing etc. The wireless infrastructure can hardly keep pace with such growth, thus it is important to make effective use of the available wireless resources in video delivery.

Even the successor to current cellular systems, 4G broadband, promises not only improvements in overall capacity but also, unfortunately, higher degrees of capacity variability, particularly in the case of mobile users. Our premise in this paper, is that approaches can be devised that exploit such capacity variations, and the nature of the underlying services. Indeed, mobile devices are increasingly equipped with video playback buffers, giving more flexibility in exploiting capacity variations without interrupting playback.

In this paper we design of application-layer opportunistic transport for stored video over wireless networks with a slow varying (average) capacity. We focus on two key ideas: (1) scheduling data transmission when capacity is high; and (2), exploiting knowledge of *future* capacity variations. The latter is possible when users' future locations are known, which can in turn be used to infer their future wireless coverage/capacity. For example this is the case for users on public transportation buses/trains, or others using navigation systems in their cars. In fact, even without prior knowledge of vehicle routes, one can still infer future vehicle's mobility. Indeed [2] demonstrate the effectiveness on real data of "K Nearest Trajectories" an algorithm to predict future capacity variations for vehicles. More generally, humans' mobility patterns tend to be highly predictable, [3] show a potential 93% average 'predictability' suggesting knowing the future (in this regard) is quite reasonable.

Let us consider a simple example. Suppose a server is delivering a constant-bit-rate stored video to a mobile user. The length of the video is 4 seconds and the streaming rate is 200kbps, thus the size of the video is 800kb. Suppose the capacity variation is as shown in Fig. 1. If the video is delivered at a fixed rate of roughly 200kbps then two of the four slots are partially utilized, leading to a 75% system utilization. However if the video is delivered greedily, i.e., at the full available capacity, then transmission completes in 2.5secs, resulting in utilization of 62.5%. However, it is easy to see that an optimal schedule would send at a full rate in Slots 1 and 3, and transmit nothing in Slots 2 and 4, which guarantees smooth video playback and results in a minimal utilization of 50%. This scheme transmits when channel conditions are good, i.e., we set a threshold and transmit only when the capacity is above the threshold. The threshold choice, however, depends on the future capacity variations and video playback requirements. We will call this a 'thresholding scheme' and define it formally in the sequel.

Related Work. There has been a substantial body of literature on stored video delivery. Below we focus on a few key works that are akin, in terms of methodology but differ terms of their objectives. Indeed to our knowledge there is no prior work on devising transport protocols that exploiting (future) capacity variations. For example, a piecewise constant-rate transmission scheme was developed in [4], wherein dynamic programming was applied to find the optimal schedules for a variety of optimization criteria, in particular minimizing the maximum transmission rate subject to a maximum initial delay, minimizing the maximum transmission rate subject to a maximum temporal-jump delay, and minimizing the average

This research was supported in part by Intel and Cisco under the VAWN program.

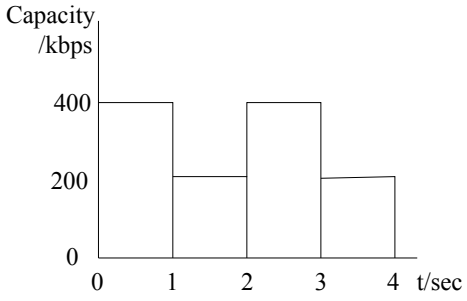

Fig. 1. The channel variations in four time slots. Transmitting the video at peak capacity in the first and third slots and transmitting nothing in the second and fourth slots, gives the lowest system utilization.

temporal-jump delay subject to a constraint on the maximum transmission rate. Similarly [5] suggest several ways to reduce transmission rate variability for stored video delivery and propose an optimal smoothing scheme which is further explored in [6], [7]. However, if one or more users will see highly variable wireless capacity, it should be clear that smoothing transmissions may not be the right objective; indeed, 'bursty' transmissions might be preferable. [8] consider a scenario where video streams share a source server. They propose a centralized prefetching protocol which schedules transmissions to users whose playback buffer queues are shortest. This does not *explicitly* account for capacity variations – present or future. The work in [9], proposes a decentralized protocol targeting a scenario where streams from multiple video servers share a multiplexer and dynamically adjust their transmit windows in a window flow control mechanism to mitigate packet loss. The key differentiating element of our work is a formal investigation of how knowledge of future capacity variations could be used towards reducing utilization (increasing capacity) while minimizing video rebuffering.

Contributions and Organization. This paper proposes a new class of cross-layer transmission schedules which minimize system utilization (and thus possibly transmit/receive energy) while avoiding, if at all possible, rebuffering/delays. In Section II we study the single-user anticipative case where all future capacity variations are known beforehand; we formally establish the optimal transmission schedule is a Generalized Piecewise Constant Thresholding (GPCT) scheme. In Section III we consider the single-user partially anticipative case where only a finite window of future capacity variations is known, we propose an online Greedy Fixed Horizon Control (GFHC). An upper bound on the competitive ratio of GFHC and GPCT is established clearly indicating how performance loss depends on the window size, receiver playback buffer, and capacity variability. Finally in Section IV, we consider the multi-user anticipative case, and we develop two multiuser schemes based on GPCT, which are suboptimal, but straightforward to implement, and able to achieve good performance. Simulations described in Section V explore the performance gains achievable for a typical scenario and explore the impact of correlation in capacity variations on these gains. Our simulations show the potential gains for such opportunistic transmission schemes exhibit an up to 70% reduction in the system utilization. Section VI briefly concludes the paper.

II. Single-User Anticipative Case

We will first consider the anticipative case where a server is delivering video content to a single user and future variations in wireless capacity are known beforehand.

A. Model Formulation

Consider a video server streaming a stored video to a mobile user via one (or more) base station(s). Suppose the wireless part is the bottleneck so that the application layer throughput mainly depends on the wireless capacity. Further let us focus on slow variations in wireless capacity, e.g., on the order of secs, so that timely end-to-end feedback actions can be realized before the capacity changes too much. Let $c(t)$ be the average[1]of the peak capacity at time t, and $r(t)$ be the actual transmission schedule such that $0 \leq r(t) \leq c(t)$. Then $r(t)/c(t)$ can be roughly regarded as the system utilization at time t. Let $s(t)$ be the cumulative amount of data sent and received[2], i.e. $s(t) = \int_0^t r(\tau)d\tau$.

Now suppose the video to be transmitted has a finite length T (seconds), a finite size S (bits), and define two functions $l(\cdot)$ and $u(\cdot)$ associated with requirements on the video's transmission. Suppose the transmission begins at time 0 and no interruptions (rebuffering) happen during the transmission. Let $l(t)$ denote the cumulative data consumed if the user watches the first t secs of the video, where $t \in [0, T]$. Note that if the user's playback buffer has finite size, s/he can only receive a limited amount of data before viewing it. We define $u(t)$ to be the maximum cumulative amount of data that can be received by the user over $[0, t]$, where $t \in [0, T]$. Further we assume $l(\cdot)$ and $u(\cdot)$ are both nondecreasing piecewise constant and right continuous functions as depicted in Fig. 2, where jumps happen at times $t_0, t_1, t_2, ..., t_n$. The jumps might correspond to individual (or groups of related, e.g., intra-coded/predicted) frames being displayed and which are no longer necessary to reconstruct future content. Note that the jump points are chosen such that $t_0 = 0$, $t_n = T$, and $t_0 \leq t_1 \leq t_2 \leq ... \leq t_n$. Also note that it is possible for $t_i = t_{i+1}$ where t_i is a jump point of $u(\cdot)$ and t_{i+1} is a jump point of $l(\cdot)$. Further let $I_l = \{i : t_i \text{ is a jump point of } l(\cdot)\}$ and $I_u = \{i : t_i \text{ is a jump point of } u(\cdot)\}$ denote the sets of jump point indices for the two piecewise constant functions. Note that if the client has a fixed playback buffer size b, then there is a vertical gap of size b between $u(\cdot)$ and $l(\cdot)$. However Fig. 2 shows a more general case where there may be time varying buffer allocations for playback.

Next, we define the cost function in our model. The cost is a sum of two terms: the average system utilization $cost_u$ and the rebuffering time $cost_r$. Assuming no rebuffering, the

[1]The average is taken over periods on the order of seconds to smooth out fast capacity variations.

[2]We assume that transport exploits playback buffering and thus can be made reliable.

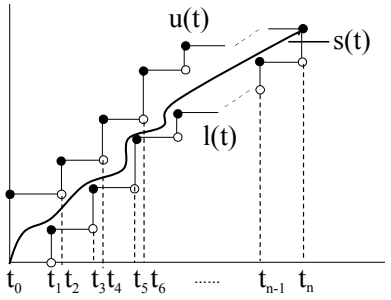

Fig. 2. The piecewise constant functions $l(\cdot)$, $u(\cdot)$ and the cumulatively transmitted data $s(\cdot)$. $l(t)$ is the cumulative amount of data consumed (i.e. watched) by the user over $[t_0, t]$. $u(t)$ is the maximum cumulative amount of data that can be received by the user over $[t_0, t]$. Jumps happen at times $t_1, t_2, t_3, ..., t_n$. $s(\cdot)$ lies between $l(\cdot)$ and $u(\cdot)$ if there are no playback buffer underflow and overflow.

average system utilization during video watch period $[0, T]$ can be defined as:

$$cost_u = \frac{1}{T}\int_0^\infty \frac{r(t)}{c(t)}dt.$$

Note we assume $r(t) = 0$ after the transmission finishes, so the above integration actually has a finite time horizon. We use the above cost function versus $\int_0^T r(t)dt / \int_0^T c(t)dt$ because the former properly captures the reduction in utilization in a system with time varying capacity using opportunistic 'scheduling.'

The rebuffering cost depends on the waiting time a user experiences when buffer underflow occurs during video playback. We assume that playback buffer underflows can only happen at the jump points for $l(\cdot)$, i.e., times in I_l. This is a reasonable assumption since the video playback can be paused only after an entire frame is displayed. Further we denote the associated waiting times as $\tau_0, \tau_1, ..., \tau_n$. Note the functions $u(\cdot)$, $l(\cdot)$ and the jump points in Fig. 2 are known before transmission, and thus they do not take into account delays due to rebuffering times. In other words, $t_0, t_1, t_2, ..., t_n$ are fixed constants before transmission, which may be shifted by $\tau_0, \tau_1, ..., \tau_n$, which are variables whose values depend on the actual wireless capacities and rebuffering during transmission.

Our rebuffering cost is defined as

$$cost_r = a\sum_{i=0}^{n}\tau_i,$$

where a is a constant. In practice it is natural to put a higher priority on minimizing rebuffering over system utilization. Thus the constant a should be large enough such that one cannot obtain a lower total cost by increasing rebuffering time to reduce system utilization. Suppose we know upper and lower bounds on the wireless capacity, denoted c_{max} and $c_{min} > 0$. Then adding a waiting time τ can result in a maximum reduction in system utilization of no more than $\frac{\tau(c_{max}-c_{min})}{Tc_{min}}$, which is obtained by assuming video content of size $c_{max}\tau$ is transmitted at rate c_{max} during the waiting time τ instead of being transmitted at rate c_{min} during a period of $\frac{c_{max}\tau}{c_{min}}$, which results in a reduction of $\frac{c_{max}\tau}{c_{min}T} - \frac{\tau}{T} = \frac{\tau(c_{max}-c_{min})}{Tc_{min}}$ in system utilization. Thus if we set $a = \frac{c_{max}-c_{min}}{Tc_{min}}$, we achieve our goal of strictly prioritizing minimization of rebuffering cost over system utilization.

Also note that if we assume $c_{min} > 0$, the normalization factor T in $cost_u$ can be replaced by $\tilde{T} = \max[\frac{S}{c_{min}}, T]$ ensuring the utilization cost remains below 1 without changing the overall character of the objective function. Correspondingly we choose $a = \frac{c_{max}-c_{min}}{\tilde{T}c_{min}}$ to prioritize minimization of rebuffering. We will use these from now on.

When minimizing our cost, we are constrained to ensure that the playback buffer does not drop below 0 or exceed the available buffer. This can be captured by the following constraints:

$$\int_0^{t_i+\sum_{k=0}^{i}\tau_k} r(t)dt + b_0 \geq l(t_i),\ \forall i \in I_l$$
$$\int_0^{t_i+\sum_{k=0}^{i}\tau_k} r(t)dt + b_0 \leq u(t_i),\ \forall i \in I_u$$

where b_0 is the initial buffer content (in bits) that is not accounted in the transmission schedule (often, it is 0). Note the upper bound in the above integration intervals is the current total time $t_i + \sum_{k=0}^{i}\tau_k$ which consists of the current time of the video t_i and the cumulative rebuffering time $\sum_{k=0}^{i}\tau_k$. We call these buffer underflow and overflow constraints respectively. Also we have the terminal condition:

$$\int_0^{t_n+\sum_{k=0}^{n}\tau_k} r(t)dt + b_0 = l(t_n),$$

which captures the fact that the video transmission must finish prior to final video playback.

Letting $\vec{\tau} = (\tau_0, \tau_1, ..., \tau_n)$ and $r(\cdot)$ denote the rebuffering times and video transmission schedule, we summarize our overall goal in terms of the following optimization problem:

Optimal Streaming Problem

$$\min_{r(\cdot),\vec{\tau}}\ \frac{1}{\tilde{T}}\int_0^\infty \frac{r(t)}{c(t)}dt + \frac{c_{max}-c_{min}}{\tilde{T}c_{min}}\sum_{i=0}^{n}\tau_i, \tag{1}$$
$$\text{s. t.}\ \int_0^{t_i+\sum_{k=0}^{i}\tau_k} r(t)dt + b_0 \geq l(t_i),\ \forall i \in I_l,$$
$$\int_0^{t_i+\sum_{k=0}^{i}\tau_k} r(t)dt + b_0 \leq u(t_i),\ \forall i \in I_u,$$
$$\int_0^{t_n+\sum_{k=0}^{n}\tau_k} r(t)dt + b_0 = l(t_n).$$

We say the above optimization problem is a video delivery optimization with initial state b_0 and terminal state $l(t_n)$. Note this problem is not convex since the constraints are not convex. However it always has a feasible solution if $c_{min} > 0$. In the sequel, we will first deal with the simpler situation where the optimization has a feasible solution without rebuffering, i.e., $\vec{\tau} = \vec{0}$. Subsequently we generalize the solution to situations where rebuffering is necessary.

B. Piecewise Constant Thresholding (PCT) Algorithm Under No Rebuffering Assumption

Assume $c(\cdot)$ is such that there is a feasible solution to Optimal Streaming (1) without rebuffering, i.e., $\vec{\tau} = \vec{0}$. Note in the model above, the constant a is chosen large enough such that $cost_r$ dominates $cost_u$ in the sense that we cannot achieve a lower cost by adding rebuffering time. Thus under the no rebuffering assumption, Optimal Streaming (1) is equivalent to the Min Utilization (2) given below.

Min Utilization Problem

$$\begin{aligned} \min_{r(\cdot)} \ & \frac{1}{\bar{T}} \int_0^{t_n} \frac{r(t)}{c(t)} dt, \qquad (2) \\ \text{s. t.} \ & \int_0^{t_i} r(t)dt + b_0 \geq l(t_i), \ \forall i \in I_l, \\ & \int_0^{t_i} r(t)dt + b_0 \leq u(t_i), \ \forall i \in I_u, \\ & \int_0^{t_n} r(t)dt + b_0 = l(t_n). \end{aligned}$$

In this subsection we determine delivery schedules that solve this problem, i.e., schedules $r(\cdot)$ achieving a minimum utilization while ensuring the cumulative data $s(\cdot)$ lies between $u(\cdot)$ and $l(\cdot)$ as shown in Fig. 2. Before introducing our algorithm let us define some terminology.

***Definition** 1:* A *single threshold transmission scheme* on an interval $[t_s, t_e]$ with initial state $s(t_s)$ and terminal state $s(t_e)$ is such that for $t \in [t_s, t_e]$:

$$r(t) = \begin{cases} c(t) & \text{if } c(t) > \alpha \text{ or } c(t) = \alpha, t \leq \tau \\ 0 & \text{if } c(t) < \alpha \text{ or } c(t) = \alpha, t > \tau \end{cases}$$

where $\alpha \in [0, \max_{t\in[t_s,t_e]} c(t)]$, and $\tau \in [t_s, t_e]$ are thresholds such that:

$$\int_{t_s}^{t_e} r(t)dt = s(t_e) - s(t_s).$$

Further, we denote by $\beta_{\alpha,\tau,t_s}(t) = \int_{t_s}^{t} r(\tau)d\tau + s(t_s)$ the cumulative amount of transmitted data for $t \in [t_s, t_e]$.

Note we refer to this as a "single" threshold scheme although in fact it is a pair: α is a threshold on wireless capacity, and τ is a threshold on time. Basically the scheme transmits data only when the capacity is above the threshold α. Thus continuously decreasing α will increase the cumulative amount of data transmitted with a potential for jumps if the capacity stays constant at some levels. By varying τ we can further control transmission so that the cumulative data transmitted varies continuously over its range.

The goal of the single threshold transmission scheme is to find α and τ, such that given an initial state $s(t_s)$ and wireless capacity $c(t)$, $t \in [t_s, t_e]$, the terminal state $s(t_e)$ is achieved. The thresholds can be computed by using binary search algorithm. However in practice, we can assume that the capacity $c(\cdot)$ is a piecewise constant function, i.e., we can use a discrete-time system model, in which case we assume the interval $[t_s, t_e]$ is divided into m slots and the capacity function is constant on each slot. We denote the ith slot as $[p_{i-1}, p_i]$, $i = 1, 2, ..., m$ and denote the associated capacity as c_i. Under these assumptions, the thresholds of the single threshold transmission scheme can be found by using a sorting algorithm, which provides a unique one-to-one mapping f: $\{1, 2, ..., m\} \to \{1, 2, ..., m\}$ such that $f(i) < f(j)$ if and only if $c_i > c_j$ or $c_i = c_j, i < j$. Note this mapping sorts the capacities of the slots in descending order. Then we can sum the sorted capacities up from the highest value to the lowest value, such that the sum exceeds $s(t_e) - s(t_s)$. Suppose this happens after we add the kth sorted slot, and the sum exceeds $s(t_e) - s(t_s)$ by s. Then the thresholds are $\alpha = c_{f^{-1}(k)}$ and $\tau = p_{f^{-1}(k)} - (p_{f^{-1}(k)} - p_{f^{-1}(k)-1})\frac{s}{c_{f^{-1}(k)}}$.

Also note that under a single threshold transmission scheme, the server only has two choices: send at peak capacity or send nothing. Hence, the corresponding utilization is proportional to the length of time the server is sending data.

We define two types of constraint violations associated with Min Utilization (2). We refer to a buffer underflow violation when one of the first set of constraints (buffer underflow constraints) in Min Utilization is not met, and we refer to buffer overflow violation when one of the second set of constraints (buffer overflow constraints) is not met.

Let $u_{[a,b]}$, $l_{[a,b]}$, $c_{[a,b]}$, $r_{[a,b]}$ and $\beta_{\alpha,\tau,t_s,[a,b]}$ denote the values of the functions $u(\cdot)$, $l(\cdot)$, $c(\cdot)$, $r(\cdot)$ and $\beta_{\alpha,\tau,t_s}(\cdot)$ on the interval $[a, b]$ respectively. Then the optimal transmission schedule for (2), $r^\star_{[t_0,t_n]}$ can be calculated using Piecewise Constant Thresholding (PCT) algorithm (Algorithm 1) with initial state $s_s = b_0$ and terminal state $s_e = l(t_n)$.

Algorithm 1 Piecewise Constant Thresholding (PCT)

Input: s_s, s_e, $l_{[t_0,t_n]}$, $u_{[t_0,t_n]}$, $c_{[t_0,t_n]}$
1: $m \leftarrow 0$, $r^\star_{[t_0,t_n]} \leftarrow 0$
2: $t_s \leftarrow t_0$, $t_e \leftarrow t_n$
3: **repeat**
4: $(t_b, s_b, r^\star_{[t_s,t_b)}) =$ **Breakpoint**$(s_s, s_e, t_s, t_e, l_{[t_s,t_e]}, u_{[t_s,t_e]}, c_{[t_s,t_e]})$
5: $m \leftarrow m + 1$
6: $s_s \leftarrow s_b$, $t_s \leftarrow t_b$
7: **until** $s_s = s_e$
Output: $r^\star_{[t_0,t_n]}$, m

The logic underlying PCT is as follows. First try to apply the single threshold scheme on the interval $[t_0, t_n]$ with initial state s_s and terminal state s_e. If the resulting transmission schedule $r_{[t_0,t_n]}$ meets the buffer overflow and underflow constraints, then it is the final solution; we call it a 1-piecewise constant thresholding solution. Otherwise if any of the constraints is violated, divide the time interval $[t_0, t_n]$ into two subintervals $[t_0, t_b)$ and $[t_b, t_n]$ at some point t_b (we call it a breakpoint), which is carefully chosen such that we can once again run the single threshold transmission scheme on $[t_0, t_b)$ to obtain a feasible solution which is output as the solution on subinterval

Algorithm 2 Breakpoint

Input: s_s, s_e, t_s, t_e, $l_{[t_s,t_e]}$, $u_{[t_s,t_e]}$, $c_{[t_s,t_e]}$

loop
  Apply single threshold transmission scheme on $[t_s, t_e]$ with initial state s_s and terminal state s_e to get α, τ and $r_{[t_s,t_e]}, \beta_{\alpha,\tau,t_s,[t_s,t_e]}$.
  if $\beta_{\alpha,\tau,t_s,[t_s,t_e]}$ does not violate any buffer underflow or overflow constraints **then**
    $t_b \leftarrow t_e$, $s_b \leftarrow \beta_{\alpha,\tau,t_s}(t_e)$
    break
  else
    find the largest i, $t_s \leq t_i \leq t_e$, such that the violations on $[t_s, t_i]$ are of the same type
    if the violation type is buffer overflow **then**
      $s_e \leftarrow u(t_i)$, $t_e \leftarrow t_i$
    else
      $s_e \leftarrow l(t_i)$, $t_e \leftarrow t_i$
    end if
  end if
end loop

Output: t_b, s_b, $r_{[t_s,t_e)}$

$[t_0, t_b)$. Then the remaining subinterval $[t_b, t_n]$ is treated as a new interval and we apply the same procedure to it as on $[t_0, t_n]$. This recursive procedure is realized by a repeat loop in Algorithm 1, where in each loop, the function "**Breakpoint**" is called to calculate the breakpoint t_b, and the transmission schedule $r(t)$ for $t \leq t_b$. The function "**Breakpoint**" is described in Algorithm 2, where we use a loop to find the breakpoint, and in each loop the single threshold transmission scheme described in Definition 1 is used. Finally if Algorithm 1 takes m loops to finish, then we end up with m subintervals and we call the solution an m-piecewise constant thresholding solution.

The following theorem states that PCT provides an optimal solution for Min Utilization (2).

***Theorem** 1:* If there exists a feasible solution to Min Utilization (2), then PCT determines an optimal transmission schedule.

The proof of Theorem 1 can be found in [10]. For succinctness we leave it out, but the general idea is simply to recognize that when the single threshold transmission scheme is not feasible, then the optimal transmission will have to break up the transmission schedule into subintervals where fixed thresholds will be used.

C. General Piecewise Constant Thresholding (GPCT)

In this subsection, we generalize PCT to solve Optimal Streaming (1), i.e., including the possibility of rebuffering. By the results in Subsection II.B, we immediately have the following corollary:

***Corollary** 1:* If there exists a feasible solution to Optimal Streaming (1) with $\vec{\tau} = \vec{0}$, then PCT solves the optimization.

However, if Optimal Streaming (1) does not allow a feasible solution with $\vec{\tau} = \vec{0}$, then rebuffering must happen and PCT cannot be used directly. Instead we require a modification of PCT to deal with the rebuffering issue. Before we state the new algorithm, we introduce some definitions.

***Definition** 2:* We say a transmission scheme is **greedy** if it sends as much as possible given the capacity and playback buffer constraints, i.e., the greedy transmission scheme tries to keep the buffer full. During the transmission the video plays as long as the playback buffer is not empty and rebuffering happens only when the buffer underflows. Consider a greedy transmission scheme starting at time t_1, and for which rebuffering happens at time t_2 where $t_2 > t_1$. We say this rebuffering is an **I-rebuffering** if 100% utilization was achieved during (t_1, t_2). Otherwise, if the greedy schedule on (t_1, t_2) was precluded from realizing 100% utilization, i.e., due to a limited playback buffer, we call the rebuffering a **B-rebuffering**.

Note if the playback buffer size is infinite (or at least larger than the video size), then there can only be I-rebufferings. While B-rebufferings are caused by playback buffer overflows. During a video transmission, both I-rebufferings and B-rebufferings can happen. Suppose they happen at the jump points $t_{n_1}, t_{n_2}, ..., t_{n_k}$, denote the corresponding rebuffering times as $\tau_{n_1}, \tau_{n_2}, ..., \tau_{n_k}$, and define $d(n_i) = \sum_{j=1}^{i} \tau_{n_j}$, the cumulative rebuffering time up to time t_{n_i}.

With the above definitions we can state our solution to Optimal Streaming (1) as General Piecewise Constant Thresholding (GPCT) (Algorithm 3) with initial state $s_s = b_0$ and terminal state $s_e = l(t_n)$.

In GPCT, we first use greedy transmission scheme to find where the rebufferings need to happen and identify all the associated rebuffering types. If no rebuffering happens, then the algorithm degenerates to PCT, and Corollary 1 ensures optimality. Otherwise if an I-rebuffering occurs, we have to use greedy transmission scheme to minimize the waiting time since our rebuffering cost dominates the utilization cost. However if a B-rebuffering happens, we can use PCT before the latest playback buffer overflow, which is denoted as $\tilde{t}(n_i)$ in the algorithm, to achieve the minimum system utilization. Thus we have the following theorem.

***Theorem** 2:* The General Piecewise Constant Thresholding (GPCT) solves the Optimal Streaming Problem (1).

III. Single-User Partially Anticipative Case

A. Fixed Horizon Control Scheme

Now we consider a 'partially anticipative' case in which only a finite window of future wireless capacity variations are known beforehand. Assume that at time t, we know the capacity $c_{[t,t+w]}$, where w is the future window size, and we do not know the capacity beyond $t + w$. Thus we cannot use an offline scheme like GPCT. However we can apply the GPCT algorithm on $[t, t + w]$, which provides a baseline for online schemes. We propose a Greedy Fixed Horizon Control (GFHC) transmission scheme in Algorithm 4 below.

GFHC is an online scheme that successively applies GPCT on a sequence of w-sized intervals. For each interval it must set the initial buffer state s_s, and target buffer state s'_e at

Algorithm 3 General Piecewise Constant Thresholding (GPCT)

Input: s_s, s_e, $l_{[t_0,t_n]}$, $u_{[t_0,t_n]}$, $c_{[t_0,t_0+\tilde{T}]}$

1: Perform greedy transmission on $[t_0,\infty]$ until $s_e - s_s$ of the video is delivered. Suppose the rebufferings under the greedy scheme happen at jump points $t_{n_1}, t_{n_2}, ..., t_{n_k}$, denote the corresponding rebuffering times as $\tau_{n_1}, \tau_{n_2}, ..., \tau_{n_k}$, and identify their rebuffering types (I or B). If a B-rebuffering happens at t_{n_i} for some $i \in [1,k]$, denote $\tilde{t}(n_i) = \max\{t < t_{n_i} + d(n_i) :$ playback buffer is full at $t\}$
2: $t_{n_0} \leftarrow 0$, $d(n_i) \leftarrow 0$, $i \leftarrow 1$
3: **while** $i \leq k$ **do**
4: **if** the rebuffering at t_{n_i} is an I-rebuffering **then**
5: do greedy transmission on $[t_{n_{i-1}} + d(n_{i-1}), t_{n_i} + d(n_i)]$. Let the corresponding transmission schedule be the solution $r^\star_{[t_{n_{i-1}}+d(n_{i-1}),t_{n_i}+d(n_i))}$
6: **else**
7: run PCT on $[t_{n_{i-1}} + d(n_{i-1}), \tilde{t}(n_i)]$ with input $\max[l(t_{n_{i-1}}), s_s]$, $u(\tilde{t}(n_i) - d(n_{i-1}))$, $l_{[t_{n_{i-1}},\tilde{t}(n_i)-d(n_{i-1})]}$, $u_{[t_{n_{i-1}},\tilde{t}(n_i)-d(n_{i-1})]}$ and $c_{[t_{n_{i-1}}+d(n_{i-1}),\tilde{t}(n_i)]}$; and do greedy transmission during $[\tilde{t}(n_i), t_{n_i} + d(n_i)]$. Let the corresponding transmission schedule be the solution $r^\star_{[t_{n_{i-1}}+d(n_{i-1}),t_{n_i}+d(n_i))}$
8: **end if**
9: $i \leftarrow i+1$
10: **end while**
11: **if** $n_k < n$ **then**
12: Run PCT on $[t_{n_k} + d(n_k), t_n + d(n_k)]$ with input $\max[l(t_{n_k}), s_s]$, s_e, $l_{[t_{n_k},t_n]}$, $u_{[t_{n_k},t_n]}$ and $c_{[t_{n_k}+d(n_k),t_n+d(n_k)]}$, and let the corresponding transmission schedule be the solution $r^\star_{[t_{n_k}+d(n_k),t_n+d(n_k)]}$
13: **end if**

Output: $r^\star_{[t_0,t_n+d(n_k)]}$

Algorithm 4 Greedy Fixed Horizon Control (GFHC)

Input: s_s, s_e, $l_{[t_0,t_n]}$, $u_{[t_0,t_n]}$, $c_{[t_0,t_0+\tilde{T}]}$

1: $i \leftarrow 0$, $d \leftarrow 0$
2: **repeat**
3: $s'_e \leftarrow s_s + m_i$, where m_i is the maximum amount of data that can be delivered (i.e., using greedy transmission) during $[t_0+iw, t_0+(i+1)w]$
4: run GPCT on $[t_0 + iw, t_0 + (i+1)w]$ with input s_s, s'_e, $l_{[t_0+iw-d,t_0+(i+1)w-d]}$, $u_{[t_0+iw-d,t_0+(i+1)w-d]}$ and $c_{[t_0+iw,t_0+(i+1)w)]}$. Let the resulting transmission schedule be $r^\star_{[t_0+iw,t_0+(i+1)w)}$, and the rebuffering time be τ
5: $d \leftarrow d + \tau$
6: $s_s \leftarrow s_s + \int_{t_0+iw}^{t_0+(i+1)w} r^\star(t)dt$
7: $i \leftarrow i+1$
8: **until** $s_s = s_e$

Output: $r(\cdot)$

the end of the interval. The latter is done in Step 3 of Algorithm 4, where $s'_e \leftarrow s_s + m_i$, which is the maximum one could achieve at the end of the ith interval using greedy transmission. The initial state s_s is initialized and subsequently updated (Step 6) once the transmission schedule for the current window is determined. GFHC runs GPCT on w intervals, during which it computes a transmission schedule and may incur rebuffering delays. Thus in Step 4, the constraints have been shifted by d the cumulative rebuffering incurred so far. The resulting transmission schedule $r^\star$ is the concatenation of those computed across w-windows.

Note in Step 3 shown in Algorithm 4 the target buffer state s'_e is chosen in a "greedy" manner, i.e., as high as possible in order to minimize rebuffering time. This is why we call the proposed scheme "greedy fixed horizon control". In fact, one can in principle define different kinds of Fixed Horizon Control (FHC) schemes by using different strategies in Step 3 of Algorithm 4 to choose s'_e. For example, we can define a "resource saving FHC" by setting $s'_e = \max[s_s, l((i+1)w)]$.

B. Competitive Optimality of GFHC

To evaluate the performance of GFHC, we use the "competitive ratio" which is defined as the ratio of the cost of GFHC and the optimal offline cost achieved by GPCT under the same problem settings (i.e., the same capacity variations, $l(\cdot)$, $u(\cdot)$, etc., but they are known ahead of time). The following theorem gives an upper bound on the competitive ratio.

***Theorem** 3:* Given a maximum playback buffer size b, video length T and size S, window size w, maximum capacity c_{max} and minimum capacity $c_{min} > 0$, the competitive ratio of GFHC to GPCT satisfies:

$$\frac{cost^G}{cost^O} \leq 1 + \max\left[\frac{1}{c_{min}}, \frac{T}{S}\right]\frac{b}{w}\left(\frac{c_{max}}{c_{min}} - 1\right).$$

Note that when $\frac{S}{T} > c_{min}$: i.e., the overall average video compression rate exceeds the minimum capacity the upper bound in Theorem 3 can be viewed as having two parts.. The first, $\frac{b}{wc_{min}}$, captures the size of the playback buffer relative to the minimum amount of data that will be delivered in future window. If these are close clearly the offline opportunistic scheduling realized by GPCT will not achieve great gains over GFHC. The second, $\frac{c_{max}}{c_{min}} - 1$ captures the worst case variability in capacity. In general, the worst case performance of GFHC is closer to GPCT under a larger minimum capacity, a larger prediction window size, a smaller playback buffer size and a smaller ratio between the maximum and the lowest capacities. The proof of Theorem 3 is as follows.

Proof: We denote by $cost^O = cost^O_u + cost^O_r$ and $cost^G = cost^G_u + cost^G_r$ the optimal costs achieved by GPCT and GFHC – the two terms correspond to the system utilization and rebuffering time. We also let $r^O()$ and $r^G()$ denote the optimal transmission schedules for GPCT of GFHC respectively. Suppose $t_0 = 0$ and let $s^O(t) = \int_0^t r^O(\tau)d\tau$ and $s^G(t) = \int_0^t r^G(\tau)d\tau$ be the the cumulative transmitted data for the two schemes. The greedy nature of GFHC ensures that $s^G(t) \geq s^O(t), \forall t \geq 0$, i.e., GFHC is always ahead of

GPCT, and so it has a rebuffering cost no higher than GPCT. Since GPCT strictly prioritizes minimization of rebuffering cost over system utilization, it follows that GPCT achieves the lowest possible rebuffering cost. We can conclude that $cost_r^G = cost_r^O$. Next we develop an upper bound for the difference in the utilization costs, i.e., $cost_u^G - cost_u^O$.

Suppose GFHC uses k window intervals, each with length w. On the ith interval, $1 \le i \le k$, the amount of data delivered by GFHC is denoted as $\delta_i^G = s^G(iw) - s^G((i-1)w)$. Also denote $\delta_i^O = s^O(iw) - s^O((i-1)w)$ as the amount of data delivered by the optimal scheme during the ith interval. Then we define two index sets as follows:

$$I_1 = \{i : \delta_i^G \le \delta_i^O\};$$

$$I_2 = \{i : \delta_i^G > \delta_i^O\}.$$

Note that $\forall i \in I_1$, GFHC delivers less data than the optimal scheme on interval i. We can define an intermediate transmission scheme on this interval, which starts at initial state $s^G((i-1)w)$, transmits at following rate $r^I(t)$:

$$r^I(t) = r^O(t)\mathbf{1}_{\{t\in[(i-1)w,\tilde{t}]\}},$$

where $\tilde{t} \le iw$ is chosen such that,

$$\int_{(i-1)w}^{\tilde{t}} r^I(t)dt = s^G(iw) - s^G((i-1)w).$$

It is easy to check that such a scheme is well-defined, and we can claim that the utilization cost of the intermediate scheme on interval i is no more than that of GPCT, since it transmits at the same rate as GPCT on $[(i-1)w, \tilde{t}]$, and transmits nothing on $[\tilde{t}, iw]$. Note on interval i, GPCT transmits an amount of $\delta_i^O - \delta_i^G$ more than the intermediate scheme. Transmission of the extra data requires an extra utilization no less than $\frac{\delta_i^O - \delta_i^G}{\tilde{T}c_{max}}$, which is obtained by assuming that the data is transmitted at maximum capacity thus achieving the best savings. Thus we have, $cost_{u,i}^I \le cost_{u,i}^O - \frac{\delta_i^O - \delta_i^G}{\tilde{T}c_{max}}$. Also since $s^G(t) \ge s^O(t), \forall t \ge 0$, we have that $s^I(t) \ge s^O(t)$ on interval i, which in turn implies $cost_{r,i}^I \le cost_{r,i}^O$ (in fact they are equal). Thus we have $cost_i^I \le cost_i^O - \frac{\delta_i^O - \delta_i^G}{\tilde{T}c_{max}}$.

However note that, the intermediate scheme and GFHC start at the same initial state and end at the same terminal state on interval i. Also note that GFHC uses GPCT on interval i and thus is optimal under fixed initial state and terminal state on that interval. As a result, we have $cost_i^G \le cost_i^I$. In conclusion we have

$$cost_i^G \le cost_i^O - \frac{\delta_i^O - \delta_i^G}{\tilde{T}c_{max}}.$$

By similar arguments we have, $\forall i \in I_2$,

$$cost_i^G \le cost_i^O + \frac{\delta_i^G - \delta_i^O}{\tilde{T}c_{min}},$$

where c_{min} comes from assuming the extra data is transmitted at the lowest capacity giving an upper bound on the additional utilization.

Summing up all the intervals we get:

$$cost^G \le cost^O - \sum_{i\in I_1} \frac{\delta_i^O - \delta_i^G}{\tilde{T}c_{max}} + \sum_{i\in I_2} \frac{\delta_i^G - \delta_i^O}{\tilde{T}c_{min}} \quad (3)$$

Note since GPCT and GFHC transmit the same amount of data at last, there must be:

$$\sum_{i\in I_1}(\delta_i^O - \delta_i^G) = \sum_{i\in I_2}(\delta_i^G - \delta_i^O) \stackrel{\text{def}}{=} A.$$

Also note that on each interval i, the difference between δ_i^O and δ_i^G cannot exceed the buffer size, i.e., $|\delta_i^O - \delta_i^G| \le b$, and since there are at most $\frac{\tilde{T}}{w}$ intervals, we have that

$$A \le \frac{\tilde{T}b}{w}. \quad (4)$$

Then we can rewrite inequality (3) as

$$cost^G \le cost^O + A(-\frac{1}{\tilde{T}c_{max}} + \frac{1}{\tilde{T}c_{min}}). \quad (5)$$

Combining inequalities (4) and (5), we have

$$\begin{aligned} cost^G &\le cost^O - \frac{\tilde{T}b/w}{\tilde{T}c_{max}} + \frac{\tilde{T}b/w}{\tilde{T}c_{min}} \\ &\le cost^O + \frac{b(c_{max} - c_{min})}{wc_{max}c_{min}}. \end{aligned}$$

Finally, dividing by $cost^O$ and noticing that $cost^O \ge cost_u^O \ge \frac{1}{\tilde{T}}\frac{S}{c_{max}}$, we obtain that

$$\begin{aligned} \frac{cost^G}{cost^O} &\le 1 + \frac{b(c_{max} - c_{min})\tilde{T}}{wc_{min}S}, \\ &\le 1 + \max[\frac{1}{c_{min}}, \frac{T}{S}]\frac{b}{w}(\frac{c_{max}}{c_{min}} - 1), \end{aligned}$$

where we used the fact that $\tilde{T} = \max[\frac{S}{c_{min}}, T]$.

■

IV. Multiuser Anticipative Case

In Section II we proposed GPCT and proved that it solves the Optimal Streaming problem (1). However, the algorithm was developed for a single-user case and it is hard to generalize it to the multiuser case. In this section, we develop sub-optimal multiuser schemes based on GPCT which have reasonable complexity.

A. Multiuser Piecewise Constant Thresholding Under Proportional Capacity Allocation (MTP)

Suppose a base station is serving n mobile users and user i has a peak capacity $c_i(t)$ at time t, $i = 1, 2, ..., n$. The peak capacities are assumed to be known beforehand. A simple way to deal with the multiuser issue is to make an up-front allocation of resources among the n users in a round robin fashion and thus the allocated capacity for user i is $\tilde{c}_i(t) = \frac{c_i(t)}{n}$, which is proportional to his/her peak capacity $c_i(t)$. Each user i is then assumed to have a capacity $\tilde{c}_i(t)$ which is independent of other users' capacities and thus we can apply GPCT to each user separately. We call this scheme multiuser piecewise

constant thresholding under proportional capacity allocation (MTP). MTP is straightforward to implement since every user can independently run a single-user algorithm on his/her own based on knowledge of his/her capacity and the number of users sharing the bottleneck, and requests transmission from server accordingly. Thus the scheme works in a decentralized way and there is no need for a centralized controller. However, although it applies GPCT which exploits temporal diversity in capacity variations well, MTP is based on a proportional capacity allocation which does not directly exploit multiuser diversity. Below we consider how both might be exploited.

B. Multiuser Piecewise Constant Thresholding Under Opportunistic Capacity Allocation (MTO)

We introduce a centralized scheme which exploits both temporal and multiuser diversity. Consider a base station serving n mobile users. To reduce the system utilization, there is a central scheduler at the base station which knows future capacity variations of the mobiles. The system operates in discrete (slotted) time and each time slot the scheduler chooses which of the video users could be served in the slot. In order to exploit capacity variations across different users while ensuring 'short-term' fairness, we use the opportunistic resource allocation scheme with a token counter mechanism proposed in [11]. It works as follows.

Each user i is associated a token counter T_i. At the beginning, all the token counters are set to be the same positive integer value m, which is referred to as the token limit. Each time slot, the scheduler searches among the users who have a non-zero token counter and chooses the user with the highest capacity[3]. T_i is decremented if user i is served in that time slot. When all the token counters are zero, they are reset to m. Using the same token limit m for all the users guarantees that the system allocates the same number (m) of time slots to each user within $m \cdot n$ slots. Subsequently each user has his/her allocated capacity $\tilde{c}_i(t)$ and thus GPCT can again be applied independently to each user. The resulting scheme is denoted the multiuser piecewise constant thresholding under opportunistic capacity allocation (MTO). MTO is of higher complexity than MTP since it is based on a centralized controller. However it can achieve a lower system utilization because it not only exploits the temporal diversity for each user but also exploits the multiuser diversity across all the users via the token counter mechanism. The choice of the token limit m affects the performance of MTO in that a higher token limit allows the system to exploit the multiuser diversity more aggressively and results in a higher decrease in the system utilization, however, a smaller token limit enforces more temporal fairness and results in a shorter rebuffering time. We will see this point later in Section V.

V. Simulation Results

Performance sensitivity of GFHC. We ran a simulation to explore the performance sensitivity of GFHC to window, playback buffer and temporal correlation in capacity variations. We considered a server delivering a 10-minute constant bit rate video, 900kbps, to a receiver. We simulated a slotted system, with slots of length 0.1sec. Thus the jump points for $l(\cdot)$ are at 0.1sec, 0.2sec, $0.3sec$,.. with values $l(0.1m) = 9m$, for $m = 1, 2, ..., 6000$. The initial buffer b_0 was set to 0.

Capacity variations, were modeled via a discrete time Markov chain whose states represent capacities, specifically $0, 250, 500, \ldots, 7500$kbps. The transition probability matrix for the chain is selected so that the invariant is a predefined stationary distribution corresponding to the PDF for the throughput of a randomly positioned user in an realistic HSPA system single antenna equalizer on the receiver under medium system load. We consider two such matrices which differ in the speed at which capacity varies. Specifically in the first case transitions can only happen between the two nearest states (e.g., from 250kbps to 500kbps) which results in slow variations (i.e., with temporal correlation); in the second case we simply take iid samples of the throughput PDF (i.e., with no temporal correlation).

Finally we let the receiver's playback buffer size b vary from $1620, 1890, 2160, 2430$ to 2700kb, and the window size w from $10, 20, 50, 100, 300$ to 600sec. Each scenario was simulated 20 times to obtain average results.

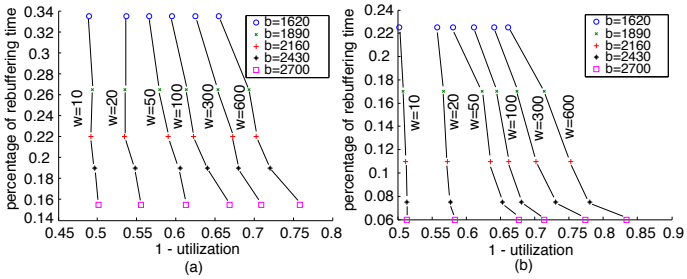

Fig. 3. (a) The performance of GFHC under correlated capacity variation. (b) The performance of GFHC under iid capacity variation. Each point shows the percent rebuffering time versus $(1 - \text{utilization})$ for a specific playback buffer size and window size. The points on the bottom right of the figures correspond to best performance.

The results shown in Figs. 3 (a) and (b) correspond to the scenarios with correlated and iid capacity variations respectively. The figures show the percent rebuffering time versus $(1 - \text{utilization})$ for varying playback buffer and window sizes. Note the points on the bottom right of the figures correspond to best performance, i.e., lowest system utilization and rebuffering time. The figures exhibit the following three observations.

1. For fixed b and capacity variation, increasing w significantly reduces utilization but does not affect the rebuffering time.

2. For fixed w and capacity variation, increasing b reduces rebuffering time and results in a marginal decrease in utilization. Note Theorem 3 suggests that a smaller b should result in a better performance, but this corresponded to worst case 'performance' vs the averages considered here.

3. For fixed b and w, temporal correlation in capacity variation results in increased rebuffering and a higher utilization.

[3]Selection could be weighted or driven by quantiles to address fairness concerns.

We also evaluated GFHC vs greedy transmission for a wireless capacity trace measured from a vehicle driving through Mountain View, CA. We considered a server delivering a 4min, 900kbps constant bit rate video to a receiver. The capacity trace was rescaled to have an average rate of 2000kbps. The playback buffer size was set to be 1/10 of the video size. The greedy transmission scheme resulted in a 67.69% utilization and 8.1sec of rebuffering time. The GFHC resulted in the same rebuffering time, but the utilization was reduced to 29.00%,48.83%,55.17% and 59.44% when the window w was set to 240, 120, 60, and 30sec respectively. This confirms the benefit of exploiting anticipated capacity variations for a trace from a real wireless network.

Performance and comparisons for multiuser algorithms. We ran simulations to compare the performance of MTP and MTO versus that of a proportional rate allocation scheme in which greedy transmission scheme is used and the time slots are assigned to the mobile users in a round robin fashion so that the transmission rate to each user is proportional to his/her peak capacities. In our simulation, we assume a fixed number of users are being served and each user is watching a 10-minute video with a constant bit rate of 90kbps. All the users have the same playback buffer size which was set to be 18000kb (i.e. one third of the video size), and they start watching the videos simultaneously. The model for capacity variations is the slotted model with correlated variations discussed above.

We let the number of users range from 1 to 12 and repeat each one 20 times to obtain average results. In MTO, we test the performance under four token limits which are 1,3,6 and 9. The average system utilization and average percent rebuffering time were computed and are plotted in Figs. 4 (a) and (b).

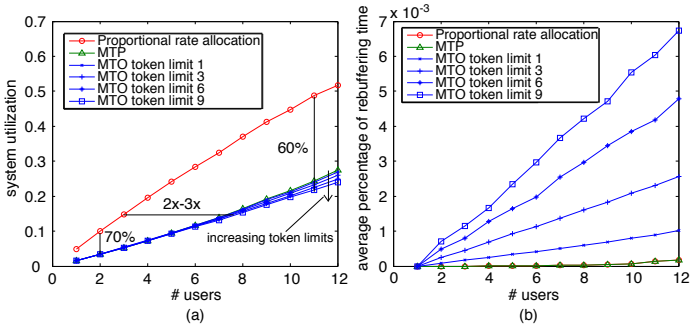

Fig. 4. (a) The system utilization. The proportional rate allocation has the highest system utilization. MTP and MTO achieve reduced system utilization. MTO schemes do a little better than MTP. An MTO scheme with a higher token limit achieves a lower system utilization. (b) The average percent rebuffering time. The proportional rate allocation and MTP have the same percent rebuffering time. MTO schemes result in higher rebuffering time which increases as the token limit increases.

Fig. 4 (a) exhibits the utilization as a function of the number of users. As can be seen, proportional rate allocation achieves the highest system utilization; by comparison, MTP and MTO achieve a $60-70\%$ reduction. Alternatively, for the same system utilization, MTP and MTO might allow 2x-3x more users. As expected MTO achieves a lower system utilization than MTP, since it exploits multiuser diversity, and MTO with a higher token limit results in a lower utilizations. However, as shown in Fig. 4 (b) this benefit is obtained at the cost of a additional rebuffering. Fig. 4 (b) exhibits the percent rebuffering time versus the number of users. It shows that MTP and proportional rate allocation require the same rebuffering time, but MTO results in more rebuffering as does MTO with higher token limits.

VI. Conclusion

By leveraging geolocation and contextual information regarding users mobility patterns it is possible to predict the large-scale wireless capacity variations mobile users are likely to see. In this paper we have developed and analyzed new cross-layer transport protocols that exploit knowledge of future capacity variations to deliver stored video (or other files) efficiently without compromising rebuffering/delays. Our analysis and simulations suggests this has substantial potential to increase the ability of wireless systems to deliver stored video in the case of mobile users which seeing high variability in their available capacity.

References

[1] "Cisco visual networking index: Forecast and methodology, 2011-2016," http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html.

[2] U. Shevade, Y. Chen, L. Qiu, Y. Zhang, V. Chandar, M. K. Han, H. H. Song, and Y. Seung, "Enabling high-bandwidth vehicular content distribution," in *Proc. ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, no. 23, 2010, pp. 1–12.

[3] C. Song, Z. Qu, N. Blumm, and A. Barabasi, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010.

[4] J. McManus and K. W. Ross, "A dynamic programming methodology for managing prerecorded VBR sources in packet-switched networks," *Telecommunication Systems*, vol. 9, pp. 133–152, 1998.

[5] J. Salehi, Z. Zhang, J. Kurose, and D. Towsley, "Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing," in *Proc. ACM SIGMETRICS*, May 1996, pp. 222–231.

[6] J.Rexford and D.Towsley, "Smoothing variable-bit-ratevideo in an internetwork," *IEEE/ACM Trans. on Networking*, vol. 7, no. 2, pp. 202–215, April 1999.

[7] G. V. der Auwera and M. Reisslein, "Implications of smoothing on statistical multiplexing of H.264/AVC and SVC video streams," *IEEE Trans. on Broadcasting*, vol. 55, no. 3, pp. 541–558, September 2009.

[8] M. Reisslein and K. W. Ross, "A join-the-shortest queue prefetching protocol for VBR video on demand," in *Proc. IEEE International Conference on Network Protocols (ICNP)*, October 1997, pp. 63–72.

[9] M. Reisslein, K. Ross, and V. Verillotte, "A decentralized prefetching protocol for VBR video on demand," in *Multimedia Applications, Services and Techniques-ECMAST'98*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 1998, vol. 1425, pp. 388–401.

[10] Z. Lu and G. de Veciana, "Optimizing stored video delivery for mobile networks: The value of knowing the future," http://users.ece.utexas.edu/~gustavo/LuD12.pdf.

[11] S. Patil and G. de Veciana, "Managing resources and quality of service in heterogeneous wireless systems exploiting opportunism," *IEEE/ACM Trans. on Networking*, vol. 15, no. 5, pp. 1046–58, October 2007.