Optimizing Stored Video Delivery for Wireless Networks: The Value of Knowing the Future

Zheng Lu and Gustavo de Veciana, *Fellow, IEEE*

***Abstract*—This paper considers the design of cross-layer opportunistic transport protocols for stored video over wireless networks with a slow varying (average) capacity. We focus on two key principles: 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, for example, 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 in 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. We also consider the multiuser case where one can exploit both future temporal and multiuser diversity. Finally, we investigate the impact of uncertainty in knowledge of future capacity variations, and propose an offline approach as well as an online algorithm to deal with such uncertainty. Our simulations and evaluation based on a measured wireless capacity trace exhibit robust potential gains for our proposed transmission schemes.**

***Index Terms*—Future capacity variations, opportunistic transport, optimization, stored video delivery.**

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 sevenfold between 2016 and 2021 and about 80 percent 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.

Manuscript received November 10, 2017; revised April 10, 2018; accepted May 21, 2018. Date of publication June 15, 2018; date of current version December 20, 2018. This work was supported in part by Intel and Cisco under the VAWN program, and in part by the National Science Foundation under Grant CNS-1343383. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Shiwen Mao. *(Corresponding author: Zheng Lu.)*

The authors are with the Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78712 USA (e-mail: zhenglu@utexas.edu; gustavo@ece.utexas.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2018.2847240

Even the successor to current cellular systems, 4G/5G 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 application-layer opportunistic transport protocols 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 Shevade *et al.* [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, Song *et al.* [3] show a potential 93% average 'predictability' suggesting knowing the future (in this regard) is quite reasonable. Moreover, the bit-rate of videos can be known or predicted ([4]–[7]) in advance, which provides opportunities for video clients to exploit knowledge of future capacity variations.

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 200 kbps, thus the size of the video is 800 kb. Suppose the capacity variation is as shown in Fig. 1. If the video is delivered at a fixed rate of roughly 200 kbps 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 2.5 secs, 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

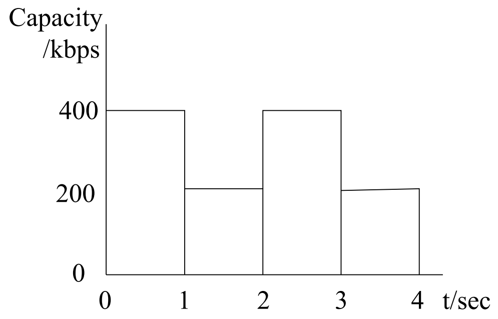


Fig. 1. The channel variations in four time slots. Transmitting the video at peak capacity in the first and third slots and transmitting nothing in the second and fourth slots, gives the lowest system utilization.

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 closest to ours, in terms of methodology, but differ in terms of their objectives. A piecewise constant-rate transmission scheme was developed in [8], 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 user interaction delay, and minimizing the average user interaction delay subject to a constraint on the maximum transmission rate. Similarly Salehi *et al.* [9] suggest several ways to reduce transmission rate variability for stored video delivery and propose an optimal smoothing scheme which is further explored in [10]–[12]. 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.

The key differentiating element of our work from the above is a formal investigation of how knowledge of future capacity variations could be used towards reducing utilization (increasing capacity) while minimizing video rebuffering. The idea of exploiting knowledge of future capacity variations has been studied by a few works, e.g., Jurca and Frossard [13] formulates multipath video streaming into an optimization problem under the assumption that the server can predict accurately the state of the network. The work however, aims at video distortion reduction, which is different from our work where we focus on minimizing system utilization. Zhou *et al.* [14] develops a class of Prediction-Based Adaptation (PBA) algorithms by utilizing predicted bandwidth, which significantly improves video as compared to the video transport protocols that do not employ such predictions. However this work focuses on maximizing video QoE, which is again different from our objective. Tadrous *et al.* [15] proposes an optimal proactive resource allocation which assumes users’ demand can be tracked, learned and predicted. And predictable peak-hour requests are served during off-peak time to reduce cost in the networks. This is similar to our work from the perspective of exploiting temporal diversity in the future capacity variations. However it has a much slower time scale, i.e., offloading traffic from peak time to off-peak time considers a time scale of hours, as compared to a time scale of seconds in our work. Also their paper does not consider the delay sensitivity of video delivery (e.g., rebuffering constraints) as we do. In [16] Bui *et al.* exploit future knowledge of capacity variations to improve the performance of video delivery. Their goal is to trade off video quality with rebuffering, which is different from our objective. Also they did not look into the case as we do, where predictions of future capacity variations have error. Bui and Widmer [17] has a similar objective as our work. It first proposes an offline optimal resource allocation that minimizes system utilization under perfect capacity predictions, then proposes an online algorithm to deal with uncertainty in the predictions. However, [17] is different from our work in the following aspects: (1) the offline algorithm in [17] only considers the optimization of one video user and treats all the other users as background traffic, but in our work we exploit multiuser diversity by jointly considering video traffic from multiple concurrent users; (2) we perform analysis and show an upper bound on the performance degradation caused by prediction error, which is not studied in [17]; and (3) to deal with prediction error, Bui and Widmer [17] proposes an online algorithm that iteratively uses the offline algorithm in a finite horizon, which has higher complexity and requires more statistical capacity information, as compared to our BDT algorithm which does not make use of the offline optimal algorithm and only requires the prediction of mean capacity in a finite horizon. Triki *et al.* [18] proposes a similar framework but with an extended objective which trades off system utilization with video quality, rebuffering and robustness. However the paper only focuses on the single-user anticipative case; it does not study the single-user partially anticipative case and the multiuser case as we do. In [19], [20], Abou-Zeid *et al.* propose a predictive green streaming optimization framework that exploits rate predictions to optimize several objectives including minimizing transmission airtime and base station power consumption. And later in [21] Atawia *et al.* propose a joint chance-constrained predictive resource allocation aiming at minimizing the energy consumed by the base stations in transmitting videos to users while satisfying their QoS level. Their objectives are similar to our work under the multiuser case. Also similar to our work, the paper indicates that the optimal solution is hard to calculate (it has non-convexity) and seeks sub-optimal solutions instead. But the two papers take different approaches: the two-stage solution in [21] is a centralized approach where all the calculation is done on the base station; whereas in our solution we decouple the multiuser diversity and the future temporal diversity, which offloads most of the computational complexity onto each individual client, and thus allows a light-weight centralized design that has a lower complexity. Moreover our work stands out from the aforementioned papers in the aspect that we perform analysis on the impact of prediction error of future capacity variations by deriving an upper bound on the performance degradation of our proposed algorithm caused by prediction error.

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. We propose approaches to exploit the knowledge of future capacity variations and study how to deal with error in the predictions of capacity variations. 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. In Section IV, we consider the multi-user anticipative case, and we develop two multiuser schemes based on GPCT, which are sub-optimal, 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. In Section VI we deal with the uncertainty in capacity predictions. We first propose an offline approach and show an upper bound on the performance degradation caused by prediction error. We then propose an online Buffer-Dependent Thresholding (BDT) algorithm that only requires the prediction of mean capacity for a finite window in the future. Simulations show that BDT is effective and robust. Section VII 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

[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.

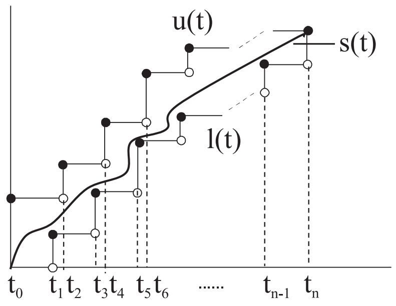


Fig. 2. The piecewise constant functions $l(\cdot)$, $u(\cdot)$ and the cumulatively transmitted data $s(\cdot)$. $l(t)$ is the cumulative amount of data consumed (i.e. watched) by the user over $[t_0, t]$. $u(t)$ is the maximum cumulative amount of data that can be received by the user over $[t_0, t]$.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

$l(\cdot)$ 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 $t_0, t_1, t_2, \ldots, t_n$. 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 $t_0 \leq t_1 \leq t_2 \leq \cdots \leq t_n$. 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 average system utilization during video watch period $[0, T]$ can be defined as:

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

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 $\tau_0, \tau_1, \ldots, \tau_n$. 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, $t_0, t_1, t_2, \ldots, t_n$ are fixed constants before transmission, which may be shifted by $\tau_0, \tau_1, \ldots, \tau_n$, which are variables whose values depend on the actual wireless capacities and rebuffering during transmission.

Our rebuffering cost is defined as

$$\text{cost}_r = a \sum_{i=0}^{n} \tau_i,$$

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 $\frac{\tau(c_{\max} - c_{\min})}{T c_{\min}}$, 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 $\frac{c_{\max}\tau}{c_{\min}T} - \frac{\tau}{T} = \frac{\tau(c_{\max} - c_{\min})}{T c_{\min}}$ in system utilization. Thus if we set $a = \frac{c_{\max} - c_{\min}}{T c_{\min}}$, we achieve our goal of strictly prioritizing minimization of rebuffering cost over system utilization.

The above rebuffering cost function is a simplification, since it only captures the cumulative rebuffering time without accounting for how rebuffering periods are distributed over time. In fact, a viewer may prefer that rebuffering happen all at once rather than being spread out and causing several interruptions during playback. To capture this concern one can take a concave function to each waiting time τ_i before summing them up so as to penalize a higher number of interruptions. In the rest of the paper, we will focus on the simpler cost function. The "concave-sum" version can be studied similarly.

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

Optimal Streaming Problem

$$\min_{r(\cdot), \vec{\tau}} \frac{1}{\tilde{T}} \int_0^{\infty} \frac{r(t)}{c(t)} dt + \frac{c_{\max} - c_{\min}}{\tilde{T} c_{\min}} \sum_{i=0}^{n} \tau_i,$$

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

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

$$\int_0^{t_n + \sum_{k=0}^{n} \tau_k} r(t)dt + b_0 = l(t_n). \tag{1}$$

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 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

$$\min_{r(\cdot)} \frac{1}{\tilde{T}} \int_0^{t_n} \frac{r(t)}{c(t)} dt,$$

$$\text{s.t.} \int_0^{t_i} r(t)dt + b_0 \geq l(t_i), \forall i \in I_l,$$

$$\int_0^{t_i} r(t)dt + b_0 \leq u(t_i), \forall i \in I_u,$$

$$\int_0^{t_n} r(t)dt + b_0 = l(t_n). \tag{2}$$

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 $[p_{i-1}, p_i]$, $i = 1, 2, \ldots, m$ 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: $\{1, 2, \ldots, m\} \to \{1, 2, \ldots, m\}$ 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.

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: $r^\star_{[t_0,t_n]}$, $m$
```

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 $s_e = l(t_n)$.

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 $[t_0, t_b)$. 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.

Algorithm 2: Breakpoint

```
Input: $s_s, s_e, t_s, t_e, l_{[t_s,t_e]}, u_{[t_s,t_e]}, c_{[t_s,t_e]}$
 1: loop
 2:   Apply single threshold transmission scheme
      on $[t_s, t_e]$ with initial state $s_s$ and terminal state $s_e$
      to get $\alpha$, $\tau$ and $r_{[t_s,t_e]}, \beta_{\alpha,\tau,t_s,[t_s,t_e]}$.
 3:   if $\beta_{\alpha,\tau,t_s,[t_s,t_e]}$ does not violate any buffer
      underflow or overflow constraints then
 4:     $t_b \leftarrow t_e, s_b \leftarrow \beta_{\alpha,\tau,t_s}(t_e)$
 5:     break
 6:   else
 7:     find the largest $i$, $t_s \le t_i \le t_e$, such that the
        violations on $[t_s, t_i]$ are of the same type
 8:     if the violation type is buffer overflow then
 9:       $s_e \leftarrow u(t_i), t_e \leftarrow t_i$
10:     else
11:       $s_e \leftarrow l(t_i), t_e \leftarrow t_i$
12:     end if
13:   end if
14: end loop
Output: $t_b, s_b, r_{[t_s,t_e)}$
```

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.

C. Proof of Optimality

We prove the optimality of PCT by induction on the total number of jumps n in $u(\cdot)$ and $l(\cdot)$.

First, when $n = 1$, the algorithm is optimal due to the characteristic of the thresholding scheme, i.e., one transmits only at the highest capacities.

As an induction hypothesis, suppose the algorithm is optimal for all the feasible video delivery optimization problems with n jumps where $n \le k-1$. We then show that it is also optimal for problems with k jumps.

If PCT results in a 1-piecewise constant thresholding solution, then it is optimal due to the characteristic of the thresholding scheme. Otherwise, if we obtain an m-piecewise solution denoted by $s_0(\cdot) = \int_0^{\cdot} r_0(t)dt$, where $m > 1$, then suppose the first breakpoint found by the algorithm is t_i, and without loss of generality we suppose $s_0(t_i) = u(t_i)$. (The other possible case is $s_0(t_i) = l(t_i)$, which can be addressed in a similar manner.) In the sequel, we will show that the solution $s_0(\cdot)$ is as good as any other feasible solution $s_1(\cdot)$. We consider two cases separately.

Case 1: Suppose $s_1(t_i) = u(t_i)$. In this case, we can change the constraint $s(t_i) \le u(t_i)$ to $s(t_i) = u(t_i)$ in the original Min Utilization (2). Then we will get a new optimization problem denoted by $\tilde{O}$. Note that both $s_0(\cdot)$ and $s_1(\cdot)$ are feasible solutions for $\tilde{O}$. On the other hand, $\tilde{O}$ can be divided into two video delivery optimization problems $\tilde{O}_1$ and $\tilde{O}_2$, where $\tilde{O}_1$ is on the interval $[t_0, t_i]$, $\tilde{O}_2$ is on the interval $[t_i, t_k]$. And we can solve them separately to obtain the optimal solution for $\tilde{O}$. Note that it follows from the characteristic of the proposed algorithm that, if we apply the algorithm on $\tilde{O}_1$ and $\tilde{O}_2$ separately, we will get the same result as $s_0(\cdot)$. Further, by the induction hypothesis, the proposed algorithm gives optimal solutions on $\tilde{O}_1$ and $\tilde{O}_2$, so $s_0(\cdot)$ is an optimal solution on $\tilde{O}$. Thus, $s_0(\cdot)$ is as good as $s_1(\cdot)$.

Case 2: Suppose $l(t_i) \le s_1(t_i) < u(t_i)$. In this case, we can remove the constraint $s(t_i) \le u(t_i)$ from the original Min Utilization (2), to get a new optimization problem $\tilde{O}$. Note that both $s_0(\cdot)$ and $s_1(\cdot)$ are feasible solutions for $\tilde{O}$, and by the induction hypothesis, we can apply the proposed algorithm on $\tilde{O}$ to get an optimal solution $s_2(\cdot)$. Suppose the utilizations of $s_1(\cdot)$ and $s_2(\cdot)$ are cost_{u1} and cost_{u2} respectively, then it should be such that $\text{cost}_{u1} \ge \text{cost}_{u2}$. Note since t_i is a breakpoint found by PCT algorithm and $s_0(t_i) = u(t_i)$ which corresponds to a buffer overflow violation, we can claim that if we relax the buffer overflow constraints at t_i, the associated optimal cumulative transmission at t_i should be greater than $u(t_i)$, i.e., $s_2(t_i) > u(t_i)$. Now we construct a feasible solution $s_0'(\cdot)$ to $\tilde{O}$ by doing time sharing between $s_1(\cdot)$ and $s_2(\cdot)$ as follows,

$$s_0'(t) = \lambda s_1(t) + (1-\lambda)s_2(t),\ t \in [t_0, t_k],$$

where λ is a parameter chosen from $(0, 1)$ such that,

$$s_0'(t_i) = s_0(t_i) = u(t_i).$$

Then, the utilization of $s_0'(\cdot)$ can be calculated as

$$\text{cost}_{u0}' = \lambda \text{cost}_{u1} + (1-\lambda)\text{cost}_{u2}.$$

Thus, we have $\text{cost}_{u2} \le \text{cost}_{u0}' \le \text{cost}_{u1}$. That means, scheme $s_0'(\cdot)$ is as good as $s_1(\cdot)$. But according to the result in Case 1, $s_0(\cdot)$ is as good as $s_0'(\cdot)$, so we can claim that $s_0(\cdot)$ is as good as $s_1(\cdot)$.

Thus, by induction the optimality of the proposed algorithm is proved.

Note that in the PCT algorithm, the dominating part of the computational complexity comes from applying single threshold transmission in a few loops. The single threshold transmission on interval $[t_i, t_j]$ can be done by sorting the capacities in all the time slots on the interval. Assume there are n time slots ($[t_1, t_2], [t_2, t_3], , [t_{n-1}, t_n]$) on interval $[t_s, t_e]$. Then in the worst case, PCT will need to apply single threshold transmission on all intervals $[t_i, t_j], \forall i < j$. Equivalently PCT needs to sort the capacities on all such intervals. One can show that the complexity of doing this is on the order of $\sum_{i=1}^{n}((n-i)\log(i))$, which is on the order of $n^2 \log(n)$.

D. 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 $t_{n_1}, t_{n_2}, \ldots, t_{n_k}$, denote the corresponding rebuffering times as $\tau_{n_1}, \tau_{n_2}, \ldots, \tau_{n_k}$, 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).

Note that GPCT algorithm requires the future capacity variations as an input. In the modern video transport architectures like HTTP video streaming, the transmission is driven by the receiver, i.e., the receiver will decide when to request for downloading the next chunk of video. In that case GPCT will be running on the receiver side, where the wireless capacity variations can also be measured and predicted. However in the conventional video transport architectures, where transmission can be driven by the video server, our algorithm will need to be running on the video server side. In that case, the clients and/or the base station need to send feedbacks to inform the video server of the predictions of capacity variations, which can impose extra implementation overhead.

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
    $t_{n_1}, t_{n_2}, \ldots, t_{n_k}$, denote the corresponding rebuffering
    times as $\tau_{n_1}, \tau_{n_2}, \ldots, \tau_{n_k}$, 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 $i \le k$ 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: $r^\star_{[t_0,t_n+d(n_k)]}$
```

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 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

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 $\tau$
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: $r(\cdot)$
```

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). We proved in [22] the following theorem, which 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:

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

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.

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, $i = 1, 2, \ldots, n$. 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 [23]. 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

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

Capacity	CDF	Capacity	CDF	Capacity	CDF
250	0.0024	2750	0.6851	5250	0.9784
500	0.012	3000	0.7284	5500	0.9832
750	0.1106	3250	0.7716	5750	0.9856
1000	0.2139	3500	0.8173	6000	0.9904
1250	0.2764	3750	0.863	6250	0.9952
1500	0.3582	4000	0.8894	6500	0.9976
1750	0.4303	4250	0.9135	6750	1
2000	0.512	4500	0.9375	7000	1
2250	0.5721	4750	0.9519	7250	1
2500	0.6226	5000	0.9615	7500	1

Fig. 3. The stationary distribution (cdf) of capacity (kbps) used in our simulation.

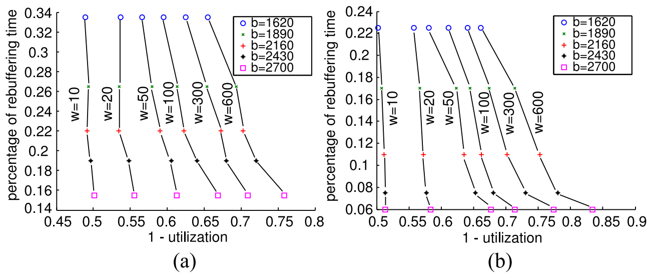


Fig. 4. (a) The performance of GFHC under correlated capacity variation. (b) The performance of GFHC under iid capacity variation. Each point shows the percent rebuffering time versus $(1 - \text{utilization})$ for a specific playback buffer size and window size. The points on the bottom right of the figures correspond to best performance.

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, 900 kbps, to a receiver. We simulated a slotted system, with slots of length 0.1 sec. Thus the jump points for $l(\cdot)$ are at 0.1 sec, 0.2 sec, 0.3 sec,.. with values $l(0.1m) = 9$ m, for $m = 1, 2, \ldots, 6000$. The initial buffer b_0 was set to 0.

Capacity variations, were modeled via a discrete time Markov chain whose states represent capacities, specifically $0, 250, \ldots, 7500$ kbps. The transition probability matrix for the chain is selected so that the invariant is a pre-defined 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. The distribution is shown in Fig. 3. The source of the distribution is from field measurements conducted by a North American mobile network operator. We consider two such matrices which differ in the speed at which capacity varies. Specifically in the first case transitions can only happen between two neighboring states (e.g., from 250 kbps to 500 kbps) 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 2700 kb, and the window size w from 10,20,50,100,300 to 600 sec. Each scenario was simulated 20 times to obtain average results.

The results shown in Figs. 4(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.

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 4 min, 900 kbps constant bit rate video to a receiver. The capacity trace was rescaled to have an average rate of 2000 kbps. 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 8.1 sec 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 30 sec 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

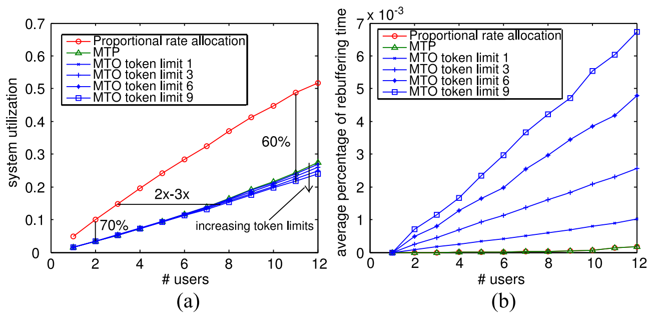


Fig. 5. (a) The system utilization. The proportional rate allocation has the highest system utilization. MTP and MTO achieve reduced system utilization. MTO schemes do a little better than MTP. An MTO scheme with a higher token limit achieves a lower system utilization. (b) The average percent rebuffering 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.

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 90 kbps. All the users have the same playback buffer size which was set to be 18000 kb (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. 5(a) and (b).

Fig. 5(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. 5(b) this benefit is obtained at the cost of a additional rebuffering. Fig. 5(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.

Note that we have used a wide range of buffer sizes (e.g., a few seconds, 24 seconds, and 200 seconds of video content) in our experiments. The results indicate that our proposed algorithms are able to achieve significant gains in all the scenarios.

VI. Uncertainty in Knowledge of Future Capacity Variations

So far we have developed video delivery policies assuming knowledge of future capacity variations (either in the anticipative case or the partially anticipative case) is perfect. However in practice there will be uncertainty in predictions of future capacity. Such uncertainty may arise from many sources, e.g., the mapping of wireless signal strength on the coverage map to capacity, uncertainty in a user's motion, interference from other mobile users, uncertainty in the number of mobile users contending for resources, etc.

Such uncertainty limits the optimality of our proposed approaches (GPCT and GFHC). For example, GPCT opportunistically chooses when to request video delivery based on future capacity variations. If the true capacity variations are lower than the predicted ones, then the users may experience longer rebuffering times. Similarly one can show that the minimum system utilization will not be achieved under imperfect capacity predictions.

In this section we explore different approaches to address such uncertainty. We first develop an offline approach which is able to achieve a minimum rebuffering time with an upper-bound on system utilization, under the assumption that the uncertainty in capacity variations can be bounded. We then propose an online algorithm that can deal with general types of uncertainty.

A. *Offline Approach Under Uncertainty in Capacity Prediction*

Let $\tilde{c}(t)$ denote the predicted capacity at time t, and $c(t)$ the true capacity which is unknown beforehand. If the prediction is good, such that $\tilde{c}(\cdot)$ is uniformly close to $c(\cdot)$, then it seems reasonable to use GPCT on the predicted capacity function to obtain an offline video delivery strategy. We shall denote such a policy by $\tilde{\mathcal{S}}^\star$. This policy can then be used under the true capacity function to mediate video delivery. Note that strategy $\tilde{\mathcal{S}}^\star$ specifies when the user should request video delivery under the predicted capacity function $\tilde{c}(\cdot)$, which is a set of time intervals, e.g., $\tilde{\mathcal{S}}^\star = \{[t_1, t_2]\}$ indicating that the user should request video delivery during $[t_1, t_2]$ under capacity $\tilde{c}(\cdot)$. However, when the user applies $\tilde{\mathcal{S}}^\star$ under the true capacity function $c(\cdot)$, some changes have to be made. For example, if $\tilde{c}(t) \leq c(t)$, $\forall t$, the user have more video delivered than expected. To avoid buffer overflow violations, the user needs to adjust policy $\tilde{\mathcal{S}}^\star$ to stop requesting video delivery whenever the playback buffer is full. We denote this modification to $\tilde{\mathcal{S}}^\star$ by $\tilde{\mathcal{S}}^\star_m$.

We can compare the perfromance of $\tilde{\mathcal{S}}^\star_m$ obtained based on the predicted capacity function to the optimal policy (GPCT) $\mathcal{S}^\star$ obtained assuming the true capacity function were known. Since we give higher priority to avoiding rebuffering, it is reasonable to require that $\tilde{\mathcal{S}}^\star_m$ result in no more rebuffering time than $\mathcal{S}^\star$. The following lemma shows that such requirement is met if $\tilde{c}(t) \leq c(t)$, $\forall t$.

Lemma 1: Suppose $\mathcal{S}^\star$ and $\tilde{\mathcal{S}}^\star_m$ are as defined above, then applying $\tilde{\mathcal{S}}^\star_m$ on the true capacity function $c(\cdot)$ results in no more rebuffering time than $\mathcal{S}^\star$, if for all t

$$0 < \tilde{c}(t) \leq c(t). \tag{3}$$

Proof: We prove the lemma by contradiction. Suppose the claim is not true, i.e., $\tilde{\mathcal{S}}^\star_m$ results in more rebuffering time than $\mathcal{S}^\star$. This implies that there exists a period $[t, t+\tau] \not\subseteq \tilde{\mathcal{S}}^\star_m$, such that the modified strategy $\tilde{\mathcal{S}}^\star_m \cup [t, t+\tau]$ reduces the rebuffering time of $\tilde{\mathcal{S}}^\star_m$. Note that since $\tilde{\mathcal{S}}^\star_m$ differs from $\tilde{\mathcal{S}}^\star$ only at times when the buffer is full, we can claim that $[t, t+\tau] \notin \tilde{\mathcal{S}}^\star$. Also by Condition (3) applying $\tilde{\mathcal{S}}^\star$ to $\tilde{c}(\cdot)$ delivers no more than $\tilde{\mathcal{S}}_m^\star$ on $c(\cdot)$ cumulatively at any time. Thus strategy $\tilde{\mathcal{S}}^\star \cup [t, t+\tau]$ results in less rebuffering time than $\tilde{\mathcal{S}}^\star$ under the capacity function $\tilde{c}(\cdot)$, which contradicts the fact that $\tilde{\mathcal{S}}^\star$ was obtained using GPCT and thus results in the minimum rebuffering time.

However in practice, Condition (3), i.e., that predictions are always pessimistic, may not hold, and the appropriate offline approaches should depend on the uncertainty in capacity prediction, i.e., the difference between $\tilde{c}(\cdot)$ and $c(\cdot)$. To pursue this further we assume that the prediction error can be bounded.

Definition 3: We say the capacity function prediction satisfies an (α, β) - *prediction error* if for all t

$$c(t) \in [(1-\alpha)\tilde{c}(t), (1+\beta)\tilde{c}(t)],\ \alpha, \beta \in [0, 1). \tag{4}$$

Note that under the (α, β) - prediction error, the true capacity function is bounded within a certain range of the predicted one. Let $\hat{c}(t) = (1-\alpha)\tilde{c}(t)$, which is thus a lower bound on the true capacity. Let $\hat{\mathcal{S}}^\star$ and $\hat{\mathcal{S}}_m^\star$ denote the strategy obtained by applying GPCT on $\hat{c}(\cdot)$ and the corresponding modified strategy obtained by applying $\hat{\mathcal{S}}^\star$ on $c(\cdot)$ avoiding buffer overflow, respectively. Now according to Lemma 1, $\hat{\mathcal{S}}_m^\star$ should result in no more rebuffering time than $\mathcal{S}^\star$.

To evaluate the performance of $\hat{\mathcal{S}}_m^\star$, we need to compare the system utilization (denoted by $\hat{u}_m^\star$) to the optimal system utilization (denoted by $u^\star$) obtained by $\mathcal{S}^\star$. The following lemma shows that $\hat{u}_m^\star$ can indeed be upper bounded.

Lemma 2: Suppose $\hat{\mathcal{S}}_m^\star$, $\mathcal{S}^\star$, $\hat{u}_m^\star$ and $u^\star$ are defined as above, $\tilde{T}$, $c_{\min}$, S are as defined in Section II, and the true capacity satisfies Condition (4), then

$$u^\star \le \hat{u}_m^\star \le u^\star + \frac{\alpha+\beta}{1+\beta}\frac{S}{c_{\min}\tilde{T}}. \tag{5}$$

Proof: The first inequality in (5) holds due to the fact that $\mathcal{S}^\star$ is the strategy obtained by applying GPCT to the true capacity function $c(\cdot)$, which according to Theorem 2 should lead to the minimum system utilization among all strategies that result in the minimum rebuffering time for a given capacity function.

We prove the second inequality in (5) as follows. Due to the fact that $\hat{\mathcal{S}}_m^\star$ is the modified version of $\hat{\mathcal{S}}^\star$ so as to avoid buffer overflow, and that $\hat{c}(t) \le c(t),\ \forall t$, we can claim that the system utilization resulting from applying $\hat{\mathcal{S}}_m^\star$ to $\hat{c}(\cdot)$ (denoted as $\hat{u}^\star$) is no less than $\hat{u}_m^\star$, which gives:

$$\hat{u}_m^\star \le \hat{u}^\star. \tag{6}$$

Paralleling the way we obtained the strategy $\hat{\mathcal{S}}_m^\star$, we will construct another strategy $\mathcal{S}_m^\star$ by applying $\mathcal{S}^\star$ to the capacity function $\hat{c}(\cdot)$. However since $\hat{c}(t) \le c(t),\ \forall t$, $\mathcal{S}^\star$ may end up delivering only a part of the video under $\hat{c}(\cdot)$. We deliver the remaining part of the video (denote its size as S_r bits) using time intervals not included in $\mathcal{S}^\star$ and in a greedy manner (i.e., deliver as early as possible) to keep the rebuffering time as low as possible. We denote this strategy by $\mathcal{S}_m^\star$ and the corresponding system utilization is denoted by $u_m^\star$. Note that according to Theorem 2, it follows that

$$\hat{u}^\star \le u_m^\star. \tag{7}$$

Next we calculate the gap between $u_m^\star$ and $u^\star$. Note that the file size of the video S can be written as:

$$S = \int_{t \in \mathcal{S}^\star} c(t)dt. \tag{8}$$

Similarly, S_r can be written as:

$$S_r = S - \int_{t \in \mathcal{S}^\star} \hat{c}(t)dt. \tag{9}$$

According to the definition of $\hat{c}(\cdot)$ and Eq. (4), we have:

$$\frac{1-\alpha}{1+\beta}c(t) \le \hat{c}(t),\ \forall t. \tag{10}$$

Combining (8), (9) and (10), we obtain:

$$S_r \le \frac{\alpha+\beta}{1+\beta}S. \tag{11}$$

Thus the extra time spent delivering S_r is no more than $\frac{\alpha+\beta}{1+\beta}\frac{S}{c_{\min}}$. Normalizing the time by $\tilde{T}$ we obtain a bound on the gap of the realized system utilization:

$$u_m^\star - u^\star \le \frac{\alpha+\beta}{1+\beta}\frac{S}{c_{\min}\tilde{T}}. \tag{12}$$

Combining (6), (7) and (12), we obtain:

$$\hat{u}_m^\star - u^\star \le \frac{\alpha+\beta}{1+\beta}\frac{S}{c_{\min}\tilde{T}},$$

which proves the second inequality in (5). ■

Lemma 2 shows that under errors in capacity prediction if the prediction is good enough (i.e. α and β are close to zero), then the offline strategy $\hat{\mathcal{S}}_m^\star$ performs nearly as well as the optimal offline strategy $\mathcal{S}^\star$.

So far we have proposed an offline approach under bounded prediction error. It first uses GPCT on a lower bound of the capacity variation (i.e., $\hat{c}(\cdot)$) and then modifies the obtained strategy to be feasible, i.e., avoiding buffer overflow when it is applied to the true capacity function. We have also showed that the proposed approach results in no more rebuffering time than GPCT, and the resulting system utilization can be bounded if the uncertainty in the capacity prediction is bounded. These results are summarized in the following theorem.

Theorem 4: Suppose $\hat{\mathcal{S}}_m^\star$ is the offline video delivery strategy proposed earlier in this section, and $\mathcal{S}^\star$ is the optimal strategy obtained by GPCT. Then under (α, β) - capacity prediction error, Eq. (4), $\hat{\mathcal{S}}_m^\star$ results in no more rebuffering time than $\mathcal{S}^\star$. Also $\hat{\mathcal{S}}_m^\star$ results in a higher system utilization than $\mathcal{S}^\star$, which can be upper bounded as shown in (5).

B. Online Approach Under Uncertainty in Capacity Prediction

In the previous subsection we proposed an offline approach, the performance of which, can only be guaranteed when uncertainty is small. If there is substantial uncertainty in the prediction of the future capacity variations, online approaches are preferable since they are able to adjust their strategies based on what they have experienced. In this subsection, we propose an online approach in which the users adaptively decide their thresholds

Algorithm 5: Buffer-Dependent Thresholding (BDT)

```
Input: $r, S, w_{\max}, \overline{b}, \underline{b}$
 1: $i \leftarrow 0, s = 0$
 2: $t_i^s \leftarrow 0$
 3: repeat
 4:     $b_i \leftarrow$ length (sec) of unwatched video in buffer at $t_i^s$
 5:     $w_i \leftarrow \min[w_{\max}, b_i]$
 6:     $t_i^e \leftarrow t_i^s + w_i$
 7:     $\tilde{c}_i \leftarrow$ predicted mean capacity in $[t_i^s, t_i^e]$
 8:     if $\underline{b} \le b_i \le \overline{b}$ then
 9:         $\gamma_i \leftarrow \max[\tilde{c}_i - r, 0]$
10:     else if $b_i > \overline{b}$ then
11:         $\gamma_i \leftarrow \max[\tilde{c}_i - r + \frac{r(b_i - \overline{b})}{w_i}, 0]$
12:     else
13:         $\gamma_i \leftarrow 0$
14:     end if
15:     $\gamma(t)|_{t \in [t_i^s, t_i^e]} = \gamma_i$
16:     During $[t_i^s, t_i^e]$, request video only when the true
        capacity is above $\gamma_i$ and the playback buffer is
        not full.
17:     $s_i \leftarrow$ the amount of video delivered during $[t_i^s, t_i^e]$
18:     $s \leftarrow s + s_i$
19:     $i \leftarrow i + 1$
20:     $t_i^s \leftarrow t_{i-1}^e$
21: until $s = S$
Output: $\gamma(\cdot)$
```

based on their playback buffer status and the average value of their predicted capacity for a finite window into the future. The basic idea of our online approach is that when a user's playback buffer is low, in order to avoid rebuffering, the user should use a small threshold to request video content as soon as possible; when a user's playback buffer is high, in order to reduce the system utilization, the user should use a relatively large threshold based on the prediction of the average future capacity to exploit the temporal opportunism in capacity variations.

To simplify the problem, we assume the requested video has a constant bit rate r, and the size of the video is S. Suppose we can predict the average capacity for a window into the future of $w_{\max}$ seconds. Based on the capacity predictions and the playback buffer status we make sequential decisions on the capacity threshold γ (i.e., video will be requested only when the true capacity is above γ and the playback buffer is not full). The thresholds are determined by a Buffer-Dependent Thresholding (BDT) strategy displayed in Algorithm 5.

In Algorithm 5, time is divided into intervals and the capacity threshold γ is determined sequentially on each interval. At the beginning of the ith interval, suppose there is b_i seconds of unwatched video content in the playback buffer, then the length of the ith interval is $w_i = \min[w_{\max}, b_i]$. Suppose the average predicted capacity in the ith interval is $\tilde{c}_i$. We set two thresholds $\overline{b}$ and $\underline{b}$ on the playback buffer with $\overline{b} > \underline{b}$, indicating the fullness of buffer. We consider the buffer is high if $b_i > \overline{b}$ and low if $b_i < \underline{b}$. The capacity threshold of the ith interval γ_i is then determined based on the buffer status.

First, if the buffer is neither high nor low, i.e., $\underline{b} \le b_i \le \overline{b}$, then the threshold is set to

$$\gamma_i = \max[\tilde{c}_i - r, 0]. \tag{13}$$

We explain the rationale of this capacity threshold as follows.

Note that in the thresholding policy (13) if $\tilde{c}_i \le r$, then $\gamma_i = 0$, which leads to the greedy strategy and results in the minimum rebuffering time. Otherwise if $\tilde{c}_i > r$, the following lemma indicates that if the prediction of the average future capacity is accurate, then the thresholding policy (13) results in no rebuffering.

Lemma 3: In the thresholding policy (13), suppose the prediction of the average capacity is accurate, i.e.,

$$\tilde{c}_i = \frac{1}{t_i^e - t_i^s} \int_{t_i^s}^{t_i^e} c(t)dt, \tag{14}$$

where t_i^s and t_i^e are the start and end time of the ith interval. If $\tilde{c}_i > r$, then the following holds:

$$\frac{1}{t_i^e - t_i^s} \int_{t \in I_i^+} c(t)dt \ge r, \tag{15}$$

where $I_i^+ = \{t | c(t) \ge \gamma_i, t \in [t_i^s, t_i^e]\}$, which is the set of times where video delivery will be requested.

Proof: Let $I_i^- = [t_i^s, t_i^e] \backslash I_i^+$, which is the set of times where no video will be requested. And let $|I_i^-|$ be the total length of time in the set I_i^-. According to (14), it follows that:

$$\tilde{c}_i = \frac{1}{t_i^e - t_i^s} \int_{t \in I_i^+} c(t)dt + \frac{1}{t_i^e - t_i^s} \int_{t \in I_i^-} c(t)dt. \tag{16}$$

Due to the definition of I_i^-, we have:

$$\frac{1}{|I_i^-|} \int_{t \in I_i^-} c(t)dt < \gamma_i = \tilde{c}_i - r. \tag{17}$$

Since $|I_i^-| \le t_i^e - t_i^s$, the following can be obtained from (17):

$$\frac{1}{t_i^e - t_i^s} \int_{t \in I_i^-} c(t)dt < \gamma_i = \tilde{c}_i - r. \tag{18}$$

Then (15) can be proved by combining (16) and (18). ■

Lemma 3 shows that the playback buffer will keep growing when $\tilde{c}_i > r$. If the buffer grows too high, i.e., $b_i > \overline{b}$ it is reasonable to set the threshold γ in a more aggressive way than (13) so as to further reduce system utilization. So when $b_i > \overline{b}$, we will reduce the amount of requested video so that the playback buffer drops to $\overline{b}$ at the end of the ith interval. Thus the total amount of video delivered is $rw_i - r(b_i - \overline{b})$. By analogy with the thresholding policy (13), we set the threshold to be:

$$\gamma_i = \max\left\{\tilde{c}_i - \frac{rw_i - r(b_i - \overline{b})}{w_i}, 0\right\},$$

which motivates thresholding policy when buffer is high in BDT (Line 11).

Lemma 3 holds under the assumption that the prediction of the average capacity is accurate. However if the prediction is

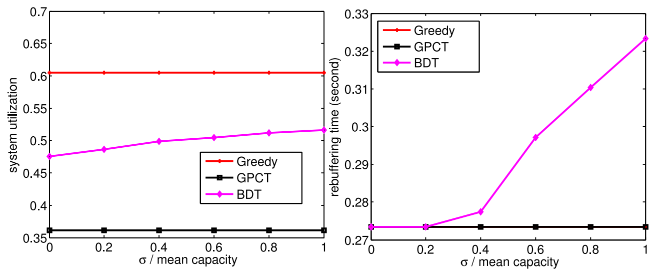


Fig. 6. Performance comparison among BDT, GPCT and the greedy strategy. BDT achieves up to a 15% reduction in system utilization as compared to the greedy strategy and only results in a slight increase in rebuffering time. Also as the prediction error grows, the performance of BDT does not drop too much.

higher than the true capacity, the thresholding policy (13) may lead to extra rebuffering time. To mitigate this problem, we use the greedy strategy, i.e., set $\gamma_i = 0$ when $b_i < \underline{b}$, which results in policy when the buffer is low in BDT (Line 13)

Note that BDT does not rely on precise capacity predictions. It only requires reasonable precision in the prediction of the average capacity variations for a finite window into the future.

To test the performance of BDT, we ran simulations where we use the same settings as in Section V. For BDT, we took $w_{\max} = 10$ s, $\underline{b} = 2$ s, $\overline{b} = 12$ s, $r = 900$ kbps, and the buffer size is 21600 kbits, which corresponds to 24 seconds of video. We generate prediction error for the average capacity variations as i.i.d. Gaussian random variables with zero mean and standard deviation σ varying within the set $\{0, 0.2\,\mathrm{c_{avg}}, 0.4\,\mathrm{c_{avg}}, 0.6\,\mathrm{c_{avg}}, 0.8\,\mathrm{c_{avg}}, \mathrm{c_{avg}}\}$, where $\mathrm{c_{avg}}$ is the average of the true capacity. We compare BDT with GPCT and the greedy strategy which always delivers video content as soon as possible. The results are shown in Fig. 6. As can be seen, BDT achieves up to a 15% reduction in system utilization as compared to the greedy strategy, while the optimal offline solution achieves a 25% reduction in the same simulation scenario. Moreover, BDT only results in a slight increase in rebuffering time (up to 0.05 seconds in the simulations, which is almost negligible as compared to the length of the video). Also as the prediction error grows, the performance of BDT does not drop too much. Thus BDT is effective in terms of reducing system utilization, and simultaneously keeping a low rebuffering time, and can work against uncertainty in future capacity variations.

VII. 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 suggest this has substantial potential to increase the ability of wireless systems to deliver stored video in the case of mobile users seeing high variability in their available capacity.

Acknowledgment

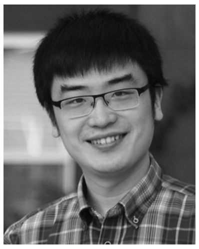
The authors would like to thank Xiaoqing Zhu and Hao Hu for the discussions and sharing their simulation data.

References

[1] "Cisco visual networking index: Forecast and methodology, 2016-2021," [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/vis ual-networking-index-vni/complete-white-paper-c11-481360.html.
[2] U. Shevade *et al.*, "Enabling high-bandwidth vehicular content distribution," in *Proc. ACM Int. Conf. Emerg. Netw. Experiments Technol.*, 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] N. Bui, F. Michelinakis, and J. Widmer, "A model for throughput prediction for mobile users," in *Proc. 20th Eur. Wireless Conf. Eur. Wireless*, May 2014, pp. 1–6.
[5] R. Margolies *et al.*, "Exploiting mobility in proportional fair cellular scheduling: Measurements and algorithms," *IEEE/ACM Trans. Netw.*, vol. 24, no. 1, pp. 355–367, Feb. 2016.
[6] H. Kalbkhani, M. G. Shayesteh, and N. Haghighat, "Adaptive lstar model for long-range variable bit rate video traffic prediction," *IEEE Trans. Multimedia*, vol. 19, no. 5, pp. 999–1014, May 2017.
[7] A. Kalampogia and P. Koutsakis, "H.264 and h.265 video bandwidth prediction," *IEEE Trans. Multimedia*, vol. 20, no. 1, pp. 171–182, Jan. 2017.
[8] J. McManus and K. W. Ross, "A dynamic programming methodology for managing prerecorded VBR sources in packet-switched networks," *Telecommun. Syst.*, vol. 9, pp. 133–152, 1998.
[9] 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 Int. Conf. Meas. Model. Comput. Syst.*, May 1996, pp. 222–231.
[10] J. Rexford and D. Towsley, "Smoothing variable-bit-rate video in an internetwork," *IEEE/ACM Trans. Netw.*, vol. 7, no. 2, pp. 202–215, Apr. 1999.
[11] G. V. der Auwera and M. Reisslein, "Implications of smoothing on statistical multiplexing of H.264/AVC and SVC video streams," *IEEE Trans. Broadcast.*, vol. 55, no. 3, pp. 541–558, Sep. 2009.
[12] W. C. Feng and J. Rexford, "Performance evaluation of smoothing algorithms for transmitting prerecorded variable-bit-rate video," *IEEE Trans. Multimedia*, vol. 1, no. 3, pp. 302–312, Sep. 1999.
[13] D. Jurca and P. Frossard, "Video packet selection and scheduling for multipath streaming," *IEEE Trans. Multimedia*, vol. 9, no. 3, pp. 629–641, Apr. 2007.
[14] X. K. Zou *et al.*, "Can accurate predictions improve video streaming in cellular networks?" in *Proc. 16th Int. Workshop Mobile Comput. Syst. Appl.*, 2015, pp. 57–62.
[15] J. Tadrous, A. Eryilmaz, and H. E. Gamal, "Proactive content download and user demand shaping for data networks," *IEEE/ACM Trans. Netw.*, vol. 23, no. 6, pp. 1917–1930, Aug. 2014.
[16] N. Bui, S. Valentin, and J. Widmer, "Anticipatory quality-resource allocation for multi-user mobile video streaming," in *Proc. 2nd Workshop Commun. Netw. Techn. Contemporary Video, Conjunction 34th IEEE Int. Conf Comput. Commun.*, Apr. 2015, pp. 245–250.
[17] N. Bui and J. Widmer, "Mobile network resource optimization under imperfect prediction," in *Proc. IEEE 16th Int. Symp. World Wireless, Mobile Multimedia Netw.*, Jun. 2015, 1–9.
[18] I. Triki, R. El-Azouzi, and M. Haddad, "Newcast: Anticipating resource management and qoe provisioning for mobile video streaming," in *Proc. IEEE 17th Int. Sympo. World Wireless, Mobile Multimedia Netw.*, Jun. 2016, pp. 1–9.
[19] H. Abou-Zeid and H. S. Hassanein, "Toward green media delivery: Location-aware opportunities and approaches," *IEEE Wireless Commun.*, vol. 21, no. 4, pp. 38–46, Aug. 2014.
[20] H. Abou-zeid, H. S. Hassanein, and S. Valentin, "Energy-efficient adaptive video transmission: Exploiting rate predictions in wireless networks," *IEEE Trans. Veh. Technol.*, vol. 63, no. 5, pp. 2013–2026, Jun. 2014.
[21] R. Atawia, H. Abou-zeid, H. S. Hassanein, and A. Noureldin, "Joint chance-constrained predictive resource allocation for energy-efficient video streaming," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1389–1404, May 2016.

[22] Z. Lu and G. de Veciana, “Optimizing stored video delivery for mobile networks: The value of knowing the future,” in *Proc. Proc. IEEE INFOCOM*, Apr. 2013, pp. 2706–2714.

[23] S. Patil and G. de Veciana, “Managing resources and quality of service in heterogeneous wireless systems exploiting opportunism,” *IEEE/ACM Trans. Netw.*, vol. 15, no. 5, pp. 1046–58, Oct. 2007.



Zheng Lu received the B.E. degree in electronics engineering from Tsinghua University, Beijing, China, in 2009. He received the M.S.E. and Ph.D. degrees in electrical and computer engineering from the University of Texas at Austin, Austin, TX, USA, in 2011 and 2015, respectively. His research focuses on algorithms and architectures to enhance perceived video quality for video streaming, and resource allocation in device-to-device networks to optimize system and user perceived performance. He interned at Intel Labs, Hillsboro during summer 2013. Since 2015, he has been working at Cisco Systems, San Jose, CA, USA.

Gustavo de Veciana (S’88–M’94–SM’01–F’09) received the B.S., M.S, and Ph.D. degrees in electrical engineering from the University of California at Berkeley, Berkeley, CA, USA, in 1987, 1990, and 1993, respectively, and joined the Department of Electrical and Computer Engineering, where he is currently a Cullen Trust Professor of Engineering. He served as the Director and Associate Director of the Wireless Networking and Communications Group, University of Texas at Austin, from 2003 to 2007. His research focuses on the analysis and design of communication and computing networks; data-driven decision-making in man-machine systems, and applied probability and queueing theory. He served as an Editor and is currently serving as Editor-at-Large for the IEEE/ACM TRANSACTIONS ON NETWORKING. He was the recipient of a National Science Foundation CAREER Award 1996 and a co-recipient of five best paper awards including IEEE William McCalla Best ICCAD Paper Award for 2000, Best Paper in ACM TODAES January 2002–2004, Best Paper in ITC 2010, Best Paper in ACM MSWIM 2010, and Best Paper IEEE INFOCOM 2014. In 2009, he was designated IEEE Fellow for his contributions to the analysis and design of communication networks. He currently serves on the board of trustees of IMDEA Networks Madrid.