Joint Transmission Scheduling and Congestion Control for Adaptive Streaming in Wireless Device-to-Device Networks

D. Bethanabhotla, G. Caire, M. J. Neely
University of Southern California
Emails: {bethanab, caire, mjneely}@usc.edu

***Abstract*—We consider the jointly optimal design of a transmission scheduling and admission control policy for adaptive streaming over wireless device-to-device networks. We formulate the problem as a dynamic network utility maximization and observe that it naturally decomposes into two subproblems: admission control and transmission scheduling. The resulting algorithms are simple and suitable for distributed implementation. The admission control decisions involve each user choosing the quality of the video chunk asked for download, based on the network congestion in its neighborhood. This form of admission control is compatible with the current video streaming technology based on the DASH protocol over TCP connections. We also consider a mechanism for dropping bits from the transmission queues in order to obtain deterministic bounds on the queueing delays, which determine the number of video chunks that should be pre-fetched in order to guarantee smooth playback without interruptions.**

I. Introduction

We consider the problem of joint transmission scheduling and congestion control for adaptive video streaming in a single hop wireless device to device (D2D) network. We formulate a Network Utility Maximization (NUM) problem in the framework of Lyapunov Optimization, and derive algorithms for joint transmission scheduling and congestion control inspired by the drift plus penalty approach [1].

Excellent surveys on various problem formulations in the NUM framework can be found in [2]–[4]. Initial work in NUM focused on networks with static connectivity and time-invariant channels. One of the first applications of NUM was to show that Internet congestion control in TCP implicitly solves a NUM problem where the variant of TCP dictates the exact shape of the utility function [2], [4]. This framework has been extended to wireless ad-hoc networks with time varying channel conditions (see [5], [6] and the references therein) and to peer-to-peer networks in [7]. Recent work on adaptive video scheduling over wireless channels appears in [8], [9].

For the problem at hand, as a consequence of the NUM formulation, we obtain an elegant decomposition of the optimal solution into two separate subproblems, which interact through the queue lengths. The subproblems are solved by distributed dynamic policies requiring only local queue length information at each network node. The network is formed by *users*, which place video streaming requests and wish to download sequences of video chunks corresponding to the desired video files, and *helpers*, which contain cached video files and serve the user requests through the wireless channel. The transmission scheduling decisions are carried out independently by the *helpers*, while the admission control decisions are carried out independently by the *users*. In particular, we notice that the independent choices made by every user in deciding the quality of the video chunk that should be downloaded at any given time is compatible with the current technology based on client-driven *Dynamic Adaptive Streaming over HTTP* (DASH) for video on demand (VoD) systems [10], [11].

Motivated by realistic typical system parameters, we assume that the time and frequency selective wireless channel fading coherence time $\times$ bandwidth product is small with respect to the number of signal dimensions spanned by the transmission of a video chunk. This implies that the rate scheduling decisions in the transmission scheduling policy can make use of the "ergodic" achievable rate region of the underlying physical layer. In contrast, other system parameters such as the distance dependent path loss and the quality-rate tradeoff profile of the video file may evolve at the same time scale of the video chunks, yielding non-ergodic dynamics. Also, we have to consider that a coded video file is formed by "chunks" (group of pictures) whose statistics may change with time. Correspondingly, variable bit rate (VBR) video coding [12] yields a quality-rate tradeoff profile that may change with time (i.e., with the chunk index). We address these non-stationary and non-ergodic dynamics of the system parameters by providing performance guarantees for *arbitrary sample paths*, using the approach developed in [1], [13]. We also provide deterministic delay guarantees for the network queues by using the bit-dropping technique developed in [1], [14]. As a result, the users can pre-fetch and fill up their playback buffers with as many chunks as the maximum delay of their neighboring helpers' queues, and thereby prevent interruptions in playback.

II. System Model

We consider a discrete, time-slotted wireless network with multiple user stations and multiple helper stations. The network is defined by a bipartite graph $\mathcal{G} = (\mathcal{U}, \mathcal{H}, \mathcal{E})$, where $\mathcal{U}$ denotes the set of users, $\mathcal{H}$ denotes the set of helpers, and $\mathcal{E}$ contains edges for all pairs (h, u) such that there exists a potential transmission link between $h \in \mathcal{H}$ and $u \in \mathcal{U}$. We denote by $\mathcal{N}(u) \subseteq \mathcal{H}$ the neighborhood of user u, i.e., $\mathcal{N}(u) = \{h \in \mathcal{H} : (h,u) \in \mathcal{E}\}$. Similarly, $\mathcal{N}(h) = \{u \in \mathcal{U} : (h,u) \in \mathcal{E}\}$. $\mathcal{H}$ also includes the base station denoted as helper 0 and in this case $\mathcal{N}(0) = \mathcal{U}$ since the cellular base station can communicate to all users in the cell. Each user $u \in \mathcal{U}$ requests a video file f_u from a library of possible files. A video file is formed by a sequence of "chunks", i.e., group of pictures (GOPs), that are encoded and decoded as stand-alone units. Chunks have the same duration in time, given by $T_{\rm gop} = (\text{\# frames per GOP})/\eta$, where η is the frame rate (frames per second). Chunks must be reproduced in sequence at the user end. The streaming process consists of transferring chunks from the helpers to the requesting users such that the playback buffer at each user contains the required chunks at the beginning of each chunk playback time. Streaming is different from downloading because the playback starts while the whole file has not been entirely transferred. In fact, the playback starts after a short pre-fetching time, where the playback buffer is filled in by a determined amount of chunks. We assume that the scheduler time-scale coincides with the chunk interval, i.e., at each chunk interval a scheduling decision is made. Conventionally, we assume a slotted time axis $t = 0, 1, 2, 3 \ldots$, corresponding to epochs $t \times T_{\rm gop}$. Let T_u denote the pre-fetching delay of user u. Then, chunks are downloaded starting at time $t = 0$ and playback starts at time T_u. A stall event for user u at time $t \geq T_u$ is defined as the event that the playback buffer does not contain chunk number $t - T_u$ at slot time t.

Helpers have caches that contain subsets of the video files in the library. We denote by $\mathcal{H}(f)$ the set of helpers that contain file f. Hence, the request of user u for a chunk at a particular slot t can be assigned to any one of the helpers in the set $\mathcal{N}(u) \cap \mathcal{H}(f_u)$. Letting $N_{\rm pix}$ denote the number of pixels per frame, a chunk contains $k = \eta T_{\rm gop} N_{\rm pix}$ pixels (source symbols). We assume that each chunk of each file f is encoded at a finite number of different quality modes $m \in \{1, \ldots, N_f\}$ which is similar to what is typically done in several recent video streaming technologies like Microsoft Smooth Streaming and Apple HTTP Live Streaming [11]. Due to the variable bit-rate nature of video coding, the quality-rate profile may vary from chunk to chunk. We let $D_f(m,t)$ and $kB_f(m,t)$ denote the video quality measure and the number of bits for file f at chunk time t and quality mode m respectively. A fundamental function of the network controller at every slot time t consists of choosing the quality mode $m_u(t)$ of the chunks requested at t by all users u. The choice $m_u(t)$ renders the choice of the point $(D_{f_u}(m_u(t),t), kB_{f_u}(m_u(t),t))$ from the finite set of quality-rate tradeoff points $\{(D_{f_u}(m,t), kB_{f_u}(m,t))\}_{m=1}^{N_f}$. We let $R_{hu}(t)$ denote the source coding rate (bit per pixel) of chunk t received by user u from helper h. In addition to choosing the quality mode $m_u(t)$ for chunk time t for all requesting users u, the network controller also allocates the source coding rates $R_{hu}(t)$ satisfying

$$\sum_{h \in \mathcal{N}(u) \cap \mathcal{H}(f_u)} R_{hu}(t) = B_{f_u}(m_u(t),t) \ \forall \ (h,u) \in \mathcal{E}. \quad (1)$$

When $R_{hu}(t)$ is determined, helper h places the corresponding $kR_{hu}(t)$ bits in its transmission queue Q_{hu}, to be sent to user u within the queuing and transmission delays. Notice that in order to be able to download different parts of the same chunk from different helpers, the network controller needs to ensure that all received bits from the serving helpers $\mathcal{N}(u) \cap \mathcal{H}(f_u)$ are useful, i.e., the union of all received bits yields the entire chunk, without overlaps and without gaps. However, in Section III, we will see that even if we allow the possibility of downloading different parts of the same chunk from different helpers, the optimal scheduling policy is such that users download an entire chunk from a single helper, rather than obtaining different parts from different helpers. Thus, it turns out that the assumption of protocol coordination to prevent overlap or gaps of the downloaded bits from different helpers is not needed. The dynamics of the transmission queues at the helpers is given by:

$$Q_{hu}(t+1) = \max\{Q_{hu}(t) - n\mu_{hu}(t), 0\} + kR_{hu}(t) \qquad \forall \ (h,u) \in \mathcal{E} \quad (2)$$

where n denotes the number of physical layer channel symbols corresponding to the duration $T_{\rm gop}$, and $\mu_{hu}(t)$ is the channel coding rate (bits/channel symbol) of the transmission from helper h to user u. We model the point-to-point wireless channel for each $(h,u) \in \mathcal{E}$ as a frequency and time selective underspread [15] fading channel. Using OFDM, the channel can be converted in a set of N_c parallel narrowband sub-channels in the frequency domain (subcarriers), each of which is time-selective with a certain fading channel coherence time. We assume the widely adopted block fading model, where the small scale Rayleigh fading coefficient is constant over time-frequency "tiles" (resource blocks) spanning blocks of adjacent subcarriers in the frequency domain and blocks of OFDM symbols in the time domain. For example, in the LTE 4G standard, for an available system bandwidth of 18MHz (after excluding the guard bands) and a scheduling slot of duration $T_{\rm gop} = 0.5$s (typical GOP duration), we have that a scheduling slots spans 100×1000 such resource blocks, each of which is affected by its own fading coefficient. Thus, it is safe to assume that channel coding over such a large number of resource blocks can achieve the *ergodic capacity* of the underlying fading channel. [1] For simplicity, in this paper we consider constant power transmission, i.e., the serving helpers transmit with constant and flat power spectral density over the whole system bandwidth, irrespective of the scheduling decisions and of the instantaneous fading channel state. We further assume that every user u when decoding a transmission from a particular helper $h \in \mathcal{N}(u)$ treats the interference from other helpers as noise. Under these system assumptions, the maximum achievable rate for link $(h,u) \in \mathcal{E}$ is given by

$$C_{hu}(t) = \mathbb{E}\left[\log\left(1 + \frac{P_h g_{hu}(t)|a_{hu}|^2}{1 + \sum_{h' \neq h} P_{h'} g_{h'u}(t)|a_{h'u}|^2}\right)\right], \quad (3)$$

[1]This is the capacity averaged with respect to the first-order fading distribution, which has the operational meaning of an achievable rate only if coding across an arbitrarily large number of fading states is possible. In contrast, the non-ergodic "outage capacity" of the fading channel is relevant when a channel codeword spans a limited number of fading states, that does not increase with the channel coding block length.

where P_h is the transmit power of helper h, a_{hu} is the small-scale fading gain from helper h to user u and g_{hu} is the slow fading gain (pathloss) from helper h to user u. We assume that each helper h serves its neighboring users $u \in \mathcal{N}(h)$ using orthogonal FDMA/TDMA. Therefore, the set of rates $\{\mu_{hu}(t) : u \in \mathcal{N}(h)\}$ is constrained to be in the "time-sharing region" of the broadcast channel formed by helper h and its neighbors $\mathcal{N}(h)$. This yields the transmission rate constraint

$$\sum_{u \in \mathcal{N}(h)} \frac{\mu_{hu}(t)}{C_{hu}(t)} \leq 1 \ \forall \ h \in \mathcal{H}. \tag{4}$$

The slow fading gain $g_{hu}(t)$ models path loss and shadowing between helper h and user u, and it is assumed to change very slowly with time. We let $\omega(t)$ denote the network state at time t, i.e.,

$$\omega(t) = \{g_{hu}(t), (D_{f_u}(\cdot, t), B_{f_u}(\cdot, t)) : \forall \ (h, u) \in \mathcal{E}, \ u \in \mathcal{U}\}.$$

Let $A_{\omega(t)}$ be the set of feasible control actions, dependent on the current network state $\omega(t)$, and let $\alpha(t) \in A_{\omega(t)}$ be a control action, comprising the vectors $\mathbf{R}(t)$ with elements $kR_{hu}(t)$ of video coded bits, $\boldsymbol{\mu}(t)$ with elements $n\mu_{hu}(t)$ of channel coded bits and the quality modes $m_u(t) \ \forall \ u \in \mathcal{U}$. A control policy is a sequence of control actions $\{\alpha(t)\}_{t=0}^{\infty}$ where at each time t, $\alpha(t) \in A_{\omega(t)}$.

III. Problem Formulation and Control Policy Design

We now formulate the Network Utility Maximization problem. We assume that the utility of a particular user is measured by its time-averaged quality. We wish to maximize the total quality over all users subject to keeping the queues at every helper stable. Letting $D(t) = \sum_{u \in U} D_{f_u}(m_u(t), t)$, we obtain the following dynamic optimization problem:

$$\max \ \lim_{t \to \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\left[D(\tau)\right] \tag{5}$$

$$\text{subject to} \ \lim_{t \to \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\left[Q_{hu}(\tau)\right] < \infty \ \forall \ (h, u) \in \mathcal{E} \tag{6}$$

$$\alpha(t) \in A_{\omega(t)} \ \forall \ t \tag{7}$$

Let $\mathbf{Q}(t)$ denote the column vector of all queues at time t, and define the quadratic Lyapunov function $L(t) = \frac{1}{2}\mathbf{Q}^{\mathsf{T}}(t)\mathbf{Q}(t)$. Defining $\Delta(t) = \mathbb{E}\left[L(t+1)|\mathbf{Q}(t)\right] - L(t)$ as the drift on slot t, the *drift plus penalty* (DPP) policy [1] is designed to solve (5)-(7) by observing only the current queue lengths $\mathbf{Q}(t)$ and the current network state $\omega(t)$ on each slot t and then choosing $\alpha(t) \in A_{\omega(t)}$ to minimize a bound on

$$\Delta(t) - VD(t)$$

Here, $V > 0$ is a control parameter of the DPP policy which affects a utility-delay tradeoff. It is shown in [16] that the above minimization decomposes into two separate minimizations:

A. Admission Control

The admission control decision involves choosing $m_u(t)$, i.e., the quality mode for all requesting users u at chunk time t and $\mathbf{R}(t)$, i.e., the number of source-coded bits $kR_{hu}(t)$ that each user u must download from helpers $h \in \mathcal{N}(u)$, relative to the chunk requested at time t. These choices are made to minimize the quantity $\mathbf{R}^{\mathsf{T}}(t)\mathbf{Q}(t) - VD(t)$ which decomposes into separate minimizations for each user, equivalently expressed as: for each $u \in \mathcal{U}$, choose $m_u(t)$ and $R_{hu}(t) \ \forall \ h \in \mathcal{N}(u) \cap \mathcal{H}(f_u)$ to minimize

$$\sum_{h \in \mathcal{N}(u) \cap \mathcal{H}(f_u)} kQ_{hu}(t)R_{hu}(t) - VD_{f_u}(m_u(t), t) \tag{8}$$

with $\{R_{hu}(t)\}_{h \in \mathcal{N}(u) \cap \mathcal{H}(f_u)}$ satisfying (1). It is immediate to see that the above problem is solved by choosing the helper $h_u^* \in \mathcal{N}(u) \cap \mathcal{H}(f_u)$ with the smallest queue backlog $Q_{hu}(t)$, and assigning the entire requested chunk to h_u^*. Notice that in this way the streaming of the video file f_u may be handled by *different* helpers across the streaming session, but each individual chunk is downloaded from a single helper. Further, the quality mode $m_u(t)$ is chosen as

$$m_u(t) = \arg \min_{m \in \{1, \ldots, N_{f_u}\}} \left\{kQ_{h_u^* u}(t)B_{f_u}(m, t) - VD_{f_u}(m, t)\right\}. \tag{9}$$

This policy requires only that each user knows the *local information* of the queue backlogs of the helpers in its own neighborhood, in order to make a congestion control decision. This policy is reminiscent of the current video streaming technology referred to as DASH (Dynamic Adaptive Streaming over HTTP) [10], [11], where the client (user) progressively fetches a video file by downloading successive chunks, and making a decision on the source encoding quality based on its knowledge of the congestion of the underlying server-client connection.

B. Transmission Scheduling

The transmission scheduling decision involves choosing the channel coding rates $\mu_{hu}(t)$ that each helper h must assign to the users u in its neighborhood $\mathcal{N}(h)$. Thus, $\boldsymbol{\mu}(t)$ is the solution of

$$\max \sum_{h \in \mathcal{H}} \sum_{u \in \mathcal{N}(h)} Q_{hu}(t)\mu_{hu}(t) \tag{10}$$

$$\text{subject to} \sum_{u \in \mathcal{N}(h)} \frac{\mu_{hu}(t)}{C_{hu}(t)} \leq 1 \ \forall \ h \in \mathcal{H}, \tag{11}$$

where $C_{hu}(t)$ have been defined in (3). Since we assume that all helpers transmit continuously with fixed power and there is no power control, the choice of transmission rates made by one helper does not affect the set of achievable rates at another helper. Therefore, the above LP decomposes into separate LP's at each of the helpers. Further, it can be shown that for each $h \in \mathcal{H}$, the solution consists of scheduling the user $u_h^* \in \mathcal{N}(h)$ with the largest product $Q_{hu}(t)C_{hu}(t)$, and serve this user at rate $\mu_{hu_h^*}(t) = C_{hu_h^*}(t)$, while all other queues are not served in slot t.

IV. Deterministic delay guarantees and playback buffer pre-fetch sizes

The problem formulation in the previous section did not take into account the fact that the chunks must be delivered within their playback deadlines. In this section, however, we modify the DPP algorithm in order to enforce that the video chunks are delivered in order to meet their playback deadlines by introducing additional *bit drop decisions* at the queues Q_{hu}. We briefly outline the necessary modifications.

Let $kd_{hu}(t)$ denote the number of bits to be dropped from Q_{hu} at time t, where $d_{hu}(t)$ indicates the dropping rate expressed in bits per pixel. Consequently, the queue update equations get modified as:

$$Q_{hu}(t+1) = \max\{Q_{hu}(t) - kd_{hu}(t) - n\mu_{hu}(t), 0\} + kR_{hu}(t) \;\forall\; (h,u) \in \mathcal{E}. \quad (12)$$

We assume that the drop decision $d_{hu}(t)$ satisfies $0 \leq d_{hu}(t) \leq R_{\max}$ where $R_{\max}$ is a uniform bound on $B_f(N_f, t) \;\forall\; t$ and for all files f. To incorporate drop decisions into the utility function, we modify the network utility as follows

$$\lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \left(\mathbb{E}\left[D\left(\tau\right)\right] - \beta\mathbb{E}\left[d\left(\tau\right)\right]\right),$$

where we define $d(t) = \sum_{(h,u)\in\mathcal{E}} kd_{hu}(t)$. To ensure that the worst case delay is bounded for all queues Q_{hu}, we introduce the following *ϵ-persistent service queues* $\{Z_{hu}\}$ (see [1]) which is a virtual queue with (virtual) buffer evolution:

$$Z_{hu}(t+1) = \max\left\{Z_{hu}(t) + n\epsilon - n\mu_{hu}(t) - kd_{hu}(t), 0\right\} \times 1_{\{Q_{hu}(t) > n\mu_{hu}(t) + kd_{hu}(t)\}} \;\forall\; (h,u) \in \mathcal{E}, \quad (13)$$

where $\epsilon > 0$ is a constant. $Z_{hu}(t)$ has the same service process $n\mu_{hu}(t) + kd_{hu}(t)$ as $Q_{hu}(t)$ but has an arrival process that adds $n\epsilon$ bits whenever the actual backlog $Q_{hu}(t)$ is not cleared at slot t, i.e., whenever the condition $Q_{hu}(t) > n\mu_{hu}(t) + kd_{hu}(t)$ is satisfied. However, when the backlog $Q_{hu}(t)$ is cleared, the virtual queue Z_{hu} is reset to 0. This ensures that Z_{hu} grows as long as the original queue backlog $Q_{hu}(t)$ remains uncleared. We incorporate the *ϵ-persistent service queues* in the Lyapunov function and derive the modified DPP policy, which makes the same admission control decisions (8) and (9), has transmission scheduling decisions that change the weights $Q_{hu}(t)$ in (10) to $Q_{hu}(t) + Z_{hu}(t)$ and makes additional dropping decisions given as: For each $(h,u) \in \mathcal{E}$,

$$d_{hu}(t) = \begin{cases} R_{\max} & \text{if } Q_{hu}(t) + Z_{hu}(t) > V\beta \\ 0 & \text{if } Q_{hu}(t) + Z_{hu}(t) < V\beta \end{cases} \quad (14)$$

Deterministic delay guarantee

It is shown in [16] that if $n\epsilon < kR_{\max}$, then $Q_{hu}(t), Z_{hu}(t) \;\forall\; (h,u) \in \mathcal{E}$ are deterministically bounded by $Q_{\max} = V\beta + kR_{\max}$ and $Z_{\max} = V\beta + n\epsilon$ respectively for all time t. Further, it is shown that the maximum delay experienced in any of the queues Q_{hu} is $W = \left\lceil \frac{Q_{\max}+Z_{\max}}{n\epsilon} \right\rceil$ slots. Consequently, every user can pre-fill its playback buffer with W chunks to avoid any interruption in playback.

V. Algorithm Performance

It is shown in [16] that when $\omega(t)$ is i.i.d., the time average utility achieved by the DPP policy comes within $O(\frac{1}{V})$ of the optimal utility associated with (5)-(7) with a deterministic worst case delay of $O(V)$. Further, when $\omega(t)$ evolves arbitrarily, the time average utility achieved by the DPP policy over any arbitrary sample path comes within $O(\frac{1}{V})$ of the utility of a genie-aided T-slot look ahead policy with a deterministic worst case delay of $O(V)$.

VI. Numerical Simulation

We simulate the DPP algorithm for the simple topology shown in Figure (1) over an arbitrary sample path of events shown in Figure (2). There are 4 users (numbered 1 to 4) arbitrarily located at the edge of a 400m $\times$400m square cell requesting 4 different video files with a macro base station located at the centre. Further, there are 2 helpers operating at the cell edge serving these 4 users as shown in Figure (1). The helper transmissions interfere with each other while the BS operates over an orthogonal band. Each helper is assumed to serve only those users within a certain radius indicated by the small circles in Figure (1). The helpers are connected to a wired backhaul and have access to all the files. Users 1, 2, 3 and 4 request four different video files f_1, f_2, f_3 and f_4 respectively from the database in [17]. The details of each video are given in Table I. Each video file is a long sequence of GOP chunks each of duration $2/3$ seconds and with a frame rate of 24 frames per second. The GOP chunks are encoded using some form of scalable variable-bit rate coding. The quality of each GOP is measured using the *Structural SIMilarity* (SSIM) index [18] and is indicated for different quality modes of a sample GOP from f_1 in Table II. The physical layer system parameters are motivated by LTE specifications [19]. We set the parameters $V = 2\times10^8$, $\beta = 2$ and $n\epsilon = 2 \times 10^6$ in our simulations.

TABLE I
Video details

Sequence	Resolution	no. of layers	avg. bitrate (kbps)
f_1	352 × 288	8	631
f_2	704 × 576	4	3908
f_3	1024 × 576	4	6679
f_4	352 × 288	8	556

TABLE II
sample GOP quality levels

Quality mode	SSIM index
1	0.8577
2	0.87016
3	0.88476
4	0.89293
5	0.90883
6	0.91453
7	0.92269
8	0.92926

We examine the adaptabilty of the DPP policy to changing network conditions by simulating the algorithm for an arbitrary

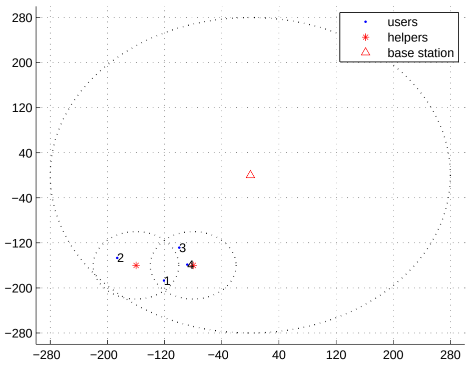

Fig. 1. Topology

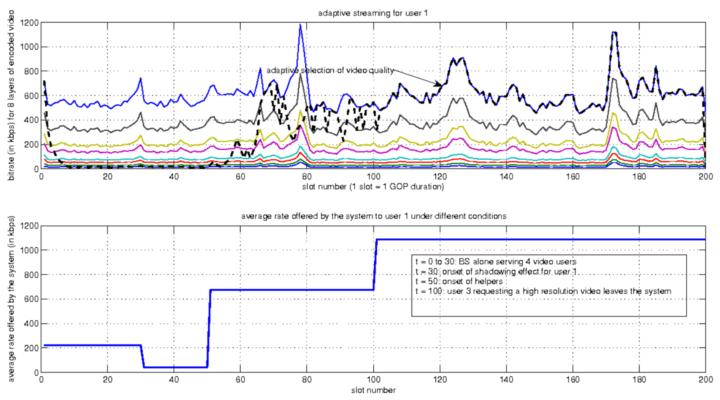

Fig. 2. Dynamic adaptive streaming

sample path of events. The bottom plot of Fig. 2 shows how the average rate offered by the system to user 1 through the DPP policy improves or degrades as the network condition changes. The sequence of events includes: onset of shadowing at $t = 30$, onset of helpers at $t = 50$ and user 3 who requests a high resolution video f_3 leaving the system at $t = 100$. In the top plot of Fig. 2, $B_{f_1}(m, t)$ is plotted over slot time t for all $m \in \{1, \dots, 8\}$. The black dashed line indicates how the DPP policy adaptively switches to downloading better quality chunks as the network condition improves. Also, W is calculated to be around 200 slots implying that interruptions in playback can be avoided if users can pre-fetch the initial 2 minutes of the video file.

VII. Acknowledgment

This work is supported in part by the Network Science Collaborative Technology Alliance sponsored by the U.S. Army Research Laboratory W911NF-09-2-0053. We also thank Hilmi Enes Egilmez and Prof. Antonio Ortega for several useful discussions and for providing scalably encoded variable bit-rate video files along with the corresponding SSIM index values.

References

[1] M. Neely, “Stochastic network optimization with application to communication and queueing systems,” *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.
[2] F. Kelly, “The mathematics of traffic in networks,” *The Princeton Companion to Mathematics*, 2006.
[3] Y. Yi and M. Chiang, “Stochastic network utility maximisation-a tribute to Kelly’s paper published in this journal a decade ago,” *European Transactions on Telecommunications*, vol. 19, no. 4, pp. 421–442, 2008.
[4] M. Chiang, S. Low, A. Calderbank, and J. Doyle, “Layering as optimization decomposition: A mathematical theory of network architectures,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 255–312, 2007.
[5] M. Neely, E. Modiano, and C. Rohrs, “Dynamic power allocation and routing for time-varying wireless networks,” *Selected Areas in Communications, IEEE Journal on*, vol. 23, no. 1, pp. 89–103, 2005.
[6] L. Chen, S. Low, M. Chiang, and J. Doyle, “Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks,” in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*. IEEE, 2006, pp. 1–13.
[7] M. Neely and L. Golubchik, “Utility optimization for dynamic peer-to-peer networks with tit-for-tat constraints,” in *INFOCOM, 2011 Proceedings IEEE*. IEEE, 2011, pp. 1458–1466.
[8] C. Chen, R. Heath, A. Bovik, and G. de Veciana, “Adaptive policies for real-time video transmission: A markov decision process framework,” in *Image Processing (ICIP), 2011 18th IEEE International Conference on*. IEEE, 2011, pp. 2249–2252.
[9] V. Joseph and G. de Veciana, “Jointly optimizing multi-user rate adaptation for video transport over wireless systems: Mean-fairness-variability tradeoffs,” in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 567–575.
[10] Y. Sánchez, T. Schierl, C. Hellge, T. Wiegand, D. Hong, D. De Vleeschauwer, W. Van Leekwijck, and Y. Lelouedec, “iDASH: improved dynamic adaptive streaming over http using scalable video coding,” in *ACM Multimedia Systems Conference (MMSys)*, 2011, pp. 23–25.
[11] A. Begen, T. Akgul, and M. Baugher, “Watching video over the web: Part 1: Streaming protocols,” *Internet Computing, IEEE*, vol. 15, no. 2, pp. 54–63, 2011.
[12] A. Ortega, “Variable bit-rate video coding,” *Compressed Video over Networks*, pp. 343–382, 2000.
[13] M. Neely, “Universal scheduling for networks with arbitrary traffic, channels, and mobility,” in *Decision and Control (CDC), 2010 49th IEEE Conference on*. IEEE, 2010, pp. 1822–1829.
[14] ——, “Delay-based network utility maximization,” in *INFOCOM, 2010 Proceedings IEEE*. IEEE, 2010, pp. 1–9.
[15] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. Cambridge Univ Pr, 2005.
[16] D. Bethanabhotla, G. Caire, and M. J. Neely, “Joint transmission scheduling and congestion control for adaptive streaming in wireless device-to-device networks,” *journal in preparation*.
[17] http://media.xiph.org/video/derf/.
[18] https://ece.uwaterloo.ca/~z70wang/research/ssim/.
[19] http://www.tsiwireless.com/docs/whitepapers/LTE%20in%20a%20Nutshell%20-%20Physical%20Layer.pdf.