Adaptive Video Streaming over Whitespace: SVC for 3-Tiered Spectrum Sharing

Xiaoli Wang, Jiasi Chen, Aveek Dutta, Mung Chiang
Princeton University, Princeton, New Jersey, USA
{xw4, jiasic, aveekd, chiangm}@princeton.edu

***Abstract*—The recently proposed 3-Tier access model for Whitespace by the Federal Communications Commission (FCC) mandates certain classes of devices to share frequency bands in space and time. These devices are envisioned to be a heterogeneous mixture of licensed (Tier-1 and Tier-2) and unlicensed, opportunistic devices (Tier-3) . The hierarchy in accessing the channel calls for superior adaptation of Tier-3 devices with varying spectral opportunity. While policies are being ratified for efficient sharing, it also calls for redesigning many common applications to adapt to this novel paradigm.**

In this paper, we focus on the ever-increasing demand for video streaming and present a methodology suitable for Tier-3 devices in the shared access model. Our analysis begins with a stress test of commonly adopted video streaming methods under the new sharing model. This is followed by the design of a robust MDP-based solution that proactively adapts to fast-varying channel conditions, providing better user quality of experience when compared to existing solutions, such as MPEG-DASH. We evaluate our solution on an experimental testbed and find that our MDP-based algorithm outperforms DASH, and partial information of Tier-2 dynamics improves video quality.

I. Introduction

In recent years, the FCC has addressed the dearth of spectrum through various rule-making efforts [1] that focus on sharing large swaths of frequencies, historically allocated to various federal and non-federal bodies. In this regard, it has been proposed that a prioritised access scheme should be adopted for sharing this common pool of resources, collectively called whitespace. While exclusive rights remain for the incumbent users of these bands, other devices and services can use those bands based on certain rules and policies. The architecture of such a sharing paradigm is unique compared to current architectures in practice, especially as it concerns the co-existence of Tier-2 (T2) and Tier-3 (T3) classes of devices.

Figure 1 shows an example of spectrum sharing between the two classes of users, where T2 networks are allowed to have a larger transmission range. T3 networks use the same frequency band in the vicinity, but are comprised of unlicensed and opportunistic devices. Being lower in priority, T3 devices will have to be adaptive in terms of their transmission ranges to prevent harmful interference to T2 clients. The degree of adaptation (low transmit power) depends on the proximity of T2 in the region where the transmission zones overlap, represented by varying ranges of T3 networks in Figure 1. Also, the frequency of arrival of T2 users will force T3 networks to adapt, often at a much smaller timescale than experienced in the current channelized mode of access.

In this paper, we focus on characterizing the collective effect of these two factors as experienced by a T3 client and study its effect on bandwidth demanding applications like video streaming. Now, a T3 client will not only experience time-varying interference from devices of its own class but also have to compensate for varying transmission ranges. This may also include spectrum outages. Designing video streaming applications for this class of devices is the goal of this work.

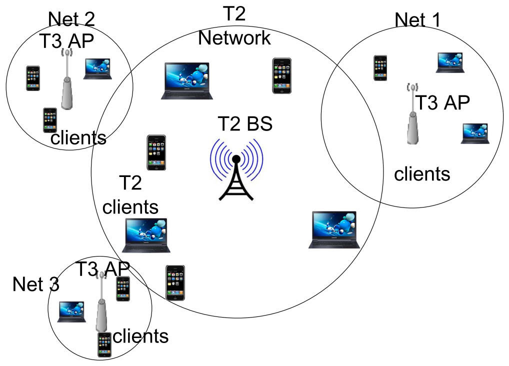


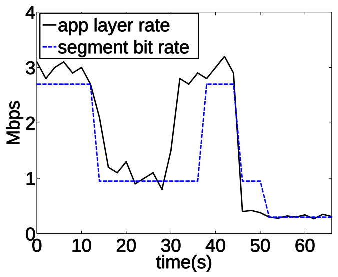
Fig. 1: Spectrum sharing in Whitespace. T3 users adapt to the dynamics of T2 clients by adjusting their transmission ranges.

Modern video streaming applications use adaptive techniques to combat variations in the throughput of the underlying channel. Dynamic Adaptive Streaming using HTTP (DASH) [2] , which was recently standardized, has been widely adopted in the video streaming industry, as it provides flexibility and reliability which leads to a high quality of experience (QoE) for the end user. DASH progressively request small *chunks* of video, called segments, of various video resolutions, while adapting to channel variations. This is implemented as a moving average of the application layer throughput observed over a finite timescale. Segments are requested based on the estimated throughput.

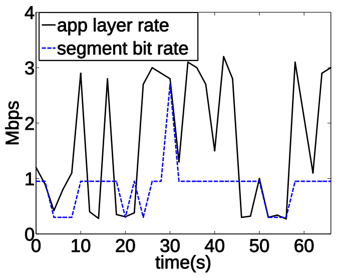
Clearly, the benefit of DASH depends on the smoothness of the estimated throughput and the availability of segments with multiple resolutions. This limits the performance gain of conventional DASH-based video streaming as it can make a decision for future segments only. Furthermore, since DASH acts on the average throughput estimated over previous time windows, it reacts slowly to changing channel conditions because the averaging operator smoothes these variations.

A. Problem Statement and Methodology

Fig. 2 shows the performance of adaptation in conventional DASH. If the application layer throughput estimated by the adaptation algorithm varies slowly, the bit-rate of the segments downloaded consistently follows the variation in throughput (Fig. 2a). However, if the channel changes quickly (Fig. 2b), the adaptation of the video segments fails to follow the maximum throughput offered by the channel, even if partial



(a) Slow-varying channel



(b) Fast-varying channel

Fig. 2: Performance of DASH in various channel conditions

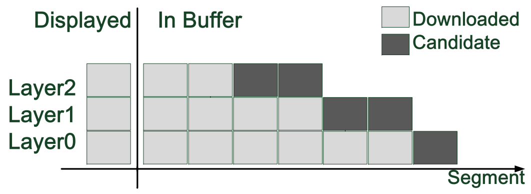


Fig. 3: DASH with SVC

information of the variation pattern is known, since DASH does not have the mechanism to handle this information. This leads to lost opportunities to download higher quality segments when the channel offers a higher throughput. *With partial information of channel condition, particularly T2 dynamics, can we do better than DASH in adapting to the fast-varying whitespace channel?* This is precisely the problem we will solve in this work, as follows.

We leverage the Scalable Video Coding (SVC) extension of the MPEG H.264 encoder [3]. With SVC, the raw video is encoded in various interdependent layers. In its simplest form, it consists of a Base Layer (BL) and multiple Enhancement Layers (EL). Although these layers are required to be multiplexed before decoding the video, they can be stored independently in smaller chunks, as are DASH-based streaming contents. The main benefit obtained from SVC is the ability to download EL for video segments at a later timestep if the corresponding BL is already in the buffer. This form of video storage and distribution is beneficial for the fast varying channel (Fig. 2b). Since SVC allows for incremental improvement of video segments over time, video streaming applications can optimize the sequence of the segments to download while maximizing the quality of each video segment before it is played back. Fig. 3 is a snapshot of the video buffer within the player which shows the possible candidate segments to request if there is enough throughput to download the particular layer. This provides various choices to the application, and the key challenge is to maximize user QoE using the flexibility of SVC.

To determine a preferred way to download a video, we use proactive adaptation. Because of the tiered architecture of whitespace and the scheduling role of a Spectrum Access System (SAS) [4], a T3 video streaming client may have prior information of T2 users' arrival and mobility patterns, and can use this to predict future throughput for video streaming. Considering the unpredictability of the whitespace channel, our proactive adaptation is Markov Decision Process (MDP) based, which chooses a segment bit rate that will maximize the expectated future video quality, unlike DASH which only considers the moving average of past throughput.

B. *Contributions*

Architecture: We design a network-aware video adaptation framework that uses future channel condition information to proactively adapt to the fast-varying whitespace channel.

Traffic model in whitespace: Based on the 3-tier model for whitespace, we use a two-state Markov chain to model the T2 dynamics, and estimate the available bandwidth pattern for a T3 user by considering T2 and T3 dynamics.

Algorithm: We design an MDP-based, proactive, adaptive video streaming algorithm using SVC and take signal strength as an input to detect current T2 ON/OFF-state.

Experiment: To test our MDP scheme, we implement a wireless system using two PCs and a channel emulator. We run real-time video streaming from a video server to a streaming client, and model the whitespace traffic in the channel emulator.

II. System Model

In this section, we show the fast-varying channel in whitespace by characterizing T3 throughput. Then, we describe our system architecture that allows video streams to adapt to the whitespace channel.

A. *Characterization of T3 Throughput*

Because of the coexistence of T2 and T3 users, the maximum transmitting power of a T3 access point (AP) depends on the location of T2 users. To simulate this scenario, we use a reverse mapping of the 802.16 Stanford University Interim (SUI) path loss model to find the maximum transmitting power of a T3 AP, given the distance between T2 and T3 users, the transmitting power of T2 base station (BS), and the minimum required receiving power of T2 clients. SUI is defined as $PL_{SUI} = A + 10\gamma \log_{10}(\frac{d}{d_0}) + X_f + X_h + s$ [5]. We choose the parameters based on an urban area, giving $PL_{SUI}(d) = 126 + 46.15 \log_{10}(\frac{d}{100})$ dB. Then we derive the throughput of a T3 user by considering (1) the dynamics of T2 users (ON/OFF) results in various SNR for all T3 users, and thus the variation in total throughput available to T3 users, and (2) a varying number of T3 users share the same channel using TDMA. Based on these factors, a sample throughput trace for a T3 user for 300s is shown in Fig. 4, which shows a fast-varying channel with frequent throughput fluctuation due to both arrival/departure and mobility of T2 users, and the varying number of T3 users sharing the same channel.

B. *Whitespace Adaptive Video Streaming Architecture*

Our architecture consists of three components: the video server, the tiered access model and the video player, as shown in Fig. 5. The server stores videos in SVC format as well as the corresponding Media Presentation Description (MPD) files for each video, which describes the video length and the segments available for each resolution. The whitespace tiered access channel is modeled based on the T2 and T3 dynamics in the previous section. At the client side, the video player runs our adaptation algorithm to select which video resolution and segment to request every segment duration slot. The segment selection algorithm runs based on the predicted future throughput from T2 dynamics pattern, and also the measurement of current signal strength. Each time a requested

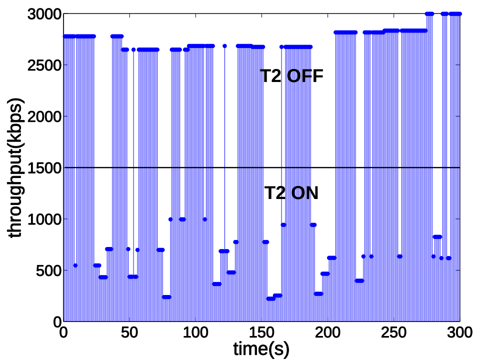


Fig. 4: A sample of T3 throughput. In this example, we have 5 T2 users randomly located in the T2 cell. The number of T3 users follows a normal distribution between 10 and 15.

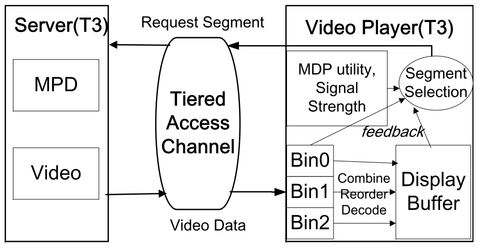


Fig. 5: System Overview

segment is downloaded, the video segments in each bin (buffer for each layer) and the display buffer will be updated, and the current number of segments in bins and the buffer become feedback for segment selection algorithm.

III. Problem Formulation

In this section, we formulate and solve the problem of choosing the video segments which maximize the user QoE over the duration of the video. We do this for two cases: (1) our proposed algorithm based on MDP, which is adaptive to uncertainty in the throughput variations of the T3 user, and also incorporates knowledge of T2 users if available, and (2) a simple scenario where the available throughput of a T3 user is known over time, serving as a benchmark. The notation in this section is given in Table I.

A. *Online Dynamic Video Adaptation Formulation*

We first tackle the problem of online dynamic video adaption to handle uncertainty in the future throughput. This uncertainty is key in whitespace networks, where T2 users can interfere with T3 users, so we first discuss the traffic model of T2 users.

T2 user dynamics: User population dynamics are generally modeled as a Poisson process [6]. Since the video requests by the application are periodic, we use a Bernoulli process, which is the discrete-time case of Poisson, to model the arrival/departure of T2 users. This forms a two-state Markov chain with states ON and OFF, denoting T2 user presence. The transition probability OFF→ON and ON→OFF are denoted as p_1 and p_2, respectively. Then the ON and OFF duration of T2 users follows a geometric distribution, with T2 presence duration(t_1) and absence duration(t_2) expressed as $P(t_1 = k) = (1-p_2)^{k-1}p_2$, and $P(t_2 = k) = (1-p_1)^{k-1}p_1$. The

N, T	total number of segments of video, number of iterations in MDP ($T \le N$)
Q_0, Q_1, Q_2	video quality(SSIM): low, medium, high
R_0, R_1, R_2	video rate for low, medium, high resolution
$S(n, r)$	size of video segment n of resolution r, $r \in \{0, 1, 2\}$
$S(r)$	average size of a video segment of resolution r, $r \in \{0, 1, 2\}$
$q_r = log[k(1+r)]$	utility of resolution r for a segment, $k \ge 10$
$R(m)$	throughput at time m
τ	playback length of a segment (segment duration)
C_r^t	number of segments in bin r at time t
A_r^t	number of segments to request for resolution r for the tth time slot
N_r^t	number of segments successfully downloaded for resolution r for the tth time slot, $N_t \le A_t$
$U(t)$	utility at time t
B	size of each bin in video segments
B_r	required minimum number of segments in bin r
S, A	states and available actions in MDP
S_A	all possible states after taking action A from state S in MDP
R_L, R_H	Average throughput of T3 when T2 is ON and OFF

TABLE I: Table of notation.

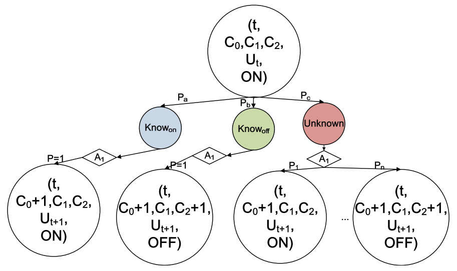
combined effect of T2's interference and T3 varying transmit power (to avoid interference to T2 users) results in varying SNR at the T3 receiver.

Markov decision process: At the beginning of each time interval, the algorithm must choose: (a) which video segment, and (b) which quality level to request. The optimal action for the current time interval is based on the available throughput, which is unknown but can be estimated from the probabilistic user dynamics model previously described. The online dynamic video adaptation can naturally be formulated as a finite horizon Markov Decision Process (MDP), where the actions are the video segments to be requested by the T3 user, and the state transition probabilities are based on T2 users' dynamics.[1]

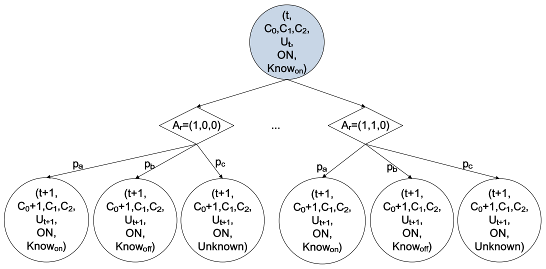
An example of the states of the MDP is shown in Fig. 6a. There are two types of states, called "super-states" and "sub-states"(colored). Each super-state is described by a tuple with elements: (1) t, the current video display time; (2) C_0, C_1, C_2, the current number of segments in bins 0, 1, and 2 respectively; (3) U_t, the utility, which will be formulated below; and (4) ON/OFF, whether T2 is currently present. Then there are three sub-states depending on whether T2 ON/OFF information in the next state is known in this super-state, with probability p_a, p_b and p_c. The actions here are how many segments to request for each layer, denoted as A. An example of A is $(1, 0, 0)$, which means requesting 1 segment of layer0 (L0), requesting none for layer1(L1) or layer2 (L2). If T2 ON/OFF information is known, as shown in the first two sub-states, then the next super-state is fixed based on an action. However, if T2 ON/OFF information is unknown, with the same action A_1, the next super-state has several candidates with transition probability based on possible throughput ranges.

To simplify MDP, we fit the three sub-states into the super-state by adding one more element $\{Know_{on}, Know_{off},$

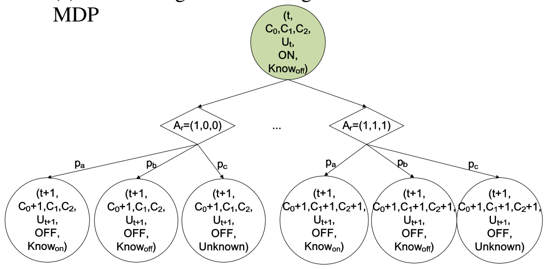
[1]The markov property arises because the buffered segments in the next state only depends on the buffered segments in current state and the action taken in this state. In addition, segment requests in a future state only depend on the buffered segments and T2 dynamics in that state, independent of segment requests in previous states. The finite horizon arises because the video has finite length.


(t, C0,C1,C2, Ut, ON)
Pa
Pb
Pc
Knowon
Knowoff
Unknown
P=1
A1
P=1
A1
A1
P1
Pn
(t, C0+1,C1,C2, Ut+1, ON)
(t, C0+1,C1,C2+1, Ut+1, OFF)
(t, C0+1,C1,C2, Ut+1, ON)
...
(t, C0+1,C1,C2+1, Ut+1, OFF)

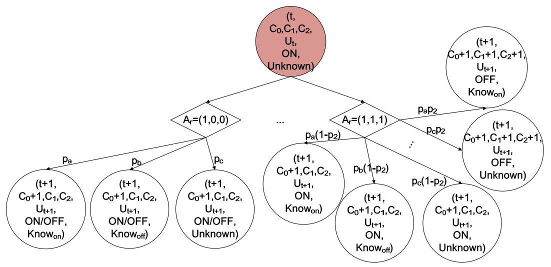
(a) Video Segment Fetching as a Finite Horizon MDP


(t, C0,C1,C2, Ut, ON, Knowon)
Ar=(1,0,0)
...
Ar=(1,1,0)
pa
pb
pc
(t+1, C0+1,C1,C2, Ut+1, ON, Knowon)
(t+1, C0+1,C1,C2, Ut+1, ON, Knowoff)
(t+1, C0+1,C1,C2, Ut+1, ON, Unknown)
(t+1, C0+1,C1,C2, Ut+1, ON, Knowon)
(t+1, C0+1,C1,C2, Ut+1, ON, Knowoff)
(t+1, C0+1,C1,C2, Ut+1, ON, Unknown)

(b) Case1: T2 is known to be on


(t, C0,C1,C2, Ut, ON, Knowoff)
Ar=(1,0,0)
...
Ar=(1,1,1)
pa
pb
pc
(t+1, C0+1,C1,C2, Ut+1, OFF, Knowon)
(t+1, C0+1,C1,C2, Ut+1, OFF, Knowoff)
(t+1, C0+1,C1,C2, Ut+1, OFF, Unknown)
(t+1, C0+1,C1+1,C2+1, Ut+1, OFF, Knowon)
(t+1, C0+1,C1+1,C2+1, Ut+1, OFF, Knowoff)
(t+1, C0+1,C1+1,C2+1, Ut+1, OFF, Unknown)

(c) Case2: T2 is known to be off


(t, C0,C1,C2, Ut, ON, Unknown)
Ar=(1,0,0)
...
Ar=(1,1,1)
pa
pb
pc
(t+1, C0+1,C1,C2, Ut+1, ON/OFF, Knowon)
(t+1, C0+1,C1,C2, Ut+1, ON/OFF, Knowoff)
(t+1, C0+1,C1,C2, Ut+1, ON/OFF, Unknown)
pa(1-p2)
pb(1-p2)
pc(1-p2)
pap2
pcp2
(t+1, C0+1,C1,C2, Ut+1, ON, Knowon)
(t+1, C0+1,C1,C2, Ut+1, ON, Knowoff)
(t+1, C0+1,C1,C2, Ut+1, ON, Unknown)
(t+1, C0+1,C1+1,C2+1, Ut+1, OFF, Knowon)
(t+1, C0+1,C1+1,C2+1, Ut+1, OFF, Unknown)

(d) Case3: T2 ON/OFF is unknown

Fig. 6: One-state transition in MDP. The last three figures are the detailed expansion of the colored sub-states in the first figure.

$Unknown\}$ indicating whether T2 ON/OFF information in the next state is known. Fig. 6 shows the detailed one-state transition by considering three starting states: $Know_{on}$, $Know_{off}$ and $Unknown$. In each state $S(t)$, the utility for taking action A_r^t is defined as:

$$U(t, A_r^t) = R(S_t, A_r^t) + \boldsymbol{P}^T \boldsymbol{U}(t+1) \tag{1}$$

where $R(S_t, A_r^t) = \sum_{r=0}^{L} N_r^n q_r$ is the current reward of taking action A_r^t in state $S(t)$ at time t. As defined in Table I, N_r^t is the number of segments successfully downloaded for each resolution, which depends on A_r^t and the actual current throughput. $\boldsymbol{P}$ is the probability transition vector from current state to all states after taking A_r^t, and $\boldsymbol{U(t+1)}$ is the vector of utilities in all states at next stage $t+1$. The current utility is the summation of current reward and expected utility in the next stage.

The MDP problem is to maximize this utility function in every time interval t, with the constraint of buffer size and available throughput:

$$\begin{aligned} &\underset{A_r^t}{\text{maximize}} \; U(t, A_r^t) \\ &\text{subject to } B_r \leq C_r^t + A_r^t \leq B \\ &\qquad S(r) A_r^t \leq \tau R(t) \\ &\qquad C_1^t + A_1^t < C_0^t + A_0^t \\ &\qquad C_2^t + A_2^t < C_1^t + A_1^t \end{aligned} \tag{2}$$

The first constraint means the number of segments in each bin cannot exceed the bin size B, and must be larger than the minimum segment number B_r in Bin r, where $r \in \{0, 1, 2\}$. The second constraint means the total requested segment size (candidate actions) cannot exceed the available throughput. Also, in SVC, lower layers should be requested before higher layers, which is represented by the last two constraints.

B. MDP-based Adaptation Algorithm

For a finite horizon MDP, the utility at each state $U(t)$ can be calculated offline. In an online algorithm, the goal is to maximize the expected utility for all future states.

Offline Utility Calculation: We construct a utility table U of size $T \times 6B^3$, where the t^{th} row stores the utility $U(t)$ for different cases at time t. $6B^3 = 2 \times 3 \times B^3$ is the total number of states, where 2 indicates the element ON/OFF, 3 represents $Know_{on}$, $Know_{off}$ and $Unknown$ of T2 dynamics in the next state, B is the buffer size of each bin, and we have three bins for three layers. The table size is proportional to video length, and only needs to be calculated once per video. The optimization problem is the same as (2), with the utility function

$$U(t, A_r^t) = R(S_t, A_r^t) + E[U(t+1)] = R(S_t, A_r^t) + \boldsymbol{P}^T \boldsymbol{U}(t+1) \tag{3}$$

Then all the utilities at different states and different time can be calculated using dynamic programming(DP), as follows.

The solution of the DP depends on the boundary cases. In our algorithm, there are only three cases for $\{C_0, C_1, C_2\}$ in the end state: $\{1, 0, 0\}$, $\{1, 1, 0\}$ and $\{1, 1, 1\}$. The utility only depends on which layer is fetched, and thus is defined as

$$U_{end} = \sum_{r=0}^{L} N_r^{end} q_r \tag{4}$$

Algorithm 1 Offline Utility Calculation

```
Input: T2 ON/OFF transition probability $p_1$, $p_2$; probability
    of knowing T2 ON/OFF for the next $\tau$ seconds $p_a$, $p_b$, $p_c$
Output: vector $\boldsymbol{U}(t)$ for each state at state t
1: Construct state transition reward matrix $\mathbf{R}$ of size $S \times S$
2: Construct state transition probability matrix $\mathbf{P}$ of size $S \times S \times A$
3: Calculate $\boldsymbol{U_{end}}$ using Eqn. (4)
4: Using DP, do backward calculation of previous $\boldsymbol{U}(t)$.
   The utility of state $S$ at stage $t$ is $U(t, S) = \max_A \sum_{S_A} P(S, S_A, A)\{R(S, S_A) + U(t+1, S_A)\}$
```

Online Segment Request: The utility at each state in each time interval $\boldsymbol{U(t)}$ is found by mapping to the offline utility table U. Among all actions A_r^t (the number of segments to request for each layer in time interval t), the action that maximizes the summation of current reward and expected future utilities is selected. This guarantees that if the algorithm is run over a number of realizations, the average utility over all realizations is the optimal. However, for every single realization, this MDP method does not guarantee the best action, as a single realization of T2 dynamics may differ from the general T2 transition probabilities. For computational complexity, though the number of states in the utility table U is large, the utility of each state can be stored at a unique location. Then looking for the action at a particular state only takes $O(N)$ time, where N is the video length in segments.

Algorithm 2 MDP-based adaptation using SVC

```
Input: bin level (in video segments); current signal strength $SS_c$; Segment duration $\tau$.
Output: video quality $Q(n)$ for each segment $n$, $n \in \{1,...,N\}$, $Q(n) \in \{Q0, Q1, Q2\}$.
1: for $t = 1 : T$ do
2:     Detect the current state by checking whether T2 is ON/OFF based on the value of $SS_c$, and the current bin level $C0, C1, C2$.
3:     Find the best action for the current state at time $t$ from the table of actions.
4:     Request $A_r$ according to the best action.
5:     Discard unfinished segments in the current request if $\tau$ seconds passed.
6: end for
```

C. Incorporating partial information about T2 users

In the previous section, the information we have about T2 ON/OFF is only for the next τ seconds. The online algorithm we developed depends on time-invariant transition probabilities between states so that the MDP model can be used. In practice, however, T3 users may have additional information about the arrival and departure of T2 users, since T2 users can register in the FCC database. We can therefore incorporate this additional information about T2 users into our model to improve the performance of the T3 users.

Suppose we have exact T2 ON/OFF knowledge for the next $\alpha N\tau$ seconds (α of entire video length, $\alpha \in [0, 1]$, depending on the percentage of T2 users that register in the database). Then the exact state transition patterns are fixed for these $\alpha N\tau$ seconds. In the extreme case, exact T2 ON/OFF times are known for the entire video ($\alpha = 1$), providing perfect information of T2 dynamics. The fixed transition patterns in the known period make the MDP non-stationary, because the transition probabilities change over time. The offline utility calculation needs to be modified to fit this non-stationary MDP, as shown in Algorithm 3. During the period where T2 dynamics are known, instead of calculating the expected future utilities based on T2 transition probabilities, we calculate the future utilities after taking each action based on the known T2 ON/OFF pattern. In the unknown period, the utility calculation is the same as in Algorithm 1. The utility table should be updated whenever there is a new known T2 pattern, even if the general T2 dynamics and video data are unchanged.

D. Benchmark: Offline Optimal Video Quality Selection

In this section, we consider an offline problem with known throughput over the entire video duration, to benchmark the

Algorithm 3 Utility Calculation for non-stationary MDP

```
Input: T2 ON/OFF transition probability $p_1$, $p_2$; probability of knowing T2 ON/OFF for the next $\tau$ seconds $p_a$, $p_b$, $p_c$; exact T2 ON/OFF for period $\Omega = \alpha N\tau$.
Output: vector $\boldsymbol{U}(t)$ for every state at state t
1: Construct state transition reward matrix $\mathbf{R}$, with size $S \times S$.
2: Construct state transition probability matrix $\mathbf{P}$, with size $S \times S \times A$.
3: Calculate $\boldsymbol{U_{end}}$ using Eqn. (4).
4: Using DP, do backward calculation of previous $\boldsymbol{U}(t)$. The utility at state $S$ at stage $t$ is $U(t, S) =$
   $\begin{cases} \max_A \{R(S, S_A) + U(t+1, S_A)\} & t \in \Omega \\ \max_A \sum_{S_A} P(S, S_A, A)\{R(S, S_A) + U(t+1, S_A)\} & t \notin \Omega \end{cases}$
```

video quality. The question is, given perfect knowledge of the T3 user's available throughput over time, how should the T3 user request video segments? If the available throughput is generally low, then the user should request the low quality version of each video segment. If the available throughput is slightly higher, then the user should request higher quality versions of some video segments, in addition to low quality versions of all video segments. Thus, there are two decision variables: which video segments should be requested, and at what quality level. This is formulated as a video quality maximization problem in (5).

$$
\begin{aligned}
&\underset{N_r^n}{\text{maximize}}\ U_{global} = \sum_{n=1}^{N}\sum_{r=0}^{L} N_r^n q_r \\
&\text{subject to } n \le \sum_{m=1}^{n} N_0^m \le n + B, n \in [1, N] \\
&\qquad C_r^n + N_r^n \le C_{r-1}^n + N_{r-1}^n, r \in [1, L] \\
&\qquad N_r^n S(r) \le R(n), n \in [1, N] \\
&\qquad \sum_{n=1}^{N} N_0^n = N
\end{aligned} \tag{5}
$$

$L + 1$ is the number of layers in a video. Here we consider a video with three layers (three resolutions), so $L = 2$. C_r^n is the current number of segments for layer r at time n, which is updated as $C_r^{n+1} = C_r^n + N_r^n$. The objective function is to maximize utility of all segments in the video. $q_r = log[k(1 + r)]$ is the utility of layer r of a segment, as defined in Table I. The concave log function is to illustrate that a user has less incentive to improve video quality if the video quality is already relatively high. The first constraint says the lowest layers must be request before its playback time, and the buffered segments cannot exceed buffer size B. The second constraint says that lower layers must be requested before higher layers, due to the layer dependency of SVC. The third constraint says that the size of all segments requested in a time slot cannot exceed the total throughput in this this time slot. The last constraint says the total number of segments of lowest layer should be video length.

This problem is a linear programming problem, but is complicated by the periodic buffer update and the time-dependency between the video request and the video playback time. To simply this problem, we divide the problem into subproblems and use dynamic programming to solve this problem, as this problem has optimal substructure and the optimization can be done for every segment request. For each suboptimal problem (every segment request), it is the same as (5) except that the objective function becomes $\sum_{r=0}^{L} N_r^n q_r$.

E. Optimality of MDP-based algorithm

In this section, we show the optimality of our MDP-based algorithm for two cases: (1) imperfect information of T2 dynamics, when only the Markov transition probability of T2 dynamics is known; and (2) perfect information, when T2 arrival/departure for the entire video duration is known ahead of time. Proofs of all the propositions can be found in our technical report.[7]

Proposition 1. *The optimal policy of MDP based algorithm in section III-B maximizes the expected utility $E(U_{global})$ in the offline global optimization problem (5).*

Proposition 2. *The lower bound of the MDP performance on a single realization is $\frac{(N-N1_{MPD})Q_0+N1_{MPD}Q_1}{(N-N1_{opt})Q_0+N1_{opt}Q_1}\%$ of optimal , $N1_{opt} = \lfloor\frac{N(R_L-R_0)}{R_1}\rfloor$, $N1_{MDP} = \lfloor\frac{N\frac{p_1}{p_1+p_2}R_L-NR_0}{R_1}\rfloor$. If T2 dynamics distribution in a realization differs from expected distribution by p_D, the performance will degrade at most by $p_D(1-p_{OFF}\frac{Q_0}{Q_2})$.*

Proposition 3. *With perfect T2 information over the entire video duration, the non-stationary MDP method in III-C provides a solution to problem (5), the offline global optimization problem.*

F. Stability of Video Quality

In previous sections, only video quality is considered. However, the stability of video quality is also important to QoE. In this section, we modify the utility function in MDP to reduce temporal variations of video quality.

Instead of using the summation of video quality of each segment, we use the geometric mean of video quality of all segments in the buffer as the utility function (shown below). Geometric mean not only maximizes video quality, but also minimizes quality variance among these segment[8].

The utility function is the same as 1, except the reward function becomes

$$R(S_{t-1}, S_t) = [(q_2)^{C_2^t}(q_1)^{C_1^t - C_2^t}(q_0)^{C_0^t - C_1^t}]^{\frac{1}{C_0^t}} \quad (6)$$

which is the reward in transition from state $S_{t-1} = \{C_0^{t-1}, C_1^{t-1}, C_2^{t-1}\}$ to state $S_t = \{C_0^t, C_1^t, C_2^t\} = \{C_0^{t-1} + N_0^t, C_1^{t-1} + N_1^t, C_2^{t-1} + N_2^t\}$, resolution $r \in \{0, 1, 2\}$. Then the boundary case becomes

$$U_{end} = [(q_2)^{C_2^{end}}(q_1)^{C_1^{end} - C_2^{end}}(q_0)^{C_0^{end} - C_1^{end}}]^{\frac{1}{C_0^{end}}}$$

IV. Performance Evaluation

In this section, we evaluate the performance of our scheme on a real testbed by comparing it with three alternatives: (1) an offline optimal algorithm that knows all throughput variations for the entire video duration, (2) a conventional DASH algorithm that requests a resolution based on moving average of past throughput, and (3) a modified DASH that knows T2 ON/OFF for the next segment length τ with probability P_{know}. We also explore the effects of T2 users' density and arrival frequency on video quality, as well as the sensitivity of our algorithm to varying levels of knowledge on T2 dynamics.

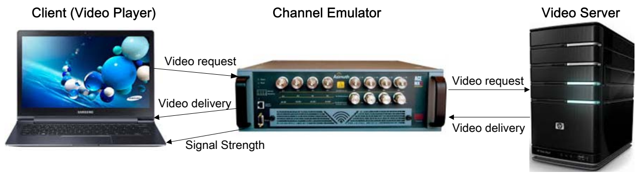
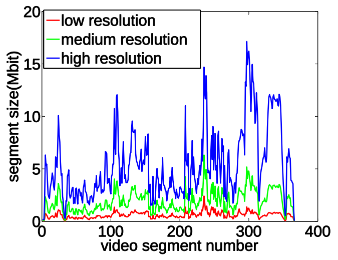
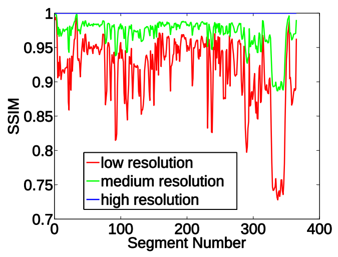


Fig. 7: Experiment Setup



(a) Segment size of raw video data



(b) SSIM of the video data

Fig. 8: Experiment video data for three resolutions

A. Experiment Setup

We implement and evaluate our algorithm using the setup in Fig. 7, where the Wi-Fi cards of the client and the video server are connected to the channel emulator [9]. We use Wi-Fi band instead of 3.5GHz band because the main difference in whitespace is the tiered access sharing architecture and the interactions between T2 and T3 users. Our method detects T2 presence by checking signal strength at T3 clients, so the propagation property, which depends on the frequency band for communication, does not affect too much in our T3 small cell of several hundred meters. The channel emulator uses Extended Pedestrian A model (EPA) to emulate the multipath fading propagation condition. The effect of T2 dynamics is programmed in the channel emulator by setting the transmitting power of the T3 access point, because T2 receivers at different locations will result in different maximum transmitting power of the T3 access point. The effect of T3 dynamics, which is the varying number of T3 users in a T3 cell, is programmed in the server PC by controlling the round-trip delay time. A large number of T3 users means a long round-trip delay. The video server stores video segments in both MPEG-DASH and SVC formats, as well as the MPD files for each format. The video streaming client is a QT video player using the open-source libdash library [10], and we modify the video player to support SVC by adding additional buffer bins for each layer and segment reordering function. The video we used in the experiment is a 12-minute short action film "Tears Of Steel" [11]. The SVC format of the video is encoded into three resolutions using spatial scalability (Table II and Fig. 8). The DASH format of the video is encoded using MPEG-DASH with the same resolutions as SVC.

B. Comparing Adaptive Streaming Algorithms

We assume that the Markov transition probabilities of T2 p_1, p_2 are always known. We first run the optimal offline

TABLE II: Video Data

Resolution	Average bit rate	Maximum bit rate	Average SSIM
320 × 180	0.29Mbps	1.28Mbps	0.9145
640 × 360	0.95Mbps	3.37Mbps	0.9705
1280 × 720	2.67Mbps	10.46Mbps	1

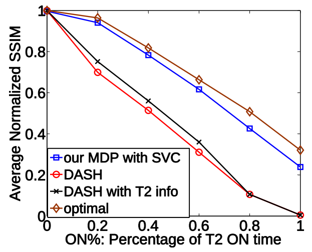
video quality selection as a video quality benchmark, which solves (5) with the knowledge of throughput variation over the entire video duration. This provides the optimal solution that maximizes the overall video quality by aggressively requesting the best video quality when the previous request is finished. However, this does not consider temporal variation in video quality (stability of video quality), so we call this the "optimal quality". For the other three cases, video request is made every τ seconds (segment length). In conventional DASH, video adaptation is reactive to throughput fluctuations and does not take future throughput variations into account. To make a fair comparison of our algorithm and DASH, we modified the DASH adaptation to make DASH know exact T2 ON/OFF for the next τ seconds with probability P_{know}. As a result, we call this "DASH with T2 information". However, the adaptation is still myopic, because the adaption only sees T2 dynamics for the next τ seconds, but our algorithm considers the expected T2 ON/OFF for the entire video duration based on the markov transition probability of T2 dynamics.

To analyze video quality, we use the average SSIM value of a video. SSIM is a metric to measure the similarity of two images, which was designed to improve on traditional methods like peak signal-to-noise ratio (PSNR) and mean squared error (MSE) [12], and is more consistent with user QoE. Typically, it is calculated on window size of 8×8. The measure between two windows x and y is calculated as $SSIM(x,y) = \frac{(2\mu_x\mu_y+c_1)(2\delta_{xy}+c_2)}{(\mu_x^2+\mu_y^2+c_1)(\delta_x^2+\delta_y^2+c_2)}$, where μ_x, μ_y, δ_x^2, δ_y^2 are the mean and variance of x and y, c_1 and c_2 are two variables to stabilize the division with weak denominators. SSIM is a decimal value between -1 and 1, and 1 denotes the case of two identical images. In our video data set, the average SSIM for low resolution is 0.9145, for high resolution is 1. To provide better visualization, we use the normalized SSIM $SSIM_{norm} = (SSIM - 0.9145)/(1 - 0.9145)$, which is 0 when SSIM is 0.9145, and is 1 when SSIM is 1.

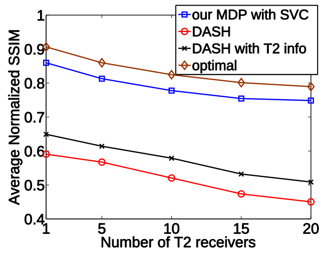
C. Effect of T2 dynamics on video quality

The effect of T2 dynamics on video quality is considered in two ways: (1) T2 density: the number of T2 receivers when T2 is ON, (2) the percentage of time that T2 is ON. The result of the experiment using the four methods is shown in Fig. 9a and 9b, respectively.

In Fig. 9a, when there are no T2 users, all methods can request the highest resolution and provide the same performance. As the percentage of T2 ON time increases, the video quality decreases, but the gap between our algorithm and the optimal increases substantially slower than the gap between DASH and the optimal (about 3 times slower). In the extreme case when T2 is always ON, the two DASH can only request the low resolution, but our algorithm can still request some segments with medium resolution. This is because in DASH, if the throughput is between the low and medium resolution bit rates, DASH always requests low resolution. However, our MDP uses SVC, where the medium resolution is the combination of base layer and enhancement layer, and thus will select some enhancement layers to improve those low resolution segments in bins. Our method is very close to the "optimal quality", and outperforms "DASH" and "DASH with T2 information" by 20% on average.

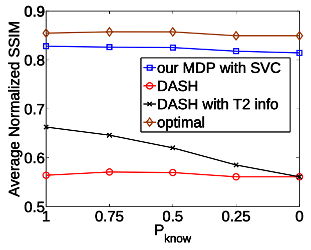


(a) Video quality vs. T2 ON%. X axis is the percentage of video duration that T2 is ON.

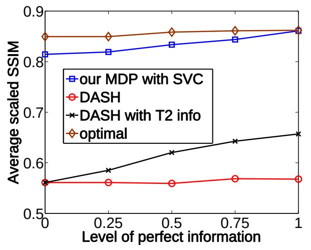


(b) Video quality vs. Density of T2 users. X axis is the number of T2 users when T2 is ON.

Fig. 9: Percentage of T2 ON time has more effect than the number of T2 users on video quality.



(a) Video quality with varying P_{know} (the probability of knowing exact T2 ON/OFF for the next segment duration).



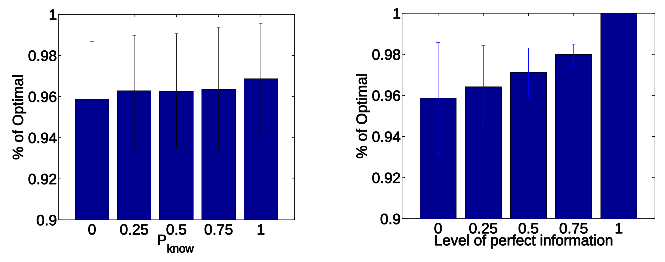
(b) Video quality with different level of perfect information. X axis is the percentage of video duration that T2 dynamics is known.

Fig. 10: Video quality with partial T2 information. Knowledge of T2 ON/OFF only for the next state does not impact our method, but impacts DASH. Knowing more of T2 ON/OFF for a set of continuous states improves our performance.

In Fig. 9b, as the number of T2 increases, video quality decreases slightly. This is due to the fact that the maximum transmitting power of T3 depends on the closest T2 receiver location. The more the number of T2 users, the more likely that some T2 users are close to the T3 cell. The video quality gap between our algorithm and the optimal keeps almost the same as T2 density increases, but the quality gap between DASH and the optimal increases as T2 density increases. This is because when the density of T2 users increases, the actual available throughput to T3 video streaming will be a slightly lower than the expected throughput. In this case, our MDP method using SVC will always finish downloading lower layers in our request queue, and if the available throughput is not enough to download all requested layers, only higher layers will be discarded. However, in DASH, video segments are requested by resolutions, not layers. If the available throughput is not enough to download the requested high resolution segment, the whole segment has to be discarded, and then the low resolution segment is requested again in the next time interval.

D. Robustness Analysis

To test the robustness of our algorithm to the level of T2 ON/OFF information in the next state, we run the experiment with different P_{know} (Fig. 10a). As P_{know} decreases, the performance of our algorithm decreases only slightly, meaning that our algorithm is robust with different values of P_{know}. This is because our MDP-based algorithm not only considers the exact T2 ON/OFF for the next state (next τ seconds), but also adapts to the general T2 dynamics for the entire video



(a) % of knowing T2 ON/OFF in the next state (b) perfect information of T2 ON/OFF for % of video length

Fig. 11: Improvement of MDP method from T2 information.

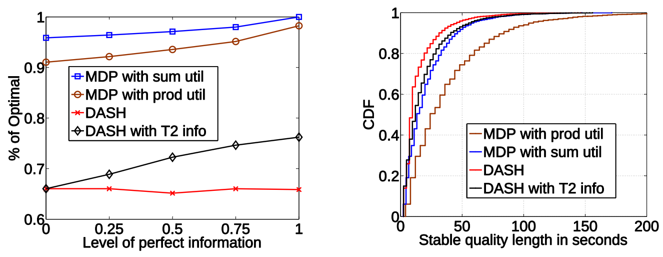
duration based on the Markov transition probabilities of T2 ON/OFF. The exact T2 ON/OFF in a state is a realization of the general T2 dynamics model, so without exact T2 information for the next state, our algorithm can choose the best action based on expected T2 dynamics for the entire video duration. In comparison, conventional DASH is the worst, because it does not have the ability to predict/use future throughput information, even if T2 dynamics in the future is given. If DASH is modified to adapt to the known exact T2 ON/OFF for the next τ seconds, then the video quality can be improved, but still worse than our method using MDP with SVC. In the case of "DASH with T2 information", the performance largely depends on P_{know}: the more information that is provided about T2 dynamics for the next τ seconds, the better the performance.

E. *Perfect vs. Imperfect Information*

In this section, we analyze whether perfect information of T2 dynamics for the entire video duration can help improve the video quality. In this case, as formulated in Section III-C, the MDP is no longer stationary, as the transition probability is time-dependent. We evaluate the video quality with respect to the percentage of video duration that T2 ON/OFF is known. The longer we know the exact T2 dynamics, the less uncertainty we have about future channel conditions, so the actions taken in our algorithm are closer to the optimal (Fig. 10b). In Fig. 11b, when level of perfect information is 1, our algorithm provides the same video quality as the optimal video quality, which means that if our algorithm knows the same information as the offline optimal algorithm, our algorithm can provides the optimal solution.

This video quality improvement will incur two types of cost: (1) scheduling T2 ON/OFF ahead of time, and (2) reconstructing the utility table for the non-stationary MDP. If a long period of T2 ON/OFF can be known ahead of time, T2 users have to follow the schedule and have less flexibility in using a channel than before. If the non-stationary MDP is used, the transition probability table is time-dependent and thus is unique for each realization. Each time a video is streamed, a new utility table of size $T \times S$ needs to be constructed, which requests more time and computation power. In stationary MDP, the utility table needs to be constructed only once and used for all realizations.

In comparison with the previous section, the knowledge of T2 dynamics for a continuous period will help to improve the performance of our algorithm, but knowledge of T2 only for the next state does not help much, as shown in Fig. 11. The



(a) Video quality comparison (b) Quality stability comparison.

Fig. 12: tradeoff between video quality and stability. 'MDP with sum util' is the algorithm defined in section III-B. 'MDP with prod util' is the MDP algorithm with geometric mean utility function, defined in section III-F.

optimality gap decreases as a longer period of T2 dynamics is fed into our algorithm, and the variation decreases as more information of T2 is provided, which means our algorithm is more certain about future and can take actions specific to each realization.

F. *Video Stability Analysis*

In addition to video quality itself, its stability is also important to user experience. Frequent quality change with bursts of high quality segments will degrade QoE. Fig. 12 shows the tradeoff between video quality and quality stability. Fig. 12b evaluates the stable video quality length for different methods while streaming the same video. Our methods with modified geometric mean utility is on the far right, clearly outperforming the other three. However, the increase in video quality stability is at the cost of lowering video quality, as shown in Fig. 12a. The video quality of this method is slightly worse than our MDP with summation utility function, though this method provides more stable qualtiy. The tradeoff between video quality and quality stability depends on which factor a video streaming client cares more.

Compared with DASH, the video quality of our MDP methods outperform DASH. For stability, the original MDP (take summation as the utility function) performs similar to DASH, but the MDP with geometric mean utility provides better stability. This is because "DASH" and "DASH with T2 information" try to make video quality stable by smoothing the estimated current throughput using the moving average of the past throughput. However, the moving average is ineffective when throughput fluctuation is very frequent. In our method, we consider the geometric mean (Eqn (6)) of all segments in bins, which not only maximizes the quality, but also minimizes the variation among these segments.

V. Related Work

The majority of research on whitespace is about spectrum sensing and dynamic spectrum access[13]. This includes spectrum allocation and AP detection[14], as well as the architecture of dynamic spectrum sharing that enables multi-tier access[15]. Sen and Zhang designed a dual technology WhiteCell that uses both cellular operator's spectrum and whitespace[16]. However, none of these architectures or algorithms consider how a particular application can adapt to the whitespace characteristics. To the best of our knowledge, adaptive video streaming in the context of whitespace has not been investigated prior to this work.

Adaptive video streaming and scalable video coding has been studied since the 1990s. The algorithms and system architectures to perform such adaptation have been intensively studied in research communities [2][17][18], as well as in industries, such as Apple, Netflix and Microsoft, who have developed HTTP video streaming protocols to perform channel-based video rate adaptation. Recently, there has also been increasing interest in DASH using SVC, such as the benefits of using SVC in DASH in terms of the efficiency of network cache[19], and experimental analysis on DASH-SVC[20], [21]. However, most of the algorithms are heuristic and rate adaptation is based on past throughput and buffer level. Compare to them, our MDP-based adaptation makes online decisions based on expected future channel conditions and provides the optimal actions in each state. Xiang[22] investigated MDP-based adaptive streaming using SVC, but only layers of future segments can be requested in this work, which does not fully utilize the flexibility of SVC to improve downloaded segments' quality. In that paper, the authors abstract the bandwidth variation model and uses a simple rayleigh fading channel. In our work, the bandwidth variation is based on T2 and T3 dynamics in whitespace. Our MDP-based adaptive algorithm is specific to whitespace dynamics and can request high layers of downloaded segments to improve video quality.

VI. Conclusion and Future Work

In this paper, we design an adaptive video streaming system specific to whitespace fast-varying channel. For the video rate adaptation, we use an MDP-based adaptation algorithm to provide the overall best video quality by considering the uncertainty in future throughput. For video coding, we use scalable video coding to provide more flexibility in adapting to the fast-varying channel. Our algorithm outperforms DASH both in terms of video quality and the temporal variation of video quality. When partial information of T2 dynamics is known ahead of time, our non-stationary MDP method is more certain about the segments to request and can improve the video quality. We also analyze the effect of T2 dynamics on T3 video streaming, and show that the frequency of T2 arrivals has more impact than the density of T2 users. In the future, we plan to generalize our MDP-based algorithm to consider the case of spectrum outage and also fit multi-tier models.

Acknowledgments

This work was in part supported by NSF NeTS grants CNS-1011962 and CNS-1318607 and a gift from Qualcomm.

References

[1] "Report to the president realizing the full potential of government-held spectrum to spur economic growth," Dec. 2012.

[2] T. Stockhammer, "Dynamic adaptive streaming over http–: standards and design principles," in *Proceedings of the second annual ACM conference on Multimedia systems*. ACM, 2011, pp. 133–144.

[3] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the h. 264/avc standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 17, no. 9, pp. 1103–1120, 2007.

[4] F. C. Commission *et al.*, "Amendment of the commissions rules with regard to commercial operations in the 3550-3650 mhz band," *docket No. 12-354, Further Notice of Proposed Rulemaking*, Apr. 2014.

[5] V. Abhayawardhana, I. Wassell, D. Crosby, M. Sellars, and M. Brown, "Comparison of empirical propagation path loss models for fixed wireless access systems," in *Vehicular Technology Conference, 2005. VTC 2005-Spring. 2005 IEEE 61st*, vol. 1. IEEE, 2005, pp. 73–77.

[6] C. Gardiner, *Stochastic methods*. Springer-Verlag, Berlin–Heidelberg–New York–Tokyo, 1985.

[7] X. Wang, J. Chen, A. Dutta, and M. Chiang, "Adaptive Video Streaming over Whitespace: SVC for 3-Tiered Spectrum Sharing," Princeton University, Tech. Rep., 11 2014. [Online]. Available: http://scholar.princeton.edu/sites/default/files/xw4/files/ws-video.pdf

[8] T. Andelin, V. Chetty, D. Harbaugh, S. Warnick, and D. Zappala, "Quality selection for dynamic adaptive streaming over http with scalable video coding," in *Proceedings of the 3rd Multimedia Systems Conference*. ACM, 2012, pp. 149–154.

[9] Azimuth channel emulator. [Online]. Available: http://www.azimuthsystems.com/products/ace-channel-emulators

[10] C. Mueller, S. Lederer, J. Poecher, and C. Timmerer, "Demo paper: Libdash-an open source software library for the mpeg-dash standard," in *Multimedia and Expo Workshops (ICMEW)*. IEEE, 2013, pp. 1–2.

[11] S. Lederer, C. Müller, and C. Timmerer, "Dynamic adaptive streaming over http dataset," in *Proceedings of the 3rd Multimedia Systems Conference*. ACM, 2012, pp. 89–94.

[12] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *Image Processing, IEEE Transactions on*, vol. 13, no. 4, pp. 600–612, 2004.

[13] S. Deb, V. Srinivasan, and R. Maheshwari, "Dynamic spectrum access in dtv whitespaces: design rules, architecture and algorithms," in *Proceedings of the 15th annual international conference on Mobile computing and networking*. ACM, 2009, pp. 1–12.

[14] X. Feng, J. Zhang, and Q. Zhang, "Database-assisted multi-ap network on tv white spaces: Architecture, spectrum allocation and ap discovery," in *New Frontiers in Dynamic Spectrum Access Networks (DySPAN), 2011 IEEE Symposium on*. IEEE, 2011, pp. 265–276.

[15] G. Gur, S. Bayhan, and F. Alagoz, "Cognitive femtocell networks: an overlay architecture for localized dynamic spectrum access [dynamic spectrum management]," *Wireless Communications, IEEE*, vol. 17, no. 4, pp. 62–70, 2010.

[16] S. Sen, T. Zhang, M. M. Buddhikot, S. Banerjee, D. Samardzija, and S. Walker, "A dual technology femto cell architecture for robust communication using whitespaces," in *Dynamic Spectrum Access Networks (DYSPAN), 2012 IEEE International Symposium on*. IEEE, 2012, pp. 242–253.

[17] S. Akhshabi, A. C. Begen, and C. Dovrolis, "An experimental evaluation of rate-adaptation algorithms in adaptive streaming over http," in *Proceedings of the second annual ACM conference on Multimedia systems*. ACM, 2011, pp. 157–168.

[18] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran, "Probe and adapt: Rate adaptation for http video streaming at scale," *Selected Areas in Communications, IEEE Journal on*, vol. 32, no. 4, pp. 719–733, 2014.

[19] Y. Sánchez de la Fuente, T. Schierl, C. Hellge, T. Wiegand, D. Hong, D. De Vleeschauwer, W. Van Leekwijck, and Y. Le Louédec, "idash: improved dynamic adaptive streaming over http using scalable video coding," in *Proceedings of the second annual ACM conference on Multimedia systems*. ACM, 2011, pp. 257–264.

[20] C. Muller, D. Renzi, S. Lederer, S. Battista, and C. Timmerer, "Using scalable video coding for dynamic adaptive streaming over http in mobile environments," in *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*. IEEE, 2012, pp. 2208–2212.

[21] C. Sieber, T. Hosfeld, T. Zinner, P. Tran-Gia, and C. Timmerer, "Implementation and user-centric comparison of a novel adaptation logic for dash with svc," in *Integrated Network Management (IM 2013)*. IEEE, 2013, pp. 1318–1323.

[22] S. Xiang, L. Cai, and J. Pan, "Adaptive scalable video streaming in wireless networks," in *Proceedings of the 3rd multimedia systems conference*. ACM, 2012, pp. 167–172.