Analysis of QoE for Adaptive Video Streaming over Wireless Networks

Sudheer Poojary*, Rachid El-Azouzi*, Eitan Altman[‡,*], Albert Sunny[‡], Imen Triki*, Majed Haddad*, Tania Jimenez*, Stefan Valentin[†] and Dimitrios Tsilimantos[†]
* CERI/LIA, University of Avignon, Avignon, France
† Paris Research Center, Huawei Technologies France
‡ Univ. Cote d'Azur, INRIA BP93, 06902 Sophia-Antipolis Cedex, France

***Abstract*—Adaptive video streaming improves users' quality of experience (QoE), while using the network efficiently. In the last few years, adaptive video streaming has seen widespread adoption and has attracted significant research effort. We study a dynamic system of random arrivals and departures for different classes of users using the adaptive streaming industry standard DASH (Dynamic Adaptive Streaming over HTTP). Using a Markov chain based analysis, we compute the user QoE metrics: probability of starvation, prefetching delay, average video quality and switching rate. We validate our model by simulations, which show a very close match. Our study of the playout buffer is based on client adaptation scheme, which makes efficient use of the network while improving users' QoE. We prove that for buffer-based variants, the average video bit-rate matches the average channel rate. Hence, we would see quality switches whenever the average channel rate does not match the available video bit rates. We give a sufficient condition for setting the playout buffer threshold to ensure that quality switches only between adjacent quality levels.**

I. Introduction

With rapid adoption of smart phones, video streaming consumption is increasing over cellular networks. YouTube statistics [1] suggest that at least one third of Internet users use YouTube and more than half of YouTube views come from mobile users. The users' perceived QoE (quality of experience) drives the revenue for content providers such as YouTube, Netflix. Hence analysis of QoE for streaming users, especially in dynamic wireless environments, is of prime importance.

Streaming clients typically play the video while it is being downloaded. The incoming video is stored in a playout buffer from which the user's player consumes the content. If the video quality can not be sustained, i.e., if the video bit-rate exceeds the available channel bandwidth, there is a risk of the playout buffer becoming empty during playout causing a video stall, also called a starvation. Ideally the client would want to play the video at the highest quality without starvation. In such scenarios, adaptive streaming techniques which match the user's video bitrate to the available capacity are highly desired.

We consider MPEG-DASH [2], an HTTP based adaptive streaming standard. In this standard, a video is split into short intervals called segments or chunks. Each of these segments are encoded at different bitrates with higher bitrates corresponding to higher quality. The segments' information viz., the timing, the resolution, video bit rate, URL are shared with the DASH clients in the form of a media presentation description (MPD) file. A DASH client could use its current playout buffer, past throughput measurements, the device features along with the MPD file to choose the quality for the next segment. The objective of the DASH client is to maximize its QoE. While DASH standardizes formats for the MPD file and expects the video to be saved in chunks with copies encoded at different bitrates, it does not specify the client's adaptation behaviour. The client could choose to use a rate-based or a playout buffer-based approach. In the rate-based approach the download rate of the previous segment is used to select the next segment quality. On the other hand, the playout buffer-based approach looks at the current playout buffer occupancy to decide the next segment's quality. In [3], [4], the authors show that to avoid starvation, rate-based approaches need to be conservative, i.e., use average video bit-rates lower than the available bandwidth. However buffer-based approaches do not suffer from this and are able to match the video bit-rate to the available bandwidth while avoiding starvation. Similar conclusions have been established in [5] for rate-based versus queue-based flow control algorithms.

While the quality of experience is a subjective quantity, there are metrics which help us quantify a user's QoE. The probability of starvation (stall/rebuffering) and the average video quality are obvious QoE metrics. Video clients typically prefetch some content before playout and also wait before playout after a starvation event. This causes a delay before playout which affects user experience. The average initial startup delay and rebuffering delay (delay post a starvation event) quantify this aspect of user QoE. With adaptive streaming, the user's video quality could change over time. Frequent quality switches are found to be detrimental to the user's QoE. Thus, the frequency of quality switching is another QoE metric. In this paper, we develop analytical approximations for the following QoE metrics: average startup delay, probability of starvation, average video quality and rate of video quality switches. While the overall QoE is subjective and is dependent on the context (e.g., type of content, user preference etc.), the above metrics are still primary indicators of QoE and are major contributing factors to the overall QoE.

We analyze a dynamic system with randomly arriving streaming users which share the network resources. Each user streams a video of random duration and exits the system on completion of stream. We model multiple classes of users, where a class could represent different user parameters, (e.g., channel conditions, arrival rate) or different DASH parameters.

A. Related Work

The reference [6] does a field study for monitoring QoE parameters for YouTube videos and finds that streaming parameters like stalling times, average time spent on a quality are more representative of subjective user QoE than network flow parameters. In [4], [7], [8], the authors describe various client side adaptation algorithms. These identify the following key QoE metrics, viz., startup delay, average video quality, stalling and quality switches. The adaptation algorithm in [7] uses video bit-rate and delay between segments as knobs to optimize the user QoE. In [4], the authors show that client-side adaptation could rely on the playout buffer measurements alone and reduce rebuffering events while sustaining high video quality. The authors in [8] point towards the lack of a principled approach towards adaptation and develop a control theoretic approach towards adaptation. In [9], the authors experimentally evaluate various adaptation algorithms based on their perceptual impact. The references [10], [11] provide a comprehensive survey of adaptive streaming techniques over HTTP.

While client-side adaptation is of prime importance, [12] points out the importance of network support for fairness, stability and efficiency when multiple adaptive flows compete for bandwidth over a shared link. The authors develop an in-network scheduling framework to achieve a balance between the conflicting requirements of efficient resource usage, fairness and user QoE. In [13], the authors present a theoretical model for computation of QoE metrics such as probability of starvation, prefetching and rebuffering delays. With a cost function which accounts for startup delay and starvation, they identify the optimal prefetch threshold. The reference [14] accounts for the flow dynamics, i.e., user arrivals and departures for computing the QoE metrics. The flow dynamics are described by a Markov process and the QoE metrics are computed using ODE and PDE models for the flow dynamics. However, these models are more suitable to study non-adaptive video streaming traffic. In [15], the authors formulate the QoE maximization as an optimization problem and provide optimal rate allocation for video streaming in a wireless network considering user dynamics.

B. Motivation and Our Contributions

Current cellular base stations incorporate sophisticated radio resource management techniques for flow scheduling. However these mechanisms are specially designed to manage media traffic with non-adaptive video streaming. In fact, for adaptive streaming such DASH protocol, LTE scheduling policy is in conflict with fairness since most adaptive streaming videos are greedy in the sense that if a user observes high throughput, higher and higher rates are requested until the maximum quality is reached. This can cause significant wastage of network bandwidth, particularly in dynamic scenarios. Additionally, accounting for user dynamics is important to design better algorithms and admission control policies which efficiently allocate resources across adaptive video streaming users.

The main contributions of our paper are as follows:

- We develop a general model for studying adaptive streaming with user dynamics.
- We model the system and user flow dynamics as Markov chains and derive the probability of starvation, the average startup delay, the average video quality and switching frequency.
- We show that the average video bitrate matches the average channel rate with buffer-based adaptive scheme in realistic scenarios with discrete video qualities.
- We observe that with buffer-based schemes, the video quality switches despite fixed channel rate. In such a scenario, we show that with appropriate spacing of the buffer level thresholds, we can prevent the video quality from jumping too many quality levels.

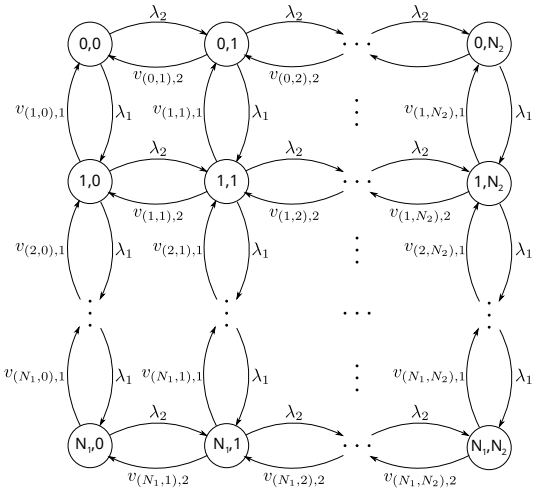

Fig. 1. Transition diagram for a two class CTMC. We have $Q(\vec{i}, \vec{i}+e_j) = \lambda_j$ and $Q(\vec{i}, \vec{i}-e_j)$ is denoted by $v_{\vec{i},j}$. For a class 1 user, states with $i_1 = N_1$ are blocking, and for class 2 user, states with $i_2 = N_2$ are blocking states.

II. Models for Dynamic Scenario

A. Markov model for multi-class dynamic traffic

We consider a dynamic network with variable number of streaming mobile users with finite-size service demands. A new mobile user joins the network and requests streaming service from a media server. The user exits the network upon completion of its stream. Let λ_k be the arrival rate of new class k user. We assume that the arrival processes are independent Poisson processes. We assume that the video duration of a class k user, measured in seconds, is exponentially distributed with mean $1/\theta_k$. The vector of mobile users, $\vec{i}(t) = (i_1(t), i_2(t), \ldots, i_K(t))$, where $i_k(t)$ is the number of class k users, defines the state of the system. Given the assumption of exponential distributed video durations, it can be shown that the service times of the mobile users are also exponentially distributed. This implies that the remaining service of a user given the state at time t is independent of the past. We restrict the number of users of each class to be finite and denote by $\mathcal{X} = \{(i_1, i_2, \ldots, i_K) : 0 \leq i_k \leq N_k\}$, the state space of our system. Under the above assumptions, the dynamics of coexisting mobile users in the cell is a continuous time Markov chain (CTMC) with finite state space $\mathcal{X}$. Figure 1 illustrates the transition diagram for a system with two classes.

The timescale of scheduling is of the order of 2 milliseconds whereas the timescale of a video segment playout is in seconds and the timescale of arrivals and departures is of the order of tens of seconds. Due on this separation of timescales, the following assumptions are reasonable. In the time

between two state changes in the CTMC, $\vec{i}(t)$, we assume that the average channel rate and the average video bitrate of a video streaming flow are functions of the current state. Users of the same class get the same average throughput. Given this, as their client adaptation algorithm is identical, their average video bitrate is also same. Let us denote the average channel rate (in bps) of a class k user by $r_k(\vec{i})$ and its average video bit rate (in bps) by $\ell_k(\vec{i})$, when the system state is $\vec{i}$. The average channel rate would depend on the scheduling policy and the channel statistics of the user. We assume that given a scheduling policy the average channel rate is a function of $\vec{i}$. For example, with weighted proportional fair scheduling on a shared channel of capacity C, the average channel rates for different users is given by the solution of

$$\max \sum_{k \in K} w_k i_k \log(r_k(\vec{i})), \tag{1}$$

subject to $\sum_{k \in K} i_k r_k(\vec{i}) \leq C$.

Let Q denote the rate transition matrix of $\vec{i}(t)$ process. Let $\{e_1, e_2, \ldots, e_K\}$ be the standard basis vectors of $\mathbb{R}^K$, i.e., e_j is a unit vector with 1 in the j^{th} position. Then $Q(\vec{i}, \vec{i} - \vec{e_k}) = \frac{i_k r_k(\vec{i}) \theta_k}{\ell_k(\vec{i})}$ is the rate at which a class k user leaves the system. We note that $Q(\vec{i}, \vec{i} + \vec{e_k}) = \lambda_k$. The stationary distribution π of $\vec{i}(t)$ satisfies $\pi Q = 0$ and $\pi \mathbf{1} = 1$ with $\mathbf{1}$ being a vector with all entries 1. For user QoE analysis, besides π, we need to understand the dynamics of the system as seen by the users of each class. We do this by analyzing the system dynamics as seen by 'tagged' users of different classes.

B. Flow dynamics observed by a tagged class j user

When a tagged class j user joins the network, in the steady state, it finds $(i_1, i_2, \ldots, i_K)$ other mobile users with probability $\pi_{\vec{i}}$ with $\vec{i} = (i_1, i_2, \ldots, i_K)$. If it finds the system in state $\vec{i}$ such that $i_j = N_j$, it is dropped. For a class j user accepted into the system, the system dynamics as experienced by it can be modelled through a finite Markov Chain Y_j with state space $\mathcal{X}_j = \mathcal{X} - \{\vec{i} : i_j = N_j\}$. Additionally, we have an absorbing state, A, corresponding to the tagged user finishing service. Let Q_j denote the rate transition matrix for Y_j. The departures and arrivals of users define the transitions of the process Y_j. The transition rate $Q_j(\vec{i}, \vec{i} - e_k)$ from state $\vec{i}$ to $\vec{i} - e_k$ is given by $\frac{i_k r_k(\vec{i}+e_j)\theta_k}{\ell_k(\vec{i}+e_j)}$ and $Q_j(\vec{i}, \vec{i}+e_k) = \lambda_k$. After the tagged class j user joins the network, the Markov process Y_j provides the system dynamics as observed by the tagged user. Specifically, the Markov process, $Y_j(t)$ is the number of other users in the system that a particular tagged user of class j sees in its sojourn in the system. We note that we use $r_j(\vec{i}+e_j)$ for average channel rate and $\ell_j(\vec{i}+e_j)$ for average video bitrate when $Y_j = \vec{i}$. We get an additional e_j corresponding to the tagged class j user observing the system. This is to ensure consistency with the system Markov chain $\vec{i}(t)$ description.

III. Computation of QoE Metrics

We now develop approximations for the QoE metrics: start-up delay, the starvation behaviours, the average quality and the quality variation. We give a summary of the notations that we use in Table I.

TABLE I. Tabular summary of notation used in the paper.

Symbol	Meaning
	System Markov chain
$\vec{i}(t)$	Markov process denoting state of system
Q	Transition rate matrix for $\vec{i}(t)$ process
π	Stationary distribution of $\vec{i}(t)$ process
$\mathcal{X}$	State space of $\vec{i}(t)$ process
	Tagged user Markov chain
$Y_j(t)$	State of system as seen by a class j user
Q_j	Transition rate matrix for $Y_j(t)$ process
M_j	T.P.M. for embedded Markov chain of $Y_j(t)$
$\mathcal{X}_j$	State space of $Y_j(t)$ process
$\mathcal{B}_j$	States in $\mathcal{X}_j$ with channel capacity $< \ell_{min}$
N_j	Maximum number of class j users in system
	QoE metrics
q_a	Prefetching threshold in seconds
$D_j(q_a, \vec{i})$	Mean startup delay for class j with initial state $\vec{i}$
$D_j(q_a)$	Unconditional mean startup delay for class j user
$P_j(\vec{i})$	Starvation probability for class j user with initial state $\vec{i}$
P_j	Unconditional starvation probability for class j user
$V_j(\vec{i})$	Mean video bitrate for class j user with initial state $\vec{i}$
V_j	Unconditional mean video bitrate for class j user
$S_j(\vec{i})$	Mean quality switches for class j user with initial state $\vec{i}$
S_j	Unconditional mean quality switches for class j user
R_j	Unconditional rate of switching for class j user

A. Start-Up Delay

The start-up delay is the duration between the time that a user initiates a session and the time that the media player starts playing video frames. In the initial prefetching phase, the player does not start playback until the duration of received video reaches the start-up threshold. Let $q(t)$ denote the remaining seconds of video in the playout buffer at time t. Assuming r is the average channel rate and ℓ is the average video bitrate in $[t, t+h]$, the player downloads video at a rate of r/ℓ video seconds per second. Assuming that the user plays out video at normal speed with no starvation event in $[t, t+h]$, the playout buffer evolves as

$$q(t+h) = \begin{cases} q(t) + rh/\ell, & \text{during prefetching,} \\ \max\{0, q(t) - h + rh/\ell\}, & \text{otherwise.} \end{cases} \tag{2}$$

Let q_a be the start-up threshold in seconds. Then the start-up delay is given by $D(q_a) = \inf\{t \geq 0 \mid q(t) \geq q_a\}$. Let $D_j(q_a)$ be the average startup delay for a user of class j. Let $D_j(q_a, \vec{i})$ denote the average startup delay for a user entering the system in state $\vec{i}$. The arrivals and departures happen at a time scale of tens of seconds and the prefetching typically finishes sooner. Thus, we can safely assume that the probability of an arrival or departure event while a user is prefetching its video is negligible. During prefetching, the DASH client is conservative and starts with the lowest video quality. We assume that prefetching finishes using the lowest video quality. Then, using Equation (2), the average startup delay $D_j(q_a, \vec{i})$ is given by

$$D_j(q_a, \vec{i}) = q_a \ell_{min} / r_j(\vec{i} + e_j),$$

where ℓ_{min} is the video bitrate for the lowest quality video. The average start up delay is computed by averaging over all possible initial states, i.e., $D_j(q_a) = \sum_{\vec{i} \in \mathcal{X}_j} \pi(\vec{i}) D_j(q_a, \vec{i}) / \sum_{\vec{i} \in \mathcal{X}_j} \pi(\vec{i})$, where π is the stationary distribution of CTMC, $\vec{i}(t)$. The normalization term $\sum_{\vec{i} \in \mathcal{X}_j} \pi(\vec{i})$ is needed as we only consider the users accepted into the system for computing the QoE metrics.

B. Starvation Probabilities

1) An upper bound: We assume that the video of a class j user experiences starvation if, during playout, its corresponding Markov chain Y_j visits state $\vec{i}$ such that $r_j(\vec{i}+e_j) < \ell_{min}$, where ℓ_{min} is the lowest video bitrate available. This is a good approximation in the regime where the video durations and the inter-arrival times are large.

Suppose that a tagged user of class j enters the system at some state $\vec{i}$. Let us denote by $P_j(\vec{i})$ the probability of starvation for this tagged user. Let $\mathcal{B}_j = \{\vec{i} \in \mathcal{X}_j : r_j(\vec{i}+e_j) < \ell_{min}\}$. Let M_j denote the transition probability matrix for the embedded Markov chain of $Y_j(t)$. Then we can write the following recursive equation for $\{P_j(\cdot)\}$: $P_j(\vec{i}) = \sum_{\vec{i'}} P_j(\vec{i'})M_j(\vec{i},\vec{i'})$ for all $\vec{i} \in \mathcal{X}_j \setminus \mathcal{B}_j$, else $P_j(\vec{i}) = 1$. This gives us one non-trivial linear equation for each $\vec{i} \in \mathcal{X}_j \setminus \mathcal{B}_j$. Thus, we get a system of $|\mathcal{X}_j \setminus \mathcal{B}_j|$ linear equations in $|\mathcal{X}_j \setminus \mathcal{B}_j|$ unknowns, which can be solved using matrix inversion. Then the unconditional starvation probability P_j for a class j user is $\sum_{\vec{i}\in\mathcal{X}_j} \pi(\vec{i})P_j(\vec{i}) / \sum_{\vec{i}\in\mathcal{X}_j} \pi(\vec{i})$.

In the preceding computation, we assumed that if the tagged class j user visits a state in $\mathcal{B}_j$, it stays long enough in that state so that its playback buffer depletes to 0 before it exits the state. This ignores the possibility that there could be a transition from a state in $\mathcal{B}_j$ to $\mathcal{X}_j \setminus \mathcal{B}_j$ before the buffer depletes to 0, thus avoiding starvation. Thus P_j, as computed above, gives us an upper bound on the probability of starvation. Next we describe an approximation which takes into account this possibility.

2) Accounting for effect of transitions: We consider two possibilities : (a) avoiding starvation due to prefetching and (b) tagged user avoiding starvation by exiting $\mathcal{B}_j$ before the buffer depletes to 0. With this, the probability of starvation for a class j user entering the system in state $\vec{i}$ is given by

$$P_j(\vec{i}) = p_j(\vec{i}) + (1 - p(\vec{i})) \sum_{\vec{i'}} M_j(\vec{i},\vec{i'})\hat{P}_j(\vec{i},\vec{i'}), \qquad (3)$$

where $p_j(\vec{i})$ is the probability that the users buffer depletes to 0 before transition out of state $\vec{i}$. The quantity $p_j(\vec{i})$ accounts for the impact of prefetching. The term $\hat{P}_j(\vec{i},\vec{i'})$ is the probability that the users video is starved after it transits from state $\vec{i}$ to $\vec{i'}$. If $\vec{i} \in \mathcal{X}_j \setminus \mathcal{B}_j$, $p(\vec{i}) = 0$, if $\vec{i'} \in \mathcal{X}_j \setminus \mathcal{B}_j$, $\hat{P}_j(\vec{i},\vec{i'}) = P_j(\vec{i'})$. If $\vec{i},\vec{i'} \in \mathcal{B}_j$, $\hat{P}_j(\vec{i},\vec{i'}) = 1$. If $\vec{i} \in \mathcal{X}_j \setminus \mathcal{B}_j$ and $\vec{i'} \in \mathcal{B}_j$, we have

$$\hat{P}_j(\vec{i},\vec{i'}) = p_j(\vec{i},\vec{i'}) + (1 - p_j(\vec{i},\vec{i'})) \sum_{\vec{i'}} M_j(\vec{i},\vec{i'})\hat{P}_j(\vec{i},\vec{i'}), \quad (4)$$

where $p_j(\vec{i},\vec{i'})$ is the probability that the user's buffer depletes to 0 before transition out of state $\vec{i'}$ given that just prior to $\vec{i'}$ the user visited state $\vec{i}$. The term $p_j(\vec{i},\vec{i'})$ takes into account the possibility of avoiding starvation by user transiting out of $\vec{i'} \in \mathcal{B}_j$ before its buffer depletes to 0. It is a function of $\vec{i}$ and $\vec{i'}$ since the initial buffer value just prior to entering $\vec{i'}$ is a function of $\vec{i}$ and the rate of buffer depletion then on is a function of $\vec{i'}$.

Computation of $p_j(\vec{i},\vec{i'})$ and $p_j(\vec{i})$: For computing $p_j(\vec{i})$ for $\vec{i} \in \hat{\mathcal{B}}_j$, we need to find the time for the buffer to deplete to 0 for a user entering system in state $\vec{i}$. Let us denote this time by $T_j(\vec{i})$. The term $T_j(\vec{i})$ includes (a) the time to prefetch q_a seconds of video and (b) the time to deplete the prefetched video. Assuming the prefetching video bitrate is ℓ_{min} and using Equation (2), we get

$$T_j(\vec{i}) = q_a/(r_j(\vec{i}+e_j)/\ell_{min}) + q_a/(1-(r_j(\vec{i}+e_j)/\ell_{min}). \quad (5)$$

The sojourn time in state $\vec{i}$ is exponentially distributed with parameter $|Q_j(\vec{i},\vec{i})|$. Hence the quantity $p_j(\vec{i})$ is given by $\exp(-|Q_j(\vec{i},\vec{i})|T_j(\vec{i}))$.

We now compute $p_j(\vec{i},\vec{i'})$ with $\vec{i} \in \mathcal{X}_j \setminus \mathcal{B}_j$ and $\vec{i'} \in \mathcal{B}_j$. Let $T_j(\vec{i},\vec{i'})$ be the time required for the buffer to deplete to 0 when the Markov chain transitions from $\vec{i}$ to $\vec{i'}$. Let us assume that just before transition, the buffer size is $b_j(\vec{i})$ sec. Then, as $\ell_j(\vec{i'} + e_j) = \ell_{min}$, using Equation (2), we get

$$T_j(\vec{i},\vec{i'}) = b_j(\vec{i})/(1 - r_j(\vec{i'} + e_j)/\ell_{min}) \qquad (6)$$

and $p_j(\vec{i},\vec{i'})$ is then given by $\exp(-|Q_j(\vec{i'},\vec{i'})|T_j(\vec{i},\vec{i'}))$. We describe computation of $b_j(\vec{i})$ for the buffer-based DASH variant in Section IV.

C. The average video quality

We now compute the average video quality using the tagged Markov chain approach. Let $\tau_j(\vec{i},\vec{i'})$ denote the proportion of time that a class j user, entering in state $\vec{i}$, spends in state $\vec{i'}$ before finishing playout. We can compute it as follows

$$\tau_j(\vec{i},\vec{i'}) = \frac{\sum_{n=0}^{\infty} M_j(\vec{i},\vec{i'})^n/|Q_j(\vec{i'},\vec{i'})|}{\sum_{\vec{k}\in\mathcal{X}_j}\sum_{n=0}^{\infty} M_j(\vec{i},\vec{k})^n/|Q_j(\vec{k},\vec{k})|}.$$

For computing the infinite series, we use the matrix identity, $(I - M)^{-1} = \sum_{n=0}^{\infty} M^n$. Then, the average video quality, $V_j(\vec{i})$ for a class j user entering the system in state $\vec{i}$ is $\sum_{\vec{i'}} \ell_j(\vec{i'})\tau_j(\vec{i},\vec{i'})$. The unconditional average video quality V_j for a class j user is $\sum_{\vec{i}\in\mathcal{X}_j} \pi(\vec{i})V_j(\vec{i}) / \sum_{\vec{i}\in\mathcal{X}_j} \pi(\vec{i})$.

D. The average rate of quality switches

Buffer-based DASH clients match the average video bitrate to the average channel rate. Thus with buffer-based DASH, in states where the channel rate does not match video bit rate, we have non-zero frequency of quality switching. Hence corresponding to each state, $\vec{i}$, there is an associated rate of switching. We also have quality switches when the system state changes. Let $s_j(\vec{i})$ denote the average number of switches that a class j user experiences during one sojourn in state $\vec{i}$. Let $z_j(\vec{i},\vec{i'})$ denote the number of switches that a class j user experiences when Y_j transitions from $\vec{i}$ to $\vec{i'}$.

Let $S_j(\vec{i})$ denote the average number of switches that a class j user experiences during the playout of its video when it enters the system in state $\vec{i}$. This can be computed as follows

$$S_j(\vec{i}) = s_j(\vec{i}) + \sum_{\vec{i'}} M_j(\vec{i},\vec{i'})\Big(z_j(\vec{i},\vec{i'}) + S_j(\vec{i'})\Big).$$

The above gives us $|\mathcal{X}_j|$ linear equations in $|\mathcal{X}_j|$ variables and we can compute $\{S_j(\vec{i}) : \vec{i} \in \mathcal{X}_j\}$ using matrix inversion. The average number of switches S_j is then given by $\sum_{\vec{i}\in\mathcal{X}_j} \pi(\vec{i})S_j(\vec{i}) / \sum_{\vec{i}\in\mathcal{X}_j} \pi(\vec{i})$. The average rate R_j of switching for a video of average duration $1/\theta$ is given by S_j/θ.

IV. DASH Performance Evaluation

For the computation of the QoE metrics described in Section III, we need to compute the following quantities:

- the average channel rate $r_j(\vec{i}+e_j)$,
- the average video bitrate $\ell_j(\vec{i}+e_j)$,
- the average buffer size $b_j(\vec{i})$,
- the average number of switches $s_j(\vec{i})$,
- the average number of switches $z_j(\vec{i},\vec{i'})$.

The channel rates $\{r_j(\vec{i}+e_j)\}$ achieved by the different classes of users depends on the base station's scheduling scheme. The other quantities depend on the DASH protocol used for adaptive streaming. For example, a throughput based DASH client could set the average video rate $\ell_j(\vec{i}+e_j)$ to be the largest supportable video bitrate smaller than the average channel rate $r_j(\vec{i}+e_j)$. In the following discussion, we will consider a buffer-based DASH client downloading video segments at a fixed rate r. This corresponds to analyzing our dynamic system when it is in state $\vec{i}$ and considering a class j user with average channel rate $r_j(\vec{i}+e_j)=r$.

A. Avoiding multiple video bitrate switches

Consider a user downloading video on a channel with average channel rate r and assume that it uses video bit rates from the set $\mathcal{L}=\{\ell_1,\ell_2,\ldots,\ell_m : \ell_{min}=\ell_1<\ell_2\ldots<\ell_m=\ell_{max}\}$. We consider the steady state QoE experienced by the user and assume that it is using a buffer-based DASH. With DASH, the video is segmented into segments (also called chunks) and at the end of every chunk download, the DASH client decides on the next video quality, which corresponds to choosing a video bit rate from $\mathcal{L}$ for the next segment. Depending on the current buffer level, the client chooses the next quality. Let us denote the buffer size (in video segments) at time t by $b(t)$. With discrete video qualities, after the client finishes downloading a video segment, it chooses the next quality as follows:

- If $b(t)\le b_1$, next segment is requested at quality ℓ_{min},
- if $b_{m-1}<b(t)$, next segment is requested at quality ℓ_{max},
- if $b_{k-1}<b(t)\le b_k$, next segment is requested at quality ℓ_k.

Figure 2 illustrates the buffer values to video bitrates mapping.

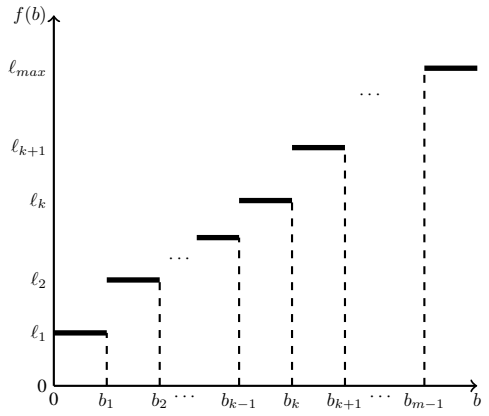

Fig. 2. A mapping from buffer occupancy to a discrete set of video bitrates

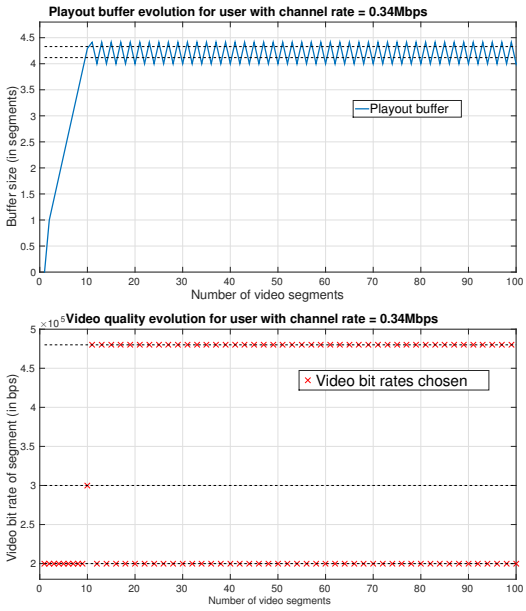

Fig. 3. The DASH client has a fixed channel rate of 0.34 Mbps and has video representations encoded at bitrates 0.2, 0.3 and 0.48 Mbps. We expect that in steady state, the video quality should switch between 0.3 and 0.48 Mbps. In the top figure we see that the buffer size (in steady state) at the end of every segment download crosses two threshold values (marked with dotted lines). This causes the video quality to switch between 0.2 and 0.48 Mbps, whereas an intermediate representation, 0.3 Mbps is never used in steady state.

If $r\notin\mathcal{L}$ and $\ell_{min}<r<\ell_{max}$, with buffer-based DASH, for long videos, video quality switches are inevitable. Suppose $\ell_k<r<\ell_{k+1}$ then one would expect the video quality to switch between ℓ_k and ℓ_{k+1}. However if the buffer thresholds are not spaced appropriately, it is possible that the client chooses other qualities in steady state. For example, we see in Figure 3 that a user with channel rate 0.34 Mbps switches video qualities between 0.2 Mbps and 0.48 Mbps skipping the intermediate representation with bit rate 0.3 Mbps. The following result gives a sufficient condition to avoid such unnecessary switches in steady state.

Proposition 1 *For buffer-based DASH with discrete video qualities $\mathcal{L}=\{\ell_1,\ell_2,\ldots,\ell_m : \ell_{min}=\ell_1<\ell_2\ldots<\ell_m=\ell_{max}\}$, if the buffer thresholds $\{b_1,b_2,\ldots,b_{m-1}\}$ are chosen such that*

$$b_k-b_{k-1}>(\ell_{k+1}/\ell_k)-1 \quad \textit{and} \tag{7}$$
$$b_{k+1}-b_k>1-(\ell_k/\ell_{k+1}), \tag{8}$$

with $b_0=0$ then for $r\in(\ell_k,\ell_{k+1})$, the system reaches a steady state eventually, where the video quality switches between ℓ_k and ℓ_{k+1}. If $r\in\mathcal{L}$, then in steady state there are no quality switches and steady state video bit rate is r.

Proof: Suppose $r\in(\ell_k,\ell_{k+1})$. Let $b[m]$ denote the buffer size (in video segments) and let $\ell[m]$ denote the video quality chosen at the beginning of m^{th} segment download. We need to show that there exists a N such that for $n>N$, $b[n]\in(b_{k-1},b_{k+1}]$. As $r>\ell_{min}$, there is no starvation and the evolution of $\{b[m]\}$ is given by

$$b[m+1]=b[m]+1-\ell[m]/r. \tag{9}$$

Suppose $b[m] \in (b_{k-1}, b_k]$, then $\ell[m] = \ell_k$. As $r < \ell_{k+1}$, using Equations (8) and (9), we get $b_k < b[m+1] \leq b_{k+1}$. Similarly, if $b[m] \in (b_k, b_{k+1}]$, using Equation (7), $b_k < b[m+1] \leq b_{k+1}$. Thus if $b[n] \in (b_{k-1}, b_{k+1}]$, for all $m \geq 0$, $b[n+m] \in (b_{k-1}, b_{k+1}]$.

Now let us consider $b[m] \in [0, b_{k-1}]$. This gives us $\ell[m] < \ell_k$ and from (9), we get $b[m+1] > b[m]$. Let $b[m] \in (b_{j-1}, b_j]$ with $b_j \leq b_{k-1}$. Then $\ell[m] = \ell_j < \ell_k$. From Equation (9),

$$b[m+1] - b[m] = 1 - \ell_j/r. \tag{10}$$

From Equation (8), we get a series of inequalities $b_{i+1} - b_i > 1 - (\ell_i/\ell_{i+1})$, for $i = \{j, j+1, \ldots, k\}$. Adding these inequalities, we get $b_{k+1} - b_j > (j-k) - \sum_{i=j}^{i=k} (\ell_i/\ell_{i+1})$. As $\ell_j/r < \ell_j/\ell_{j+1}$ and $\ell_i/\ell_{i+1} < 1$, $(j-k) - \sum_{i=j}^{i=k} (\ell_i/\ell_{i+1}) > 1 - \ell_j/r$. This shows that $b[m+1]$ can not exceed b_{k+1}. Thus for $b[m] \in [0, b_{k-1}]$, $b[m] < b[m+1] < b_{k+1}$, which ensures that for some $N > m$, $b[n] \in (b_{k-1}, b_{k+1}]$, for all $n > N$.

Next we consider $b[m] \in (b_{k+1}, \infty)$. As $\ell[m] > \ell_{k+1}$, using Equation (9), we get $b[m+1] > b[m]$. From our earlier arguments, if $b[m] \in [0, b_{k+1}]$, there exists N such that $b[n] \in (b_{k-1}, b_{k+1}]$ for all $n > N$. Using this and the fact that $b[m+1] < b[m]$, as earlier for some $N > m$, $b[n] \in (b_{k-1}, b_{k+1}]$, for all $n > N$. Thus, we have shown that irrespective of the initial value of the buffer, there exists some N such that $b[n] \in (b_{k-1}, b_{k+1}]$, for all $n > N$, which proves the first claim of the Proposition.

When $r \in \mathcal{L}$, using similar steps as above, we can show that eventually the buffer sizes converge to the region corresponding to $f(b) = r$. From then on as video bitrate matches r, there is no change in buffer size and hence no further change in video bitrate. ∎

B. Average video bitrate for buffer-based DASH

In [4], the authors prove that when the video qualities are continuous and the map, $f(b)$ from buffer levels to video quality is increasing, the DASH clients choose video qualities such that the average video bitrate matches the average channel rate. We now prove the result when video bitrates are discrete.

Proposition 2 *Consider a buffer-based DASH algorithm where the map $f(b)$ from buffer levels to video quality is increasing. Also assume that the buffer thresholds are chosen as described in Proposition 1. Then if $\ell_{min} < r < \ell_{max}$, the average video bitrate is r.*

Proof: If the buffer spacing criterion of Proposition 1 is satisfied. Then for r such that $\ell_k < r < \ell_{k+1}$, the sequence $\{b[m]\}$ of playout buffer sizes at the beginning of m^{th} video segment eventually ends up in $(b_k, b_{k+1}]$, i.e., for some N, $b[n] \in (b_k, b_{k+1}]$ for all $n > N$. For computing the asymptotic average bitrate, the initial N video segments can be ignored. Hence for computing the asymptotic average bitrate, we assume that $b[0] \in (b_{k-1}, b_{k+1}]$, so that we download videos at bit rate ℓ_k or ℓ_{k+1}. To compute the average video bit rate, we need to compute the (steady state) fraction of videos downloaded at bitrates ℓ_k and ℓ_{k+1}. If $b[m] \in (b_{k-1}, b_k]$, $b[m+1] = b[m] + 1 - \ell_k/r$ and if $b[m] \in (b_k, b_{k+1}]$, $b[m+1] = b[m] + 1 - \ell_{k+1}/r$. At the beginning of the N^{th} segment download, we have $b[N] = b[0] + n_1(1 - \ell_k/r) + n_2(1 - \ell_{k+1}/r)$, where $N = n_1 + n_2$ and n_1, n_2 are the number of segments downloaded at bit rates ℓ_k and ℓ_{k+1} respectively. Let $\alpha \triangleq \lim_{N\to\infty} n_1/N$ denote the fraction of segments downloaded at bit rate ℓ_k. Since $|b[N] - b[0]| < b_{k+1} - b_{k-1}$ as $N \to \infty$, $(b[N] - b[0])/N \to 0$. Hence we get $\alpha(1 - \ell_k/r) + (1-\alpha)(1 - \ell_{k+1}/r) = 0$ which gives us $\alpha\ell_k + (1-\alpha)\ell_{k+1} = r$). ∎

C. Average rate of video quality switches

Proposition 1 shows that we can avoid switching between non-adjacent quality levels provided the buffer thresholds are chosen appropriately. In the following proposition, we present an approximation for the frequency of switching when $\ell_k < r < \ell_{k+1}$. We will assume that the buffer size oscillates between a and c in the steady state, with $a < c$.

Proposition 3 *Suppose that the average channel rate, r is such that $\ell_k < r < \ell_{k+1}$ and the buffer thresholds are chosen as stated in Proposition 1, then the frequency, $f(r)$ of switching in steady state is given by*

$$f(r) = \begin{cases} \frac{2}{v(\ell_{k+1}/r + (\ell_k/r)(\ell_{k+1}-r)/(r-\ell_k)))}, & \text{if } r < \frac{\ell_k+\ell_{k+1}}{2} \\ \frac{2}{v(\ell_k/r + (\ell_{k+1}/r)(\ell_k-r)/(r-\ell_{k+1}))}, & \text{otherwise} \end{cases}$$

switches per video second, with v being the number of seconds in one video segment.

Proof: We prove the result for $r < \frac{\ell_k+\ell_{k+1}}{2}$, the other result can be shown in similar fashion. Let T_1, T_2 denote the time for the buffer size to go from a to c and from c to a respectively. When $r < \frac{\ell_k+\ell_{k+1}}{2}$, the DASH client downloads one segment at rate ℓ_{k+1} and the buffer goes from c to a. It then switches to ℓ_k. As r is closer to ℓ_k than ℓ_{k+1}, it takes more than one segment download to go from a to c. Using (2), the time T_2 to download one segment at rate ℓ_{k+1} is given by $v\ell_{k+1}/r = (c-a)\ell_{k+1}/(\ell_k - r)$ seconds. The time T_1 is given by,

$$T_1 = (c-a)\ell_k/(r-\ell_k) = v(\ell_{k+1} - r)\ell_k/(r(r-\ell_k)). \tag{11}$$

As there are two quality switches in time $T_1 + T_2$, we have

$$f(r) = 2/(T_1 + T_2) = \frac{2}{v(\ell_{k+1}/r + (\ell_k/r)(\ell_{k+1} - r)/(r - \ell_k)))}.$$

∎

Computation of $\ell_j(\vec{i} + e_j)$, $b_j(\vec{i})$, $s_j(\vec{i})$ and $z_j(\vec{i}, \vec{i'})$: We now give approximations for $\ell_j(\vec{i}+e_j)$, $b_j(\vec{i})$, $s_j(\vec{i})$ and $z_j(\vec{i}, \vec{i'})$ using Propositions 1, 2 and 3, From Proposition 2, we have $\ell_j(\vec{i}+e_j) = r_j(\vec{i}+e_j)$, whenever $\ell_{min} < r_j(\vec{i}+e_j) < \ell_{max}$. If $r_j(\vec{i}+e_j) < \ell_{min}$, $\ell_j(\vec{i}+e_j) = \ell_{min}$ and if $r_j(\vec{i}+e_j) > \ell_{max}$, $\ell_j(\vec{i}+e_j) = \ell_{max}$. If $\ell_k < r_j(\vec{i}+e_j) < \ell_{k+1}$, the buffer is close to b_k. Hence, in this case, we set $b_j(\vec{i}) = b_k$. From Proposition 3, $s_j(\vec{i}) = f(r_j(\vec{i}+e_j))/|Q_j(\vec{i}, \vec{i})|$. For computation of $z_j(\vec{i}, \vec{i'})$, we count the number of video qualities in $\mathcal{L}$ which are between $r_j(\vec{i} + e_j)$ and $r_j(\vec{i'} + e_j)$. Using these, we can compute the QoE metrics outlined in Section III.

V. Simulation Results

In this section, we validate our analytical results against simulations. Some simulation results are omitted here due to the lack of space. These can be found in our technical report [16].

We simulate about 10^5 users entering the system, whose arrival into the network is a Poisson process of rate λ. Each arriving user requests a video with exponential distribution

and leaves the system once the video is downloaded. The system enforces admission control rejecting new users when the system is full. The average channel rate depends on the number of streaming users of the different classes. The average video bit-rate of each user is dictated by the DASH buffer-based bit-rate adaptation algorithm. The segments are downloaded sequentially and the decision on the quality of the next segment is based on the buffer level at the end of current segment download. The buffer thresholds (as described in Figure 2) are set as follows, $b_1 = 4$ segments, $b_{m-1} = 10$ segments. The other values are uniformly spaced which, for our simulation parameters, ensures that the buffer spacing condition of Proposition 1 is satisfied.

A. *Homogeneous case*

Here, we consider a single class of users, The users receive equal share of network capacity. Each video segment has 2 seconds of video. The playout buffer prefetch threshold, q_a is set to 1 segment. The available video bitrates are $\mathcal{L} = \{0.2, 0.3, 0.48, 0.75, 1.2, 1.85, 2.85, 4.3, 5.3\}$ Mbps. The users arrive at a rate of 0.01 arrivals per sec. The average video duration is 20 minutes while the total channel capacity available to the streaming users is set to 4 Mbps.

We first investigate the impact of admission control on the QoE metrics. In Figures 4, 5, 6, 8 we show the effect of changing the maximum number of users on the probability of starvation, startup delay, average quality and the probability of dropping. In Figure 7, we compare the rate of switching as predicted by the model with simulations. As the maximum number of users allowed to contend for the channel increases, the probability of starvation and average startup delay increases whereas the average video bit rate and the probability of user blocking decreases. We see that the probability of starvation is close to 0 when the maximum number of users N is less than 10. This occurs as the average rate per user when $N \leq 10$ is greater than $\ell_{min} = 0.2$ Mbps. This indicates that the buffer-based rate adaptation strategy does avoid starvation whenever the available capacity is at least ℓ_{min}. Also our model quantifies the trade-off between user blocking and user QoE, demonstrating the use of admission control for achieving the desired QoE.

We note that in Propositions 1, 2 and 3, we assumed that the video is downloaded at a fixed rate of r bps and that the user's client adaptation algorithm is operating in its steady state. In the simulations, the channel rate and the client adaptation changes due to user arrival/departure. Our simulation results are close to the analytical values which suggests that the DASH dynamics converge fairly quickly, thus validating our model and approximations.

B. *Multiple Classes*

In this section, we consider multiple class of users and compare the analytically obtained QoE metrics for each class of users with simulations. We consider two classes of users, where a user of class 1 gets twice the rate as compared to a user of class 2. The disparity is representative of a scenario where class 1 users are closer to the base station whereas class 2 users are cell-edge users. Also, this setup corresponds to a weighted proportional fair scheduler (Equation (1) in Section II) with two classes with weights 2 and 1. The total capacity available to all the users is 5 Mbps. The arrival rate for each class user is 0.01, the average video duration is 10 minutes. The maximum number of users for each class is chosen from $\{5, 10\}$ as stated in the first column of Tables II, III and IV. The DASH parameters for all users are set as in the homogeneous case.

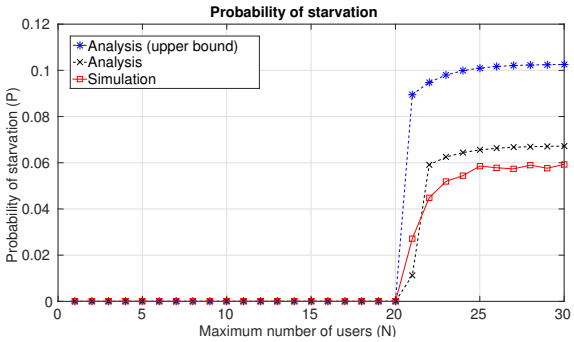

Fig. 4. Effect of admission control on probability of starvation.

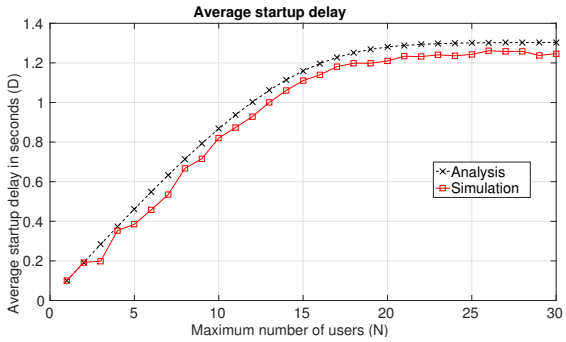

Fig. 5. Effect of admission control on startup delay.

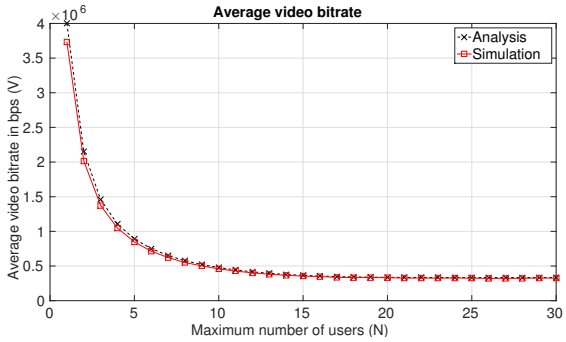

Fig. 6. Effect of admission control on average video bit rate.

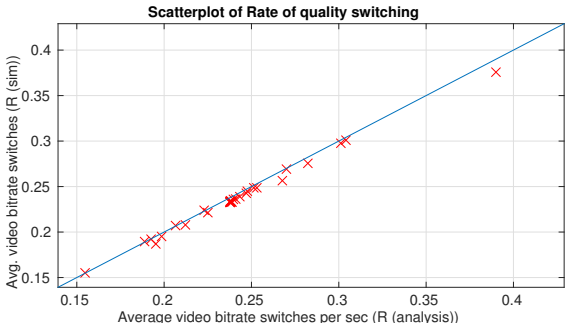

Fig. 7. Comparison of average rate of switching (number of switches/sec) with simulation results.

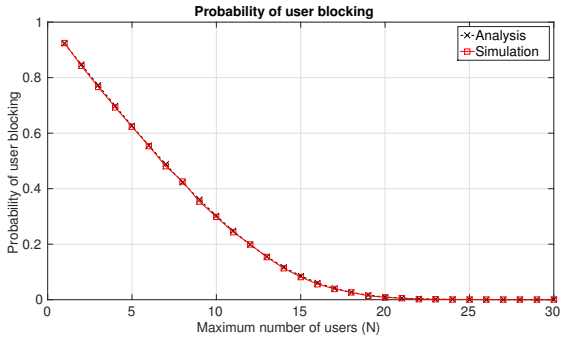

Fig. 8. Probability of user blocking.

TABLE II. PROBABILITY OF STARVATION

N_1, N_2	P_1, P_2 (sim.)	P_1, P_2 (analysis) Exact	P_1, P_2 (analysis) Upper Bound
$5, 5$	$0, 0$	$0, 0$	$0, 0$
$5, 10$	$0, 0$	$0, 0$	$0, 0$
$10, 5$	$0, 0$	$0, 0$	$0, 0$
$10, 10$	$0, 0.11$	$0, 0.17$	$0, 0.27$

The different QoE metrics for different values of N_1, N_2 are tabulated in Tables II, III and IV. We see that there is a good match between simulations and our analytical results. As in the single class case, there is a trade-off between user blocking and the QoE metrics, starvation, startup delay and average video bit rate. For example, if we increase the maximum number of either class from 5 to 10, the probability of user dropping of the corresponding class decreases from ≈ 0.36 to ≈ 0.04 (see Table IV). However, when we increase both N_1 and N_2 from 5 to 10, the users of class 2 experience non-zero probability of starvation (see Table II).

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we have developed a model based on Markov chains for QoE analysis of adaptive streaming clients over a dynamic wireless environment. Specifically, we derived approximations for probability of starvation, average startup delay, average video quality and switching frequency for buffer-based DASH clients. We have shown that, in case of buffer-based adaptive schemes with discrete video qualities, the average video bitrate matches the average channel rate. We observed that with buffer-based schemes, the video quality switches despite a constant channel rate. In such a scenario, we have shown that with appropriate spacing of the buffer level thresholds, we can prevent the video quality from jumping too many quality levels.

In our future works, we plan to extend our model to account for background traffic and user abandonments.

TABLE III. AVG. STARTUP DELAY (IN SEC) AND AVG. VIDEO BITRATE (IN MBPS)

N_1, N_2	D_1, D_2 sim.	D_1, D_2 analysis	V_1, V_2 sim.	V_1, V_2 analysis
$5, 5$	$0.46, 0.92$	$0.48, 0.95$	$0.79, 0.42$	$0.85, 0.45$
$5, 10$	$0.53, 1.11$	$0.56, 1.14$	$0.69, 0.35$	$0.75, 0.37$
$10, 5$	$0.65, 1.22$	$0.68, 1.26$	$0.59, 0.33$	$0.64, 0.35$
$10, 10$	$0.72, 1.41$	$0.75, 1.44$	$0.53, 0.29$	$0.57, 0.30$

TABLE IV. AVERAGE RATE OF SWITCHING (IN SWITCHES PER SEC) AND PROBABILITY OF USER BLOCKING

N_1, N_2	R_1, R_2 sim.	R_1, R_2 analysis	$\pi(N_1, \cdot), \pi(\cdot, N_2)$ sim.	$\pi(N_1, \cdot), \pi(\cdot, N_2)$ analysis
$5, 5$	$0.21, 0.27$	$0.21, 0.27$	$0.35, 0.35$	$0.36, 0.36$
$5, 10$	$0.24, 0.23$	$0.25, 0.23$	$0.35, 0.04$	$0.36, 0.04$
$10, 5$	$0.23, 0.23$	$0.23, 0.24$	$0.04, 0.36$	$0.04, 0.36$
$10, 10$	$0.24, 0.21$	$0.24, 0.22$	$0.04, 0.04$	$0.04, 0.05$

REFERENCES

[1] "Youtube Statistics." https://www.youtube.com/yt/about/press/. Accessed: 2017-12-26.

[2] I. Sodagar, "The MPEG-DASH Standard for Multimedia Streaming Over the Internet," *IEEE MultiMedia*, vol. 18, pp. 62–67, April 2011.

[3] T.-Y. Huang, R. Johari, and N. McKeown, "Downton abbey without the hiccups: Buffer-based rate adaptation for HTTP video streaming," in *SIGCOMM FhMN workshop*, pp. 9–14, ACM, 2013.

[4] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," *ACM SIGCOMM CCR*, vol. 44, no. 4, pp. 187–198, 2015.

[5] E. Altman, F. Baccelli, and J. C. Bolot, "Discrete-time analysis of adaptive rate control mechanisms," in *High Speed Networks and their performance*, pp. 121–140, H. G. Perros and Y. Viniotis, North Holland, 1994.

[6] M. Seufert, P. Casas, F. Wamser, N. Wehner, R. Schatz, and P. Tran-Gia, "Application-layer monitoring of QoE parameters for mobile YouTube video streaming in the field," in *IEEE Sixth International Conference on Communications and Electronics (ICCE)*, pp. 411–416, July 2016.

[7] K. Miller, E. Quacchio, G. Gennari, and A. Wolisz, "Adaptation algorithm for adaptive streaming over HTTP," in *Packet Video Workshop (PV), 2012 19th International*, pp. 173–178, IEEE, 2012.

[8] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over HTTP," *ACM SIGCOMM CCR*, vol. 45, no. 4, pp. 325–338, 2015.

[9] T. C. Thang, H. T. Le, A. T. Pham, and Y. M. Ro, "An evaluation of bitrate adaptation methods for http live streaming," *IEEE J. Sel. Areas Commun.*, vol. 32, pp. 693–705, April 2014.

[10] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hofeld, and P. Tran-Gia, "A Survey on Quality of Experience of HTTP Adaptive Streaming," *IEEE Commun. Surveys Tuts*, vol. 17, no. 1, pp. 469–492, 2015.

[11] J. Kua, G. Armitage, and P. Branch, "A Survey of Rate Adaptation Techniques for Dynamic Adaptive Streaming Over HTTP," *IEEE Commun. Surveys Tuts*, vol. 19, no. 3, pp. 1842–1866, 2017.

[12] J. Chen, R. Mahindra, M. A. Khojastepour, S. Rangarajan, and M. Chiang, "A Scheduling Framework for Adaptive Video Delivery over Cellular Networks," in *MOBICOM*, pp. 389–400, ACM, 2013.

[13] Y. Xu, E. Altman, R. El-Azouzi, M. Haddad, S. Elayoubi, and T. Jimenez, "Analysis of Buffer Starvation With Application to Objective QoE Optimization of Streaming Services," *IEEE Trans. Multimedia*, vol. 16, pp. 813–827, April 2014.

[14] Y. Xu, S. E. Elayoubi, E. Altman, and R. El-Azouzi, "Impact of flow-level dynamics on QoE of video streaming in wireless networks," in *INFOCOM*, pp. 2715–2723, IEEE, 2013.

[15] V. Joseph, S. Borst, and M. I. Reiman, "Optimal Rate Allocation for Video Streaming in Wireless Networks With User Dynamics," *IEEE/ACM Trans. Netw.*, vol. 24, pp. 820–835, April 2016.

[16] S. Poojary, R. El-Azouzi, E. Altman, A. Sunny, I. Triki, M. Haddad., T. Jimenez, D. Tsilimantos, and S. Valentin, "Analysis of QoE for Adaptive Video Streaming over Wireless Networks," in *Technical Report, http://lia.univ-avignon.fr/chercheurs/elazouzi/Technical-report-Wiopt18.pdf.*