QoE-Aware Dynamic Video Rate Adaptation

Yanjiao Chen[§], Fan Zhang[†], Fan Zhang[§], Kaishun Wu[‡], Qian Zhang[§]

[§]Department of Computer Science & Engineering, Hong Kong University of Science and Technology
[†]Department of Electronic Computer & Engineering, Hong Kong University of Science and Technology
‡College of Computer Science and Software Engineering, Shenzhen University and HKUST Fok Ying Tung Research Institute

***Abstract*—Dynamic video streaming protocols allow video clients to adaptively choose video rates to improve viewer Quality-of-Experience (QoE). Existing works on rate adaptation choose the video rate according to network capacity or buffer state, however, they didn't consider the early quitting problem. Most viewers quit before the video ends, in which case optimal solutions for complete video sessions may become suboptimal. In this paper, we design a video rate adaptation scheme which considers incomplete video sessions and aims at maximizing the quality of the video chunks that are actually watched by viewers. We formulate the video rate adaptation problem as a Markov Decision Process (MDP), in which the viewers' decision of quitting depends on the video quality and the rebuffer time. To solve the MDP problem, we propose two online learning algorithms for regular viewers and new viewers with no viewing history, respectively. The simulation results show that the proposed scheme increases the video quality truly consumed by the viewers, thus improving the viewer QoE.**

I. Introduction

With the ever-increasing popularity of video streaming over the Internet, especially over the mobile network, the video service providers make every endeavor to improve viewers' Quality-of-Experience (QoE). Throughout the world, video traffic is expected to be 69% of all consumer Internet traffic by 2017 [1], and mobile video traffic will be over one third of mobile data traffic by the end of 2018 [2].

Video streaming protocols, such as HTTP Live Streaming (HLS) protocol and Dynamic Adaptive Streaming over HTTP (MPEG-DASH), allow video clients to adaptively select video rates. Each video file is divided into video chunks (usually of the same length, say $2 \sim 10$ seconds), and each video chunk is available at various rates from 235kb/s standard definition to 5Mb/s high definition [3]. The major target of the video rate adaptation method is to maximize viewer QoE, which is affected mainly by two factors: video quality (quantified by video rates) and rebuffer time. Existing video rate adaptation methods aim at achieving a balance between the video quality and the rebuffer time by monitoring the network conditions or/and buffer state. The video clients will switch to a low rate if the network capacity is estimated to drop [4]–[6] or the buffer is expected to deplete [3], [7], [8].

Existing works often implicitly make the assumption that viewers will watch the entire video, however, many measurement studies have shown that most viewers will abandon the video sessions halfway [9]–[11]. In this case, as shown in Fig. 1, simply maximizing video quality or minimizing rebuffer time will drive viewers away. Therefore, rate adaptation schemes for complete video sessions can not be directly applied to incomplete video sessions since the latter have different objective function and constraints.

In this paper, we propose a video rate adaptation scheme, which aims at maximizing the rate of video chunks that are actually consumed by the viewer. On the one hand, we retain viewers as long as possible by lowering rebuffer time; on the other hand, we track the network capacity and buffer state to select the proper video rate. We formulate the video rate adaptation as a Markov Decision Process (MDP), in which the probability of viewer quitting depends on the video rate and the rebuffer time. We consider two scenarios: a regular viewer whose viewing habit is known, and a new viewer whose viewing habit is unknown. We propose two online learning algorithms to solve the MDP for these two scenarios.

We make the following contributions:

- *QoE-aware dynamic video rate adaptation scheme* which considers the early quitting problem and maximizes the video quality that is actually consumed by the viewer.
- *MDP problem formulation* which involves network condition, buffer state and viewer dynamics.
- *Optimal rate adaptation algorithms* based on online learning for both regular viewers and new viewers.

The rest of the paper is organized as follows. In Section II, we briefly introduce existing video rate adaptation approaches. In Section III, we describe the system model and design rationale of the proposed rate adaptation scheme. In Section IV, we formulate the rate adaptation as an MDP problem and propose two algorithms to solve the MDP problems in cases of complete and incomplete knowledge of viewers' viewing habits. We evaluate the performance of the proposed rate adaptation scheme in Section V, and finally summarize our work in Section VI.

II. Background of Video Rate Adaptation

In this session, we give a brief introduction of existing video rate adaptation approaches, describing their advantages and disadvantages.

Video rate adaptation aims at selecting the "right" rate for each video chunk to maximize viewer QoE. Two major criteria that are considered in video rate adaptation are video quality (quantified by video rate) and rebuffer time. Fig. 2 shows a typical video session. The viewer initiates a video request to

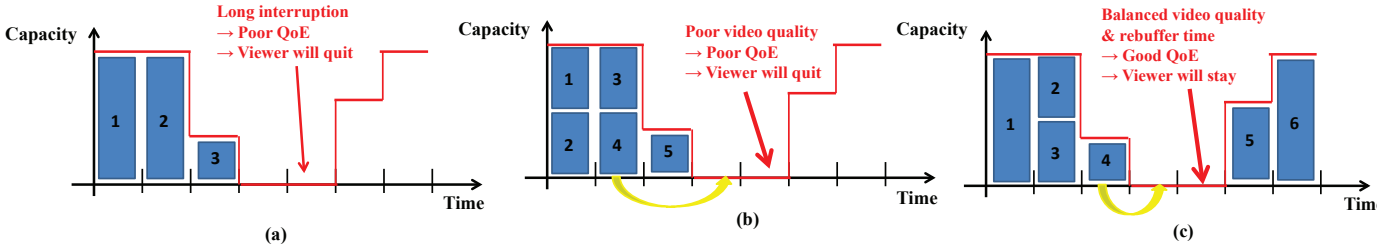

Fig. 1: Different video rate adaptation strategies: (a) maximizing video rate; (b) minimizing rebuffer time; (c) proposed video rate adaptation.

establish connection with the server. Then, a certain number of video chunks are downloaded in the buffer before the video starts playing. During playing, the video player fetches the video chunks in the buffer to display to the viewer; and meanwhile, downloads more video chunks (of different rates) from the server. If the speed of displaying the video chunks exceeds that of downloading the video chunks (e.g., due to poor network capacity), the buffer will be exhausted. In this case, the video player has to freeze to fill its buffer to a certain level before starts playing again. The duration of this interruption is referred to as rebuffer time. Viewers may watch the whole video (complete video session) or abandon the video halfway (incomplete video session).

Fig. 2: A typical video watching session.

Ideally, the best viewing experience calls for highest video quality and minimum rebuffer time. It is easy to achieve either one of the goals, but the two goals are conflicting with each other. If we only care about the video quality, we can choose the highest possible rate (supported by the network capacity) for all chunks. In this case, it is very likely that the viewers will experience long rebuffer time. If we only want to minimize the rebuffer time, we can choose the lowest possible rate for all chunks. Given that this rate is less than the network capacity at all times, the rebuffer time will be zero. Nevertheless, the video quality will be very poor. Existing video rate adaptation approaches aim at a tradeoff of both video quality and rebuffer time. There are three mainstream video rate adaptation methods:

Network capacity based method. Intuitively, if the network capacity is high, we are able to choose high quality video chunks; if the network capacity is low, we are forced to choose low quality video chunks. Therefore, the performance of the network capacity based method largely depends on the accuracy of capacity estimation. Network capacity based method usually formulates the video rate adaptation as an optimization problem. The objective function is the targeted video quality or rebuffer time, and the constraint is the estimated network capacity. The one-off solution plans the video rate adaptation before the video session begins, which does not adjust to the real network condition during the video session.

Buffer state based method. Since network capacity may vary dramatically within a short period of time, especially in the mobile network, accurate capacity estimation is hard to achieve. In contrast, the buffer state is relatively stable and easy to monitor. The basic idea of buffer based method is to leverage the buffer state as much as possible to guide rate adaptation: use more aggressive rate adaptation policy (choose high quality video chunks) if the buffer is nearly full, and more conservative rate adaptation policy (choose low quality video chunks) if the buffer is nearly empty [3]. Compared with network capacity based method, the buffer state based method dynamically schedule the video quality. But one problem with the buffer state based method is that, not all the video chunks in the buffer will be consumed by the viewer if he/she quits during the video session. Therefore, it is not enough to only focus on the video chunks in the buffer.

Hybrid method. Hybrid method takes advantage of both network capacity based method and buffer state based method to reach a better solution. One way of hybrid method is to adapt video rate according to network capacity estimation while use buffer level as a feedback [7], [12]. Another way of hybrid method is to use different methods at different phases: when the buffer starts to build from empty, network capacity based method can be leveraged, since the buffer may not be a good indicator for network condition; when the buffer has been built up, we may rely solely on buffer state based method [3]. The proposed QoE-aware dynamic video rate adaptation belongs to hybrid method.

III. System Model

As shown in Fig. 3, we consider a single video stream, which has a total length of N seconds. The entire video file is divided into T chunks of the same time duration. Therefore, each video chunk has a length of N/T seconds. Each video chunk is available at different rates, thus having different sizes and different quality. The video chunks with higher rate have larger size and better quality, vice versa.

We assume that the rate adaptation is made every N/T seconds, that is, every time the viewer consumes exactly one

video chunk. We consider an infinite time horizon of $t = 1, 2, ...$; each time stage lasts for N/T seconds [1]. At each time stage, the scheduler chooses the rate for the video chunks to be transmitted. Let $\mathcal{E}$ denote the finite set of available rates, and $e_i \in \mathcal{E}$ denote a certain rate. We have $0 \in \mathcal{E}$ as the scheduler may choose not to require video chunks. We assume that the video chunks are transmitted and received sequentially in the right order[2]. This is reasonable because even if a later video chunk is transmitted before an early video chunk, the media player cannot play the later video chunk and has to wait for the early video chunk.

The transmitted video chunks will be put in a buffer with size L. The buffer is a first-in-first-out (FIFO) queue. If the buffer is full, but the video transmission is ongoing, the extra received video chunks will be discarded. At each time stage, if the buffer is not empty, the media player will take one video chunk from the buffer, and play it to the viewer; if the buffer is empty, the viewer will experience rebuffer interruption.

At time stage t, the current system state is $S(t)$, and the action is to choose the rate $E(t)$. This leads to utility $U(S(t), E(t))$ for the viewer, and system state transition to $S(t+1)$. The goal is to maximize the utility over the entire time horizon, subject to the system state transition function. Before giving detailed description of the MDP problem formulation, we first outline our design rationale.

Modeling the system state.

The system state $S(t)$ has to track the viewer state (whether the viewer is active or not) and the elements that have an influence on viewers' utility. We assume that the viewers' utility is determined by the rate of the video chunk watched by the viewer, which is the first video chunk in the buffer. Therefore, the system state has to include the buffer state. The buffer state is affected by the video chunk transmission. Hence, the system state has to incorporate the network capacity, which determines how many video chunks can be transmitted.

Modeling the viewer state transition.

To maximize the video quality that is actually consumed by the viewer, we have to know when the viewer decides to quit the video session, that is, when the viewer state changes from active to inactive. At each time stage, the probability that the viewer state changes from active to inactive is referred to as the probability of quiting. In this paper, we assume that the probability of quiting is affected mainly by video quality and rebuffer time. Extensions to other influential factors can be modeled in a similar way.

Modeling the buffer state.

Although at each time stage, the viewer only watches the first video chunk in the buffer, we have to keep track of the rate of all video chunks in the buffer. If not, we don't know the rate of the video chunk to be played in the next time stage. Therefore, we use a vector to denote all the video chunks in the buffer. But the problem is that the number of video chunks in the buffer is not fixed, because different chunks have different sizes and the buffer can be partially full. To address this problem, we specify the vector length to be $M = L/\min_{e_i \in \mathcal{E}} e_i$, the maximum possible number of video chunks in the buffer. When the buffer is partially full, we assign 0 to the empty entries in the vector.

[1]Due to rebuffer time, the time horizon is longer than the video length.

[2]It is likely that some media players will require the same video chunks that are already in the buffer but with a higher rate when the network capacity is good. We will consider this situation in the future.

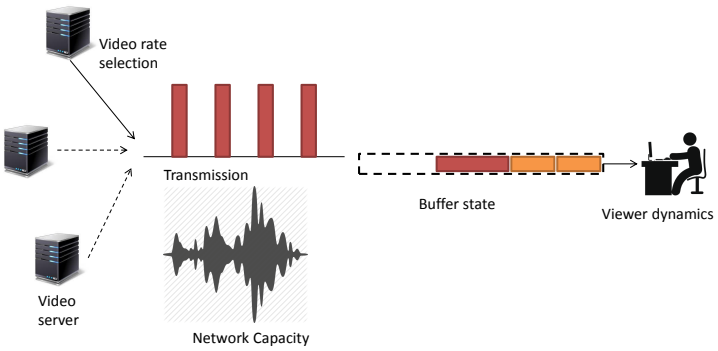

Fig. 3: System model of video scheduling scheme.

IV. QoE-aware Dynamic Video Rate Adaptation

In this section, we first formulate the video rate adaptation as an MDP problem. Then, we propose two algorithms to solve the MDP problem, which give the optimal rate adaptation schemes.

A. MDP Problem Formulation

A complete Markov Decision Process includes the system state, the action, the state transition function and the utility function.

1) System State $S(t)$: As we discussed in Section III, the system state should include network capacity, buffer state, and viewer state.

- *Network state $C(t)$.* $C(t)$ denotes the network capacity at time stage t. Here we consider the application-level network capacity[3], which has the same time granularity as the rate adaptation.
- *Buffer State $B(t)$.* $B(t)$ is denoted by $(b_1(t), b_2(t), ..., b_M(t))$, in which $b_i(t)$ is the rate of the ith video chunk. If there is fewer than ith video chunks in the buffer, $b_i(t) = 0$.
  Valid buffer state should satisfy two constraints: 1) FIFO queue constraint: $b_i = 0, b_j \neq 0, i < j$ should not happen; 2) buffer length constraints: $\sum_i b_i < L$. For example, if the buffer size is 40Mb, available rate set is $\{2, 3\}$Mb/s, and video chunk duration is 10 second; then there are

[3]Application-level network capacity is largely affected by physical-layer network capacity, which is fast-changing, especially in wireless networks. Here, the application-level network capacity can be taken as smoothed or averaged physical-layer network capacity. It is our future direction for cross-layer video scheduling scheme design.

four possible buffer states $(0,0),(2,0),(3,0),(2,2)$. The buffer state space is finite.

- *Viewer State* $V(t)$. If the viewer is active (watching the video or waiting during rebuffer), $V(t)=1$; if the viewer is inactive (quit), $V(t)=0$.

The system state is $S(t)=[C(t),B(t),V(t)]$. We discretize the value of $C(t)$ to make the state space finite. Infinite state space is future work. Let $\mathcal{S}$ denote the system state space, and $\{c_1,c_2,...,c_{n_c}\}$ denote the network capacity space.

2) Action $E(t)$: The action is to choose the rate of video chunks to be transmitted at time stage t. If the scheduler chooses to transmit nothing, $E(t)=0$. If $E(t)$ is chosen, the size of the video chunks is $E(t)N/T$. Therefore, the number of video chunks that can be successfully transmitted in time stage t is:

$$k=\frac{C(t)N/T}{E(t)N/T}=\frac{C(t)}{E(t)}. \tag{1}$$

3) State Transition Function $Pr[S(t+1)|S(t),E(t)]$: Given the current system state $S(t)$ and action $E(t)$, the probability of the next system state being $S(t+1)$ is $Pr[S(t+1)|S(t),E(t)]$. We make the assumption that the entries in the system state are independent[4]. Hence, we can decouple the state transition function as:

$$\begin{aligned}&Pr[S(t+1)|S(t),E(t)]=Pr[C(t+1)|S(t),E(t)]\\&\times Pr[B(t+1)|S(t),E(t)]\times Pr[V(t+1)|S(t),E(t)]\end{aligned} \tag{2}$$

Network state transition function. The network capacity is merely determined by network conditions. We assume that the network capacity is a random variable, following a certain distribution, which can be obtained from measurement results or empirical models. The network state transition function is:

$$Pr[C(t+1)|S(t),E(t)]=p_i^c, \text{if } C(t+1)=c_i.$$

Buffer state transition function. The buffer state is determined by video chunk transmission and consumption. At time stage t, a number of $k=C(t)/E(t)$ video chunks are transmitted. They will be put in the buffer if there is enough space; otherwise, extra video chunks will be discarded. If the viewer is active ($V(t)=1$) and the buffer is non-empty, the first video chunk will be consumed.

- Viewer active, buffer non-empty:
  $V(t)=1, B(t)=(b_1(t),...,b_i(t),0,...,0), 1\le i\le M$. If $i=M$, the buffer is full. The next buffer state must be:
  $$\widetilde{B}=(b_2(t),...,b_i(t),\underbrace{E(t),...,E(t)}_{\widetilde{k}},0,...,0),$$
  in which $\widetilde{k}\le k$, and $N[b_2(t)+...+b_i(t)+\widetilde{k}E(t)]/T\le L$.
- Viewer active, buffer empty:
  $V(t)=1, B(t)=(0,...,0)$. The next buffer state must be:
  $$\widetilde{B}=(\underbrace{E(t),...,E(t)}_{\widetilde{k}},0,...,0),$$
  in which $\widetilde{k}\le k$, and $N\widetilde{k}E(t)/T\le L$. Note that in this condition, the viewer will experience rebuffer interruption.
- Viewer inactive, buffer empty or non-empty:
  $V(t)=0, B(t)=(b_1(t),...,b_i(t),0,...,0), 0\le i\le M$. If $i=0$, the buffer is empty. The next buffer state must be:
  $$\widetilde{B}=(b_1(t),...,b_i(t),\underbrace{E(t),...,E(t)}_{\widetilde{k}},0,...,0),$$
  in which $\widetilde{k}\le k$, and $N[b_1(t)+...+b_i(t)+\widetilde{k}E(t)]/T\le L$.

The buffer state transition function is:

$$Pr[B(t+1)|S(t),E(t)]=\begin{cases}1, & \text{if } B(t+1)=\widetilde{B}\\ 0, & \text{otherwise}\end{cases} \tag{3}$$

Note that (3) shows that, given the current system state and action, the buffer state in the next time stage is definite.

Viewer state transition function. If the current viewer state is inactive, the next viewer state must be inactive. If the current viewer state is active, the next viewer state is determined by the watched video rate. Given that the viewer is active, and $b_1(t)=e_i\in\mathcal{E}$, the viewer state transition function is:

$$Pr[V(t+1)|S(t),E(t)]=\begin{cases}p_i^v, & \text{if } V(t+1)=0\\ 1-p_i^v, & \text{if } V(t+1)=1\end{cases} \tag{4}$$

If the viewer experiences rebuffer interruption, $b_1(t)=0$. p_i^v is the quitting probability for video rate e_i.

4) Utility Function: If the viewer is inactive, the utility is zero. If the viewer is active, the viewer utility is the rate of the video chunk currently watched by the viewer, which equals the rate of the first video chunk in the buffer. If the buffer is empty, i.e. $b_1(t)=0$, the viewer will experience rebuffer interruption. Therefore, the utility function is:

$$U(S(t),E(t))=\begin{cases}0, & \text{if } V(t)=0\\ b_1(t), & \text{if } V(t)=1\end{cases} \tag{5}$$

If the viewer quits the video session before t, his utility will always be zero for the rest of the time stages. Since the video length is limited, the utility function will converge to zero as the time horizon goes to infinity.

Our goal is to maximize the expected total viewer utility throughout the entire time horizon[5].

$$\max_{E(t),t=1,2,...}\sum_t U(S(t),E(t)) \tag{6}$$

[4]It is reasonable that the network capacity is independent from buffer state or viewer state. However, other elements may interact with each other. In the future, we will study how to account for the interdependency of different elements in the system state.

[5]We ignore the notation of expectation without confusing.

B. Optimal Video Rate Adaptation for Regular Viewers

We first consider a regular viewer, whose probability of quitting $p_i^v, \forall e_i \in \mathcal{E}$ is known. Let s denote the observed system state, and $\Omega(s)$ denote the rate selected by the video rate adaptation scheme. We have the following sufficient condition for optimal video rate adaptation.

Theorem 1: Optimal video rate adaptation for regular viewers. Video rate adaptation Ω is optimal if Ω achieves the minimum of the following equation for all $s \in \mathcal{S}$:

$$V(s) = \min_{\Omega} \Big\{ U(s, \Omega(s)) + \sum_{s'} Pr[s'|s, \Omega(s)] V(s') \Big\}, \forall s \in \mathcal{S} \tag{7}$$

in which $V(s)$ is referred to as value function of s.

Equation (7) is known as Bellman equation [13]. We propose Algorithm 1 to solve Equation (7) by dynamically learning the value function $V(\cdot)$. In the learning stage, the value function $V(\cdot)$ is updated in each iteration, based on the observed system state and corresponding utility. When the value function $V(\cdot)$ converges, we can get the optimal video rate adaptation Ω^*.

Algorithm 1 Optimal Video Rate Adaptation for Regular Viewers

1: Initialize value function $V^0(s), \forall s \in \mathcal{S}$
2: **for all** Time stage $t, t = 1, 2, ...$ **do**
3: Observe the system state $S(t)$ as s
4: **if** $\sum_s |V^t(s) - V^{t-1}(s)| > \delta$ **then**
5: **Learning stage**:
6: The optimal rate based on s is

$$e^* = \arg\min_{e \in \mathcal{E}} \Big\{ U(s, e) + \sum_{s'} Pr[s'|s, e] V^t(s') \Big\} \tag{8}$$

7: After chosen the optimal video rate e^*, update the value function of s as follows:

$$V^{t+1}(s) = V^t(s) + \epsilon_{n(s,t)} \Big\{ U(s, e^*) + V^t(S(t+1)) - V^t(s) \Big\} \tag{9}$$

in which $n(s, t)$ is the number of occurrence of state s during 0 till t; the learning step $\epsilon_n = \log n / n$; $S(t+1)$ is the real observed system state in $t+1$.
8: Other value functions remain unchanged.
9: **else**
10: **Steady stage**:
11: Choose rate by optimal video rate adaptation $\Omega^*(s)$ according to Equation (7).
12: **end if**
13: **end for**

C. Optimal Video Rate Adaptation for New Viewers

For a new viewer, his probability of quitting $p_i^v, \forall e_i \in \mathcal{E}$ is unknown. Therefore, Algorithm 1 cannot be applied because we don't know $Pr[s'|s, e]$ in Equation (8). In this case, we have a different sufficient condition for optimal video rate adaptation.

Theorem 2: Optimal video rate adaptation for new viewers. Video rate adaptation Ω is optimal if Ω achieves the minimum of the following equation for all $s \in \mathcal{S}$:

$$\Omega^*(s) = \min_{e \in \mathcal{E}} Q(s, e), \forall s \in \mathcal{S} \tag{10}$$

in which $Q(s, e)$ is referred to as Q-factor:

$$Q(s, e) = U(s, e) + \sum_{s'} Pr[s'|s, e] \min_{e' \in \mathcal{E}} (s', e') \tag{11}$$

Equation (11) is another form of Bellman equation [13]. We propose Algorithm 2 to solve Equation (11). In Algorithm 2, neither the derivation of optimal rate in (12) nor the update of Q-factor in (13) requires the knowledge of p_i^v. However, the Achilles heel of Algorithm 2 is that, there are far more Q-factors than value functions, because the Q-factor is a function of both s and e, while the value function is only the function of s. Therefore, it will take longer time for Algorithm 2 to converge.

Algorithm 2 Optimal Video Rate Adaptation for New Viewers

1: Initialize Q-factor $Q^0(s, e), \forall s \in \mathcal{S}, e \in \mathcal{E}$
2: **for all** Time stage $t, t = 1, 2, ...$ **do**
3: Observe the system state $S(t)$ as s
4: **if** $\sum_s \sum_e |Q^t(s, e) - Q^{t-1}(s, e)| > \delta$ **then**
5: **Learning stage**:
6: The optimal rate based on s is

$$e^* = \arg\min_{e \in \mathcal{E}} Q^t(s, e) \tag{12}$$

7: After chosen the optimal video rate e^*, update the Q-factor (s, e^*) as follows:

$$Q^{t+1}(s, e^*) = Q^t(s, e^*) + \epsilon_{n(s,e^*,t)} \Big\{ U(s, e^*) + \min_{e' \in \mathcal{E}} Q^t(S(t+1), e') - Q^t(s, e^*) \Big\} \tag{13}$$

in which $n(s, e^*, t)$ is the number of updates of (s, e^*) during 0 till t; the learning step $\epsilon_n = \log n / n$; $S(t+1)$ is the real observed system state in $t+1$.
8: Other value functions remain unchanged.
9: **else**
10: **Steady stage**:
11: Choose rate by optimal video rate adaptation $\Omega^*(s)$ according to Equation (10).
12: **end if**
13: **end for**

V. Performance Evaluation

We use the latest rate adaptation in [12] as the benchmark, which optimizes video rate adaptation for complete sessions. We choose the following three objective functions for the benchmark scheme: 1) maximize video quality, denoted as "Max rate"; 2) minimize rebuffer time, denoted as "Min rb"; 3) maximize (video quality - rebuffer time), denoted as "Max rate-rb". Default parameter values are shown in Table I [3].

Fig. 4(a) shows the accumulated video rate consumed by the viewer during a video session. The proposed rate adaptation

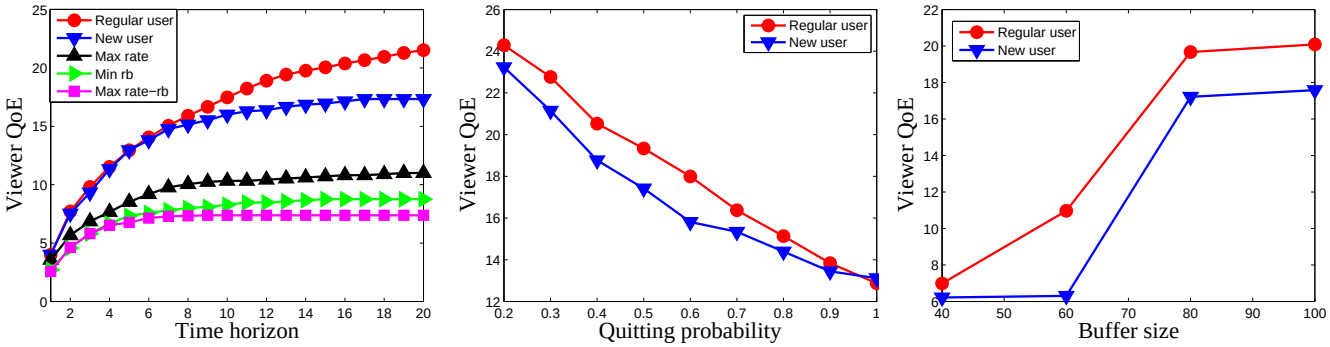

Fig. 4: (a) Performance comparison; (b) impact of quitting probability; (c) impact of buffer size.

TABLE I: Default parameter values

Parameter	Value
Number of chunks T	20
Video length N (second)	200
Available rates $\mathcal{E}$ (Mb/s)	$\{0, 2, 4\}$
Probability of quitting	$\{0.5, 0.2, 0.1\}$
Buffer size L (Mb)	80
Capacity space (Mb/s)	$\{2, 5.5, 10\}$

scheme for regular viewers has the highest viewer QoE. Since the benchmark schemes are based on the assumption of complete video sessions, the solutions are sub-optimal compared with the proposed schemes. The new viewers have lower utility than the regular viewers because the important information of viewer quitting probability is missing.

Fig. 4(b) shows the influence of quitting probability on viewer QoE. We vary the quitting probability for rebuffer interruption and keep others fixed. It is shown that as the quitting probability increases, the video rate consumed by the viewers decreases, because viewers easily abandon the video session once rebuffer interruption happens. As the quitting probability increases, the gap between regular viewers and new viewers decreases, because the viewer quits very soon, leaving few time for the rate adaptation to take effect.

Fig. 4(c) shows the influence of buffer size on viewer QoE. It is shown that as the buffer size increases, the viewer QoE increases because more video chunks can be stored when the network capacity is high, reducing the rebuffer interruption in case of capacity drop.

VI. Conclusion

Existing works on video rate adaptation implicitly presume that the viewers watch the entire video, but many measurement studies have shown that most viewers abandon the video sessions halfway. Therefore, rate adaptation schemes for complete video sessions may be sub-optimal for incomplete video sessions. In this paper, we propose a QoE-aware dynamic video rate adaptation scheme that considers incomplete video sessions and maximizes video quality that is truly consumed by the viewers. We leverage MDP to model the video rate adaptation problem, in which the rate selection is based on the network capacity, buffer state, and the viewers' quitting decision depends on video quality and rebuffer time. The performance evaluation results have shown that the QoE-aware video rate adaptation can greatly improve viewer QoE.

VII. Acknowledgement

This work was supported by grants from 973 project 2013CB329006, China NSFC under Grant 61173156, RGC under the contracts CERG 622613, 16212714, HKUST6/CRF/12R, and M-HKUST609/13, the grant from Huawei-HKUST joint lab, Program for New Century Excellent Talents in University (NCET-13-0908), Guangdong Natural Science Funds for Distinguished Young Scholar (No.S20120011468), the Shenzhen Science and Technology Foundation (Grant No. JCYJ20140509172719309), and China NSFC Grant 61472259.

References

[1] "Cisco visual networking index: Forecast and methodology, 2012−2017," 2014.

[2] "Cisco visual networking index: Global mobile data traffic forecast update, 2013−2018," 2014.

[3] 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," in *ACM conference on SIGCOMM*, 2014, pp. 187–198.

[4] L. De Cicco and S. Mascolo, *An experimental investigation of the Akamai adaptive video streaming*. Springer, 2010.

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

[6] 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," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 32, no. 4, pp. 719–733, 2014.

[7] G. Tian and Y. Liu, "Towards agile and smooth video adaptation in dynamic http streaming," in *ACM international conference on Emerging networking experiments and technologies*, 2012, pp. 109–120.

[8] L. De Cicco, V. Caldaralo, V. Palmisano, and S. Mascolo, "Elastic: a client-side controller for dynamic adaptive streaming over http (dash)," in *IEEE International Packet Video Workshop (PV)*, 2013, pp. 1–8.

[9] H. Yu, D. Zheng, B. Y. Zhao, and W. Zheng, "Understanding user behavior in large-scale video-on-demand systems," in *ACM SIGOPS Operating Systems Review*, vol. 40, no. 4. ACM, 2006, pp. 333–344.

[10] H. Yin, X. Liu, F. Qiu, N. Xia, C. Lin, H. Zhang, V. Sekar, and G. Min, "Inside the bird's nest: measurements of large-scale live vod from the 2008 olympics," in *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*. ACM, 2009, pp. 442–455.

[11] L. Chen, Y. Zhou, and D. M. Chiu, "Video browsing-a study of user behavior in online vod services," in *Computer Communications and Networks (ICCCN), 2013 22nd International Conference on*. IEEE, 2013, pp. 1–7.

[12] M. Draxler and H. Karl, "Cross-layer scheduling for multi-quality video streaming in cellular wireless networks," in *IEEE International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2013, pp. 1181–1186.
[13] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2009, vol. 414.