ICC 2021 - IEEE International Conference on Communications | 978-1-7281-7122-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICC42927.2021.9500784

# FRAB: A Flexible Relaxation Method for Fair, Stable, Efficient Multi-user DASH Video Streaming

Bo Wei
Department of Computer Science and Communication Engineering
Waseda University
Tokyo, Japan
weibo@aoni.waseda.jp

Hang Song
School of Microelectronics
Tianjin University
Tianjin, China
songhang168@tju.edu.cn

Jiro Katto
Department of Computer Science and Communication Engineering
Waseda University
Tokyo, Japan
katto@waseda.jp

***Abstract*—Dynamic adaptive streaming over HTTP (DASH) has been widely adopted in modern video streaming services. In DASH, the core technique is adaptive bitrate (ABR) control which can adjust the requested video bitrate level according to the network conditions to tradeoff between video quality and rebuffering risk. It is a challenge for the ABR methods in the scenarios when multiple DASH streaming users compete over the network bottleneck. This paper proposes a client-side ABR control method, flexible relaxation assisted by buffer (FRAB), to achieve fair, stable and efficient video streaming among different users. The idea of FRAB is to "relax" the change of the video quality based on current buffer level, which can enhance the stability of video streaming. Meanwhile, by flexibly adjusting the relaxation, the efficiency and fairness among all users are improved. FRAB is evaluated in real experiments under three different network conditions and compared with conventional multi-user ABR algorithms. Results indicate FRAB has the best performance in fairness, which reduces the unfairness by a maximum of 69.5% under real-world measured network condition. It also improves the efficiency by 71.3% comparing with PANDA, and enhances the stability by 73.3% comparing with TFDASH. The experiment results demonstrated that the proposed method has superior performances in multi-user DASH video streaming.**

***Keywords—adaptive bitrate control, flexible relaxation method, multiple users, DASH, fairness***

## I. Introduction

Video traffic has been the main part of the global IP traffic which account for over 80% of the total internet traffic [1]. It is essential to provide high-quality video streaming service to users. Among the video transmission technologies, dynamic adaptive streaming over HTTP (DASH) has become the *de facto* standard which allows the video session on client side to select the video bitrate adaptively [2]. In DASH, the video content is encoded at different bitrates which correspond to different quality levels. Then, the video is divided into segments with the same duration, e.g. 2 seconds, 4 seconds etc. When the available bandwidth is large, video segment with a high bitrate can be requested to enhance the visual experience quality. On the contrary, a low bitrate will be chosen to avoid rebuffering events when the bandwidth is low.

To achieve good performance of the video streaming, the adaptive bitrate (ABR) control plays the key role [3]. ABR algorithm decides when and which bitrate level to be chosen, thus well balance the video quality and interruption risk. There are several types of ABR methods. Rate-based (RB) method utilizes bandwidth prediction to choose the bitrate level [4-7]. Buffer-based method selects the video bitrate by monitoring the buffer occupancy [8, 9]. Other methods such as learning-based method [10, 11] and control-theoretic [12, 13] method use bandwidth prediction, buffer occupancy and other information to adaptively control the bitrate. In addition, with the rapid increasing of network structure, it becomes common that multiple users compete over a bottleneck network, e.g. local or home Wi-Fi networks through a router. In multi-user scenarios, although users are provided a fair shared bandwidth when they are downloading video simultaneously, their perceived TCP throughput may be different because of the ON-OFF phase of the request process [14]. This phenomenon causes problems such as unfairness, inefficiency, instability [15]. Many works have been engaged in improving the total performance of the multi-user video streaming [16-24]. However, this problem still remains challenging.

In this paper, a client-side ABR method, flexible relaxation assisted by buffer (FRAB) is proposed for fair, stable, efficient multi-user DASH video streaming. The main idea of FRAB is to "relax" the change of the video quality to enhance the stability of video streaming. Meanwhile, the relaxation threshold is flexibly adjusted according to the buffer level to avoid the OFF phase, which can improve the efficiency and fairness among all users. In FRAB, the throughput is estimated with harmonic mean and smoothing to make the client less sensitive to the change of the bandwidth. Furthermore, a flexible relaxation threshold is configured to decide whether to change the video bitrate level. When the buffer level is high, FRAB will keep the current bitrate or increase to a higher level to avoid buffer filling. If the buffer level is low, the algorithm will take a conservative action to avoid rebuffering. A testbed environment is built by using commercial home Wi-Fi router to investigate the performance of the FRAB. The comparison with other multi-user ABR algorithms is conducted. Results indicate that the proposed FRAB method has superiority in terms of fairness, stability, efficiency under different network conditions.

The rest of this paper is organized as follows. A brief review of the related work is given in Section II. The proposed FRAB method is detailed in Section III. In Section IV, the testbed construction is presented. Section V shows the experiment results and discussion. Finally, the conclusion is made and future work is depicted in Section VI.

This research is supported by JSPS KAKENHI Grant Number 20K14740 and Waseda University Grant for Special Research Projects (Project Number: 2020C-608).

## II. Related Work

In multi-user DASH streaming, the design goal of ABR strategy is to achieve fair, stable, efficient video transmission among different users. However, when some user is at OFF phase during the video streaming, other users may overestimate the fair shared bandwidth and select larger bitrate, resulting in unfairness problem. Some methods have been proposed to improve the fairness, efficiency and stability. Basically, these methods can be categorized into three kinds, server-side, network-side, and client-side approaches. Server-side methods are applied on server to shape the traffic to maximize fairness and efficiency of the end users [16-18]. Network-side approaches are employed on the network structure for optimal traffic management and resource allocation [19-22]. Client-side methods are implemented on the client directly, which are easy to deploy and do not increase the computing burden of server or change the network structure [4, 23, 24].

For the client-side methods, Jiang *et al.* proposed FESTIVE to improve the performance of multi-user DASH streaming [4]. In FESTIVE, a randomized scheduling is set to avoid the bias introduced by initial conditions. The stateful and delayed bitrate update are put forward to improve the stability. Li *et al.* proposed PANDA method where the "probe and adapt" principle is employed for video bitrate adaptation [23]. PANDA demonstrated its ability to reduce the instability of video bitrate selection without increasing the risk of buffer underrun. Zhou *et al.* proposed TFDASH, a throughput-friendly control scheme [24]. In TFDASH, the OFF periods is avoided to ensure fairness and efficiency. Meanwhile, a probability-driven rate adaption logic is proposed, taking into account a number of key factors to guarantee high-quality streaming.

## III. Flexible Relaxation Assisted By Buffer

In this section, the design of FRAB is presented. As a feature of the DASH standard, the video content is requested segment by segment. When the download of a certain segment is completed, the TCP throughput $r[n]$ can be measured as:

$$r[n] = \frac{S_n}{\Delta T_n} \tag{1}$$

where $S_n$ is the size of the requested $n$th segment, $\Delta T_n$ is the consumed time for downloading. Then, the harmonic mean is applied to $r[n]$ to estimate the shared bandwidth as follows:

$$r_h[n] = \frac{\sum_{k=n-M}^{n} \Delta T_k}{\sum_{k=n-M}^{n} \frac{\Delta T_k}{r[k]}} \tag{2}$$

where $M$ is the number of segments used for harmonic mean.

The basic ABR logic is to choose the maximum video bitrate that is lower than the estimated bandwidth, which is known as RB method. However, since many videos are encoded using variable bitrate (VBR), the actual size of one segment can be larger than bitrate times duration. Therefore, the time spent on downloading may be longer than the video duration of one segment. At the initial start stage, a quick establishment of the video session is essential since long waiting time annoys the user. It is also important to avoid rebuffering events which is the key factor to improve user engagement. For these reasons, the bitrate selection should be more conservative than regular RB at initial stage and low-buffer condition. In FRAB, the requested bitrate $q[n]$ is chosen as follows when the buffer occupancy is low:

$$q[n] = \max(l_{i-1} | l_i \leq r_h[n]) \quad B(t) \leq B_{\min} \tag{3}$$

where $B(t)$ is the current buffer occupancy state, $B_{\min}$ is the threshold to trigger low-buffer logic, and $l_{i-1}$ represents one level lower than the possible maximum bitrate.

Except for the low-buffer condition, bitrate choice can be more stable since the bandwidth estimation error can be absorbed by the buffer. In FRAB, the first relaxation is realized by smoothing the bandwidth estimation as follows:

$$\tilde{r}[n] = \tilde{r}[n-1] + \alpha \cdot (r_h[n] - \tilde{r}[n-1]) \tag{4}$$

where $\tilde{r}[n]$ is the relaxed bandwidth. $\alpha$ is the coefficient to determine the relaxation rate. Then, two flexible thresholds are defined to decide whether to change the requested bitrate level. The basic logic is decreasing the video level when $\tilde{r}[n]$ is lower than the current bitrate. However, if the buffer is sufficient, the downgrade of bitrate can be relaxed. The threshold $r_{\text{dec}}$ for decreasing the bitrate is calculated as:

$$r_{dec} = \tilde{r}[n] \cdot (1 + \gamma_1 \cdot max(0, B(t) - B_{low})) \tag{5}$$

where $B_{\text{low}}$ is the threshold to determine how much the buffer amount is available for absorbing the bitrate mismatch. $\gamma_1$ is the coefficient to determine the additive bandwidth. If the buffer occupancy is large enough, it is more confident to keep the current bitrate. With this relaxation threshold, the stability of the video streaming is improved. When the buffer occupancy reaches the upper bound $B_{\max}$, the client enters into the idle period where new video segment is not requested until the buffer drops lower than $B_{\max}$. This feature causes the overestimation of the bandwidth by some users, hence leads to the inefficient and unfair problems. The solution by FRAB is to greedily download the segment. When the buffer occupancy is close to $B_{\max}$, the bitrate is chosen aggressively to avoid buffer filling. The threshold $r_{\text{inc}}$ for increasing the bitrate is calculated as:

$$r_{\text{inc}} = \tilde{r}[n] \cdot (\beta + \gamma_2 \cdot \max(0, B(t) - B_{\text{high}})) \tag{6}$$

where $\beta$ is the coefficient in the basic RB method to judge whether to change the bitrate. $B_{\text{high}}$ is the threshold to examine how much the buffer is close to the max. $\gamma_2$ is the coefficient to determine the additive bandwidth for aggressive choice. By avoiding idle period, the efficiency and fairness can be enhanced. After calculating the two thresholds $r_{\text{dec}}$ and $r_{\text{inc}}$, the next bitrate is chosen as follows:

$$q[n] = \begin{cases} \max(l_i | l_i \leq r_{\text{dec}}), & q[n-1] > \max(l_i | l_i \leq r_{\text{dec}}) \\ \max(l_i | l_i \leq r_{\text{inc}}), & q[n-1] < \max(l_i | l_i \leq r_{\text{inc}}) \\ q[n-1], & \text{otherwise} \end{cases} \tag{7}$$

## IV. Experiment Configuration

### A. Testbed System Design

In order to evaluate the performance of the proposed FRAB method and compare with existing ABR methods, a multi-user

DASH streaming testbed is constructed. As shown in Fig.1, the testbed consists of a video content server, two routers, and various kinds of devices. The server is established with Apache HTTP Server 2.4 running on Ubuntu 14.04. The Dummynet [25] is installed on the server and utilized to shape the bandwidth, emulating different network conditions. One router is used as the aggregation router and the other is used as local/home router equipped with Wi-Fi connection function. During the experiment, the TCP traffic out of the server is manipulated by Dummynet, creating a constrained bandwidth network between server and clients. All the clients are connected to home router and compete over the bottleneck. Except for the traffic out of server, other connections are not constrained intentionally.

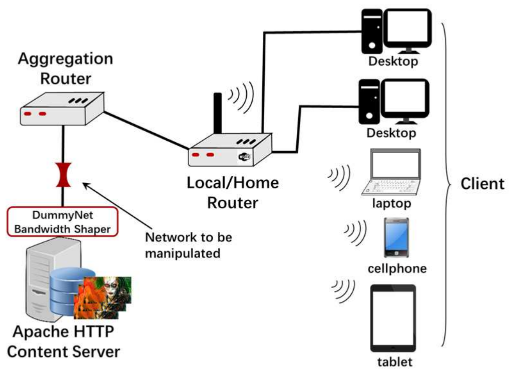
Aggregation Router
DummyNet Bandwidth Shaper
Network to be manipulated
Apache HTTP Content Server
Local/Home Router
Desktop
Desktop
laptop
cellphone
tablet
Client

Fig. 1. Network configuration for the multi-user DASH streaming.

On the client side, the framework dash.js (version 3.1.1) [26] is employed to implement DASH streaming. As shown in Fig.2, the developed ABR algorithms are implemented as additional functions with separate JavaScript files. The streaming logs such as HTTP requests, buffer occupancy state, requested bitrate levels are accessed and downloaded by using the newly added functions after the video session is over. The logs are saved as .txt files and the log files from different clients are collected for post-processing.

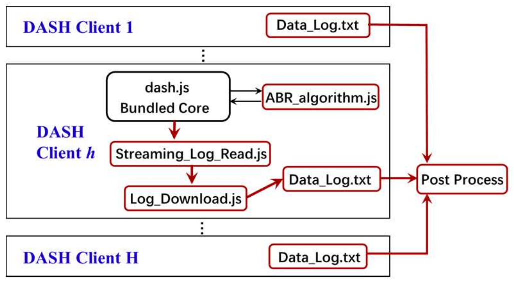

Fig. 2. Implementation of DASH on the client side.

## B. Performance Evaluation Setup

For performance evaluation, the video clip EnvivioDash3 is utilized [27]. This video is encoded at 9 quality levels with the bitrate of 200, 300, 480, 750, 1200, 1850, 2850, 4300, and 5300 kbps. The video clip is divided into 97 segments and each segment contains 2-second content. The length of the entire video is 193 seconds. All the contents are stored on the server. Google Chrome browser is deployed on all devices to play the video. In this paper, three scenarios are considered: (1) synthetic time-variant bandwidth, (2) synthetic time-variant bandwidth with random sudden change, (3) measured trace in real-world experiment from open dataset [28].

In FRAB, the number of segments for harmonic mean is chosen as 5. $B_{\min}$, $B_{\text{low}}$, and $B_{\text{high}}$ are set as 5, 10, 20 seconds, respectively. The coefficients $\alpha$, $\beta$, $\gamma_1$, and $\gamma_2$ are set as 0.3, 0.85, 0.05, 0.07. Besides the proposed FRAB method, other multi-user ABR methods, FESTIVE, PANDA, and TFDASH are also implemented for comparison. For fair comparison, the maximum buffer size is set as 30 seconds in FRAB, FESTIVE, and TFDASH. While in PANDA, the parameter $B_{\min}$ is set to be 26 seconds which keeps the buffer level in steady-state as 30 seconds. Other parameters in FESTIVE, PANDA, TFDASH are set as the default values provided in [4, 23, 24]. As for the $q_{\text{ref}}$ in TFDASH which is not given, it is set as 15 seconds in this paper.

## C. Evaluation Metrics

The unfairness, instability, inefficiency metrics are inherited from FESTIVE [4]. For completeness, the three metrics are depicted as follows.

- *Unfairness*: It is defined as $\sqrt{1-JFI(t)}$. $JFI(t)$ is the Jain fair index (JFI) at time $t$. In this paper, $t$ is the wall-clock time. The video bitrate downloaded by client $h$ at time $t$ is denoted as $q_h(t)$. Then the JFI is calculated as:

$$JFI(t) = \frac{\left(\sum_{h=1}^{H} q_h(t)\right)^2}{H \cdot \sum_{h=1}^{H} q_h^2(t)} \tag{8}$$

  where $H$ is the total number of clients.
- *Instability*: The instability metric for user $h$ at time $t$ is calculated as:

$$InSta(t) = \frac{\sum_{d=0}^{k-1} |q_h(t-d) - q_h(t-d-1)| \cdot \omega(d)}{\sum_{d=0}^{k-1} q_h(t-d) \cdot \omega(d)} \tag{9}$$

  where $\omega(d) = k - d$ indicates that larger penalty is given to more recent switches. $k$ is chosen as 20 seconds, equals to 10 segments for the instability calculation.
- *Inefficiency*: Suppose the available bandwidth at time $t$ is $C(t)$, the inefficiency of the streaming is calculated as:

$$InEff(t) = \frac{\max\left(0, C(t) - \sum_{h=1}^{H} q_h(t)\right)}{C(t)} \tag{10}$$

  When the sum of downloaded bitrate by all clients is larger than the bandwidth, the inefficiency is 0, indicating that the network is fully utilized.

In the post analysis, the metrics are first calculated per second and per client. Then, the averaged value is used for performance comparison.

## V. Experiment Results

In the experiment, two devices are utilized, a Windows 10 laptop computer and an Android cellphone. They are connected to the same home router through Wi-Fi. For each device, only one video session is executed at the same time. And two clients use the same ABR method in one round.

## A. Synthetic Time-variant Bandwidth

In this scenario, the bandwidth is set from 2000 to 5000 kbps, lasting from 30 to 60 seconds. The buffer occupancy and requested video bitrate logs of different users using various ABR methods are shown in Fig. 3. In FESTIVE, the buffer occupancy grows to the maximum value around 80 s as shown in Fig. 3(b). And the downloading process enters into periodic mode in both users. The discontinuity of the bitrate log indicates the idle time between two segment requests. The OFF phase can also be observed in PANDA as shown in Fig. 3(c). On the contrary, the buffer occupancies of FRAB and TFDASH do not reach the maximum and the download process is continuous without idle time. The measured throughput logs together with the constrained bandwidth are shown in Fig. 4. The throughput is plotted as rectangular column by segment. The width indicates the consumed time for downloading, and the height is the measured throughput. The area of the column is the size of the segment. As can be observed in Fig. 4(a) and (d), the bandwidth is fair-shared by two users in FRAB and TFDASH. The throughput of user 2 increases at the end. This is because user 1 has finished the video and quit the bottleneck competition. On the contrary, in FESTIVE and PANDA, the throughputs measured by the users are sometimes larger than the shared bandwidth as shown in Fig. 4(b) and (c). This is caused by the OFF phase when there may be only one active user.

Although the bandwidth is overestimated, the bitrate selection is conservative in FESTIVE owing to the stateful and delayed update mechanism as shown in Fig. 3(b). When the idle periods of all the users overlap, the network is not utilized, causing inefficient problem. In PANDA, the buffer approaches to the maximum rapidly as the bandwidth estimation increases additively by the probing mechanism. In TFDASH, the buffer occupancy is kept at lower level as shown in Fig. 3(d), compared with FESTIVE and PANDA. This is because TFDASH tends to take an aggressive action if the buffer occupancy surpasses the $q_{\text{ref}}$. The advantage of this strategy is that the utilization of the network is improved. However, the stability of the streaming is damaged. The bitrate drops and then increases around 130 s. This is because the aggressive bitrate selection causes continuous buffer draining. When the buffer becomes low, the bitrate selection has to be switched to lower level to avoid rebuffering. After the buffer is refilled, aggressive bitrate selection is made again. Since the bandwidth estimation is not included in the probability-driven model, the switch may increase because of the mismatch of bitrate and available bandwidth. In FRAB, both the bandwidth estimation and buffer occupancy are taken into consideration for bitrate selection. It can be observed in Fig. 3(a) that the selection is kept higher than the shared bandwidth from 45 to 70 s. This is owing to the relaxation mechanism that the buffer occupancy is sufficient to maintain the current bitrate selection. When the buffer is approaching the maximum around 130 s, the threshold to judge whether to raise the bitrate is increased. Then the aggressive action is taken and the idle period is avoided, improving the efficiency compared with FESTIVE and PANDA. Meanwhile, the bitrate selection is kept stable, showing the superiority over the TFDASH method.

The evaluation metrics of different ABR methods are shown in Fig. 5. While the instability is low in FESTIVE and PANDA, the unfairness and inefficiency is high. TFDASH performs well in terms of efficiency owing to the aggressive strategy in bitrate request. However, the instability value is large. The proposed FRAB method has the lowest unfairness, which reduces the unfairness by a maximum of 88.7%. Compared with PANDA, the inefficiency of FRAB is reduced by 68.1%. Meanwhile, the instability is reduced by 72.0% compared with TFDASH. The results demonstrate that FRAB has the best performance in fairness, efficiency, and stability.

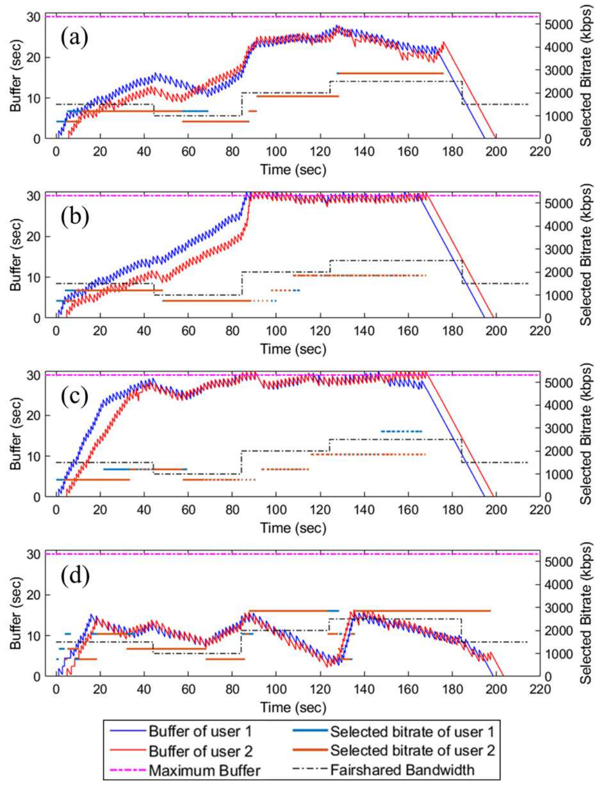

Fig. 3. Buffer occupancy and requested bitrate in scenarios 1 of different ABR methods: (a) FRAB. (b) FESTIVE. (c) PANDA. (d) TFDASH.

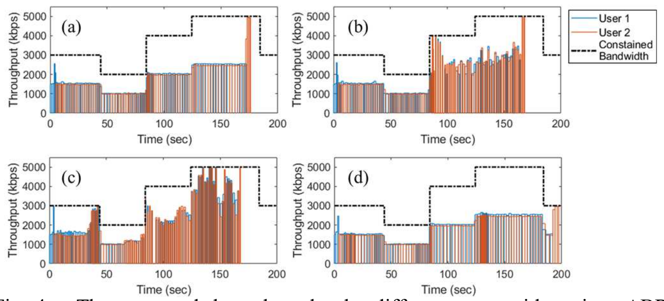

Fig. 4. The measured throughput log by different users with various ABR methods: (a) FRAB. (b) FESTIVE. (c) PANDA. (d) TFDASH.

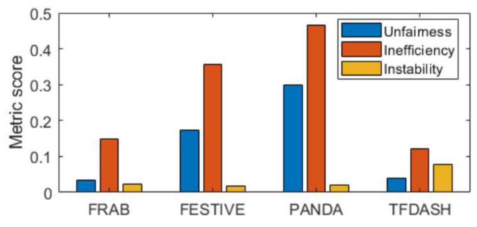

Fig. 5. Unfairness, inefficiency, and instability in scenario 1.

## *B. Synthetic Time-variant Bandwidth With Sudden Change*

In this scenario, there are some sudden changes happen in the bandwidth as shown in Fig. 6. The buffer occupancy and the bitrate logs are shown in Fig. 7. It can be observed that the sudden changes in the bandwidth have little influence on the bitrate selection by all methods. The bitrate log of FRAB in this scenario is almost the same as the one in the former scenario. The evaluation metrics of different ABR methods are shown in Fig. 8. FRAB maintains the best performance in fairness compared with conventional methods, reducing the unfairness by a maximum of 89.3%. It can be observed in Fig. 7(d) that the buffer occupancy of user 2 around 140 s is slightly higher than that of user 1 and the bitrate selection using TFDASH diverges by two users. Due to the diverse bitrate selection, the unfairness of TFDASH is increased in this scenario. Regarding to the inefficiency metric, FESTIVE and PANDA still do not utilize the network effectively. FRAB improves the efficiency by 70.8% compared with PANDA and the stability is retained. Compared with TFDASH, the instability is reduced by 76.1%. FRAB holds the best balance among the three metrics.

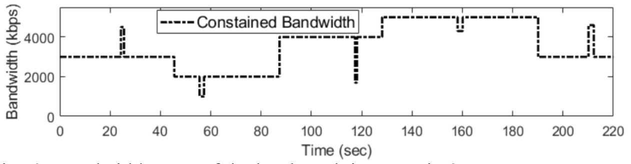

Fig. 6.  Bandwidth trace of the bottleneck in scenario 2.

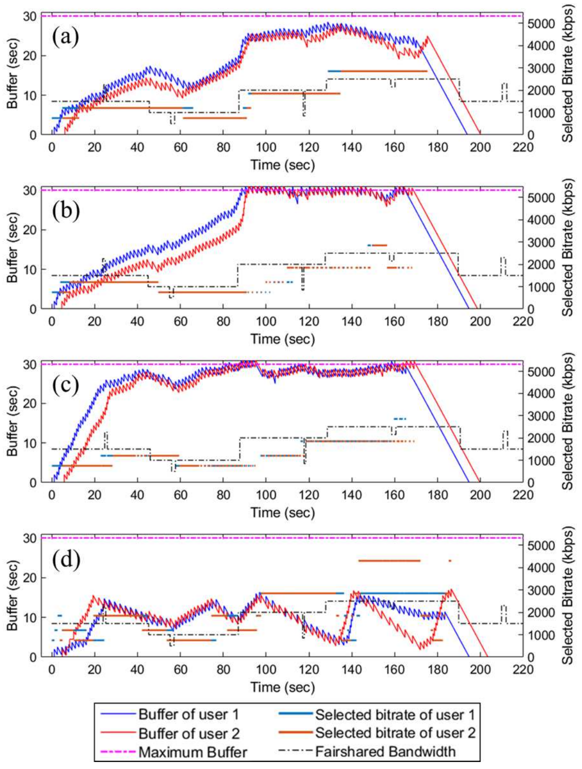

Fig. 7.  Buffer occupancy and requested bitrate in scenario 2 of different ABR methods: (a) FRAB. (b) FESTIVE. (c) PANDA. (d) TFDASH.

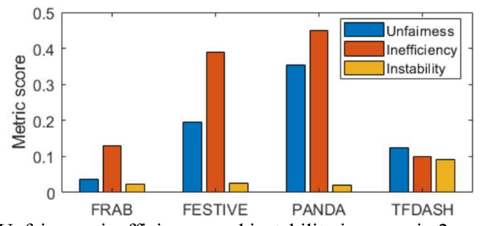

Fig. 8.  Unfairness, inefficiency, and instability in scenario 2.

## C. Measured Trace in Real-world

The performance evaluation is also carried out using the measured trace in real-world. This trace is from an open dataset measured with mobile high-speed downlink packet access (HSDPA) [28]. As shown in Fig. 9, there is a short low-bandwidth period around 30 s and a long low-bandwidth period from 140 s to 160 s. The buffer occupancy and the bitrate logs are shown in Fig. 10. It can be observed in Fig. 10(a) that the short low-bitrate period does not affect the bitrate selection due to the relaxation mechanism in FRAB. On the contrary, both FESTIVE and PANDA downgrade the bitrate after the low-bandwidth periods. The bitrate selection of TFDASH is not stable at the beginning.

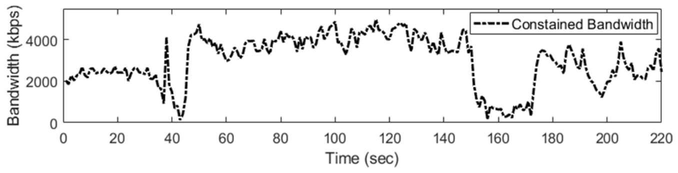

Fig. 9.  The measured trace of HSDPA.

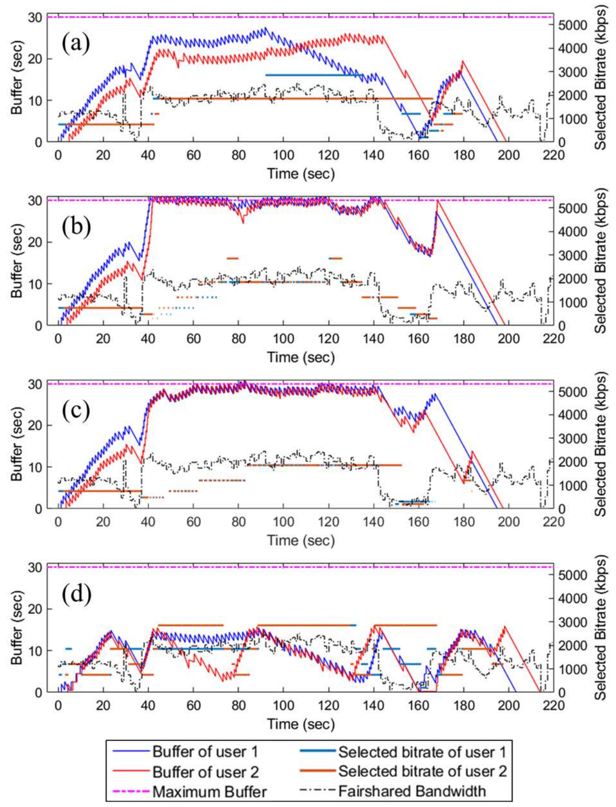

Fig. 10.  Buffer occupancy and requested bitrate in scenario 3 of different ABR methods: (a) FRAB. (b) FESTIVE. (c) PANDA. (d) TFDASH.

From 40 s to 140 s, the bottleneck bandwidth is relatively stable. FRAB upgrade the bitrate to the shared bandwidth rapidly, while FESTIVE and PANDA increase the bitrate gradually with idle periods in between. Therefore, the efficiency of FRAB is much better than FESTIVE and PANDA. TFDASH tends to choose the bitrate larger than the bandwidth, causing rebuffering events around 160 s. As shown in Fig. 10(a), user 1 choose a larger bitrate from 90 s because buffer occupancy is approaching the maximum. After buffer drops to about 15 s, the bitrate bounces back to the one near the shared bandwidth. When the bandwidth decreases from 140 s, FRAB downgrades the bitrate gradually because of the relaxation mechanism. PANDA reacts to the period of bandwidth decreasing sensitively and selects the low-level bitrates, trying to keep the buffer level. The unfairness, inefficiency, and instability metrics by different ABR methods under measured trace are shown in Fig. 11. In this scenario, FRAB shows the best performance in fairness, which reduces the unfairness by a maximum of 69.5%. In addition, the efficiency of FRAB is improved by 71.3% compared with PANDA, and the stability is not impaired. Comparing with TFDASH, the stability of FRAB is improved by 73.3%.

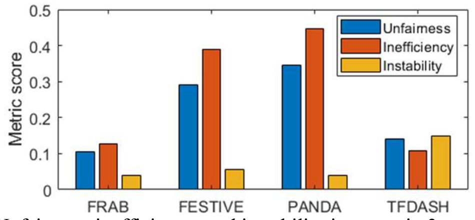

Fig. 11. Unfairness, inefficiency, and instability in scenario 3.

## D. Discussion

From the experiment results, it can be found that the proposed FRAB method has the best performance in fairness. As for the stability, although FESTIVE and PANDA perform well consistently, the idle periods impair the efficient utilization of the network. As for the efficiency, although it is enhanced dramatically in TFDASH by avoiding the idle period, the bitrate selection is aggressive which causes instable problems and results in the rebuffering events. FRAB takes advantages of the bitrate estimation, buffer occupancy information and preventing idle period. Therefore, it can improve the efficiency significantly compared with FESTIVE and PANDA as well as keep the stability of video streaming. Results demonstrate that the proposed method has a superior performance compared with conventional methods.

## VI. Conclusion

In this paper, a client-side ABR method, FRAB is proposed for multi-user DASH streaming. This method introduces a relaxation mechanism on bitrate selection considering the current buffer occupancy state to improve the stability of video streaming. In addition, the relaxation threshold is flexibly adjusted to prevent the idle period. Eliminating idle time improves the efficiency and fairness significantly. The proposed FRAB is evaluated in three different network scenarios and compared with conventional multi-user ABR methods. Results demonstrate that FRAB has the best performance in fairness. And it performs better than FESTIVE and PANDA in efficiency, and shows superiority over TFDASH in terms of stability. Future work includes testing FRAB with more users under various network conditions.

## References

[1] "Cisco Visual Networking Index: Forecast and methodology, 2017-2022," Cisco Syst., Inc., San Jose, CA, USA, 2018.

[2] T. Stockhammer, "Dynamic adaptive streaming over HTTP: Standards and design principles," in *Proc. ACM Multimedia Syst. Conf.*, Feb. 2011, pp. 133–144.

[3] J. Kua, G. Armitage, and Philip Branch, "A survey of rate adaptation techniques for dynamic adaptive streaming over HTTP," *IEEE Commun. Surv. Tutor.*, vol. 19, no. 3, pp. 1842-1866, 2017.

[4] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in HTTP-based adaptive video streaming with festive," *IEEE/ACM Trans. Netw.*, vol. 22, no. 1, pp. 326–340, Feb. 2014.

[5] B. Wei *et al*., "Evaluation of throughput prediction for adaptive bitrate control using trace-based emulation," *IEEE Access*, vol. 7, pp. 51346–51356, 2019.

[6] B. Wei *et al*., "TRUST: A TCP throughput prediction method in mobile networks," in *Proc. IEEE GLOBECOM*, Abu Dhabi, 2018, pp. 1–6.

[7] B. Wei *et al*., "HOAH: A hybrid TCP throughput prediction with autoregressive model and hidden Markov model for mobile networks," *IEICE Trans. Commun.*, vol. E101-B, no. 7, pp. 1612–1624, Jan. 2018.

[8] T. Huang *et al*., "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," in *Proc. ACM SIGCOMM 2014,* Chicago, IL, USA, 2014, pp. 187-198.

[9] K. Spiteri, R. Urgaonkar and R. K. Sitaraman, "BOLA: Near-optimal bitrate adaptation for online videos," in *Proc. IEEE INFOCOM 2016*, San Francisco, CA, USA, 2016, pp. 1-9.

[10] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with pensieve," in *Proc. ACM SIGCOMM 2017,* Los Angeles, CA, USA, 2017, pp. 197-210.

[11] T. Huang *et al*., "Stick: A Harmonious Fusion of Buffer-based and Learning-based Approach for Adaptive Streaming," *in Proc. IEEE INFOCOM 2020*, Toronto, ON, Canada, 2020, pp. 1967-1976.

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

[13] Z. Akhtar *et al*., "Oboe: Auto-tuning video ABR algorithms to network conditions," in *Proc. ACM SIGCOMM*, New York, 2018, pp. 44–58.

[14] S. Akhshabi *et al*., "What happens when HTTP adaptive streaming players compete for bandwidth?" in *Proc. ACM Int. Workshop Netw. Oper. Syst. Support Digit. Audio Video*, 2012, pp. 9–14.

[15] X. Yin *et al*., "On the efficiency and fairness of multiplayer HTTP-based adaptive video streaming," in *Proc. 2017 American Control Conference*, Seattle, WA, USA, 2017, pp. pp. 4236-4241.

[16] S. Akhshabi, L. Ananthakrishnan, C. Dovrolis, and A. Begen, "Server-based traffic shaping for stabilizing oscillating adaptive streaming players," in *Proc. NOSSDAV*, 2013, pp. 19–24.

[17] O. El Marai and T. Taleb, "Online Server-Side Optimization Approach for Improving QoE of DASH Clients," in *Proc. IEEE GLOBECOM*, Singapore, 2017, pp. 1-6.

[18] S. Altamimi and S. Shirmohammadi, "QoE-fair DASH video streaming using server-side reinforcement learning," *ACM Trans. Multimed. Comput. Commun. Appl.*, vol. 16, no. 2s, pp. 68:1-21, 2020.

[19] T. Ojanpera and H. Kokkoniemi-Tarkkanen, "Experimental Evaluation of a Wireless Bandwidth Management System for Multiple DASH Clients," in *Proc. IEEE GLOBECOM*, Washington, DC, 2016, pp.1-7.

[20] E. Ozfatura *et al*., "Optimal Network-Assisted Multiuser DASH Video Streaming," *IEEE Trans. Broadcast.*, vol. 64, no. 2, pp. 247-265, 2018.

[21] Z. Yan et al., "Toward guaranteed video experience: Service-aware downlink resource allocation in mobile edge networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 6, pp. 1819-1831, 2019.

[22] M. Catalan-Cid *et al*., "FALCON: Joint Fair Airtime Allocation and Rate Control for DASH Video Streaming in Sofware Defined Wireless Networks," in *Proc. NOSSDAV*, New York, USA, 2020, pp. 14–20.

[23] Z. Li *et al*., "Probe and adapt: Rate adaptation for HTTP video streaming at scale," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 4, pp. 719–733, 2014.

[24] C. Zhou *et al*., "TFDASH: A Fairness, Stability, and Efficiency Aware Rate Control Approach for Multiple Clients Over DASH," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 1, pp. 198-211, 2019.

[25] M. Carbone and L. Rizzo, "Dummynet Revisited," *ACM SIGCOMM Comp. Commun. Rev.*, vol. 40, no. 2, pp.12-20, 2010.

[26] https://github.com/Dash-Industry-Forum/dash.js

[27] https://dash.akamaized.net/envivio/EnvivioDash3/

[28] HSDPA Dataset. http://home.ifi.uio.no/paalh/dataset/hsdpa-tcp-logs