MUABR: Multi-user Adaptive Bitrate Algorithm based Multi-agent Deep Reinforcement Learning

Haoyue Yuan[1], Hancheng Lu[1, 2], Linghui Meng[1], Mengjie Liu[1]
[1]CAS Key Laboratory of Wireless-Optical Communications,
University of Science and Technology of China, Hefei, 230027, China
[2]Institute of Artificial Intelligence,
Hefei Comprehensive National Science Center, Hefei, 230027, China
Email: yhyue@mail.ustc.edu.cn, hclu@ustc.edu.cn

ICC 2022 - IEEE International Conference on Communications | 978-1-5386-8347-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICC45855.2022.9839087

***Abstract*—Adaptive bitrate (ABR) algorithms have been considered as efficient ways to improve the performance of video streaming by adaptively selecting appropriate video transmission bitrate. However, traditional ABR algorithms show poor adaptability to complex and dynamic networks, thus limit user's quality of experience (QoE). In the case of multi-user streaming, multiple users simultaneously requesting video from the server often compete for a bottleneck link. The bandwidth sharing problem should also be carefully considered, otherwise it will result in poor fairness and insufficient bandwidth utilization. To address these issues, this paper proposes a multi-user adaptive bitrate algorithm (*MUABR*) based on multi-agent reinforcement learning. *MUABR* takes the overall QoE of multi-user as the optimization goal, designs neural network structure and training method by reinforcement learning, so as to solve the problem of bitrate selection and poor adaptability when performing video streaming. Furthermore, the bandwidth allocation strategy adopted in *MUABR* ensures fairness to a certain extent and improves bandwidth utilization. Comprehensive experimental results show that *MUABR* has reached the desired goal. Compared with *Pensieve* and *Festive* algorithms, *MUABR* can improve the average QoE of users by about 7.5%-24.6%.**

***Index Terms*—Adaptive Bitrate Algorithm; Bandwidth Allocation; Reinforcement Learning; Video Streaming; QoE**

I. Introduction

With the rapid growth of wireless (i.e., 5G) and wired network bandwidth, video services have contributed the majority of global mobile data traffic [1]. In ordinary non-streaming transmission, users can only start watching after downloading. The emergence of *HTTP* Adaptive Streaming (*HAS*) solves the problem and is capable of play while downloading. *HAS* divides videos into multiple small chunks. When client requests, the video is transmitted in chunks, so that it can be played during downloading. Moving Picture Experts Group (*MPEG*) and the Third Generation Partnership Project (*3GPP*) proposed an international standard for Dynamic Adaptive Streaming over *HTTP*, known as *MPEG-DASH over HTTP* [2]. It can encode divided video chunks at different bitrate, so clients can freely select to download according to requirements.

In video streaming, the selected transmission bitrate determines the quality of video. Traditional adaptive bitrate (ABR) algorithms can be subdivided into rate-based and buffer-based. *Festive* [3] and *CS2P* [4] are both rate-based. They use predictive throughput models to select bitrate, but it is difficult for them to ensure the accuracy since the real environment is complex and dynamic. Buffer-based algorithms [5], [6] only consider the impact of buffer one-sidedly, which cannot make full use of the information known by client. So traditional algorithms have poor adaptability and unable to make a comprehensive and objective selection of bitrate.

The complexity of network is an important reason for the performance degradation of traditional algorithms. Reinforcement learning(RL) is an important branch of machine learning with the advantage of online learning. It can fine-tune models based on real environment to maximize the performance of algorithms. The ABR algorithms based on reinforcement learning can dynamically select bitrate according to network state and user status, so as to select appropriate chunks for transmission and improve user's quality of experience (QoE). Reference [7] introduced reinforcement learning into ABR algorithms firstly. But since the Q-learning used is only for low-dimensional state, the accuracy is relatively low. *Pensieve* [8] uses reinforcement learning to make and optimize bitrate selection strategy based on historical information and different network state. In [9], it combines reinforcement learning with video super-resolution technology to improve QoE. Reference [10] uses deep reinforcement learning to automatically obtain video bitrate adaptive decisions at each time stage, so as to provide better video streaming services. The above methods are all under the assumption that users are independent of each other. In fact, users often compete for bottleneck bandwidth. Every user has selfish nature, they will try to occupy more bandwidth to provide higher video quality for themselves.

Reference [11] points out that when multiple users compete for a bottleneck link, it may cause problems such as instability, unfairness, and insufficient bandwidth utilization. It mentioned in [12] that video servers should strive to improve the QoE of each user on the basis of ensuring fairness. For this reason, academics have proposed many solutions. *Festive* [3] combines three mechanisms of chunks scheduling, bandwidth estimation and bitrate selection to improve fairness and efficiency of multiple users. Reference [13] proposed a bandwidth prediction method. It not only increases the bandwidth utilization in multi-user scenarios, but reduces the instability of bitrate selection. Reference [14] performed QoE control and can process multiple users video streams at the same time. In [15], the proposed RL-based ABR algorithm considers buffer and video quality, improves QoE and also enables multiple users to ensure

application-level fairness when compete for a bottleneck link. Reference [16] mentioned that in the multiple users cases, machine learning and bandwidth allocation technologies have shown advantages in providing high quality and equal QoE for all users. In a word, bandwidth allocation among multiple users should be carefully considered.

In summary, some previously proposed algorithms have problems such as decision inaccurate, poor adaptability, insufficient bandwidth utilization, and failure to consider multi-user competition for bandwidth. In *MPEG-DASH*, the Server and Network Assisted DASH (*SAND*) standard has been proposed to control the bandwidth usage of clients, but it does not define any bitrate algorithms for clients [17]. In this paper, we propose a multi-user bitrate adaptive algorithm (*MUABR*) based on multi-agent deep reinforcement learning. It combines ABR algorithm with bandwidth allocation strategy to optimize the overall QoE and can improve viewing experience of each user as much as possible. To further verify the performance of the proposed algorithm, we conduct comparative experiments with *Pensieve* and *Festive* algorithms. Experimental results show both the average QoE and the bandwidth utilization of the *MUABR* algorithm are increased. The main contributions of this paper are summarized as follows:

- We make the bitrate decision model output an extra variable that can represent the urgency of user's demand for bandwidth occupancy, by utilizing the Hybrid Actor Critic (*HAC*). This lets users have a certain degree of initiative in subsequent decision-making.
- We design a "share convolutional layer and dedicate fully connected layer" network structure. The two output modules of the decision model can share historical information extracted by the convolutional layer, and then use different fully connected layers to decide the respective output.
- We integrate users status and historical bandwidth allocation information to design and modify the bandwidth allocation strategy so as to ensure fairness utmostly.
- We combine the ABR algorithm and bandwidth allocation strategy to build a multi-agent system. Each user can be regarded as an independent agent that executes bitrate decision logic. The server allocates bandwidth to each user according to the results output by each agent, achieving the objective of optimizing user's QoE.

The rest of this paper is organized as follows. Section II introduces problem scenario and modeling. The proposed *MUABR* method is detailed in Section III. Then, we show experimental results and analysis in Section IV. Finally, Section V concludes this paper.

II. System Model

A. Problem Statement

The multi-user video streaming scenario investigated in this paper is shown in Fig. 1. The HAS system includes a video server and *N* clients, and each link connected does not have a maximum limit on bitrate selection. Each client independently sends *HTTP* requests to download video chunks, but they all connect to the server and compete for the same bottleneck link. This scenario can be extended to many situations in reality, such as the competition among multiple users who are connected to the same edge server and share a common link. Each video chunk in the server will be encoded and stored at different bitrate. The downloaded chunk will be stored in the client buffer for users to watch later.

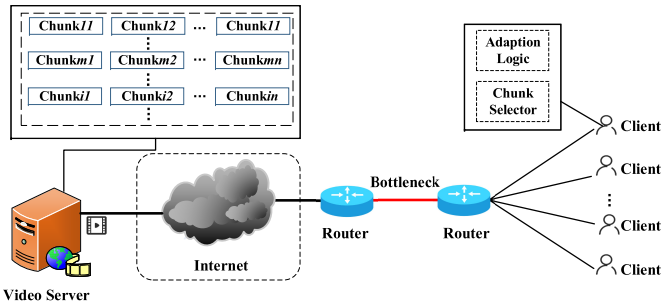


Fig. 1: Multi-user video streaming.

In the single-user scenario, there is no influence from others. But in the multi-user scenario, how to select appropriate video chunk and how to allocate bandwidth for users are problems that must consider. Each user will be in a user queue, and some may join or exit at any time. So we also need to consider how to solve the problem of variable length of historical information. When multiple users compete for bandwidth, it is very likely that some users have sufficient buffers but still occupy a larger bandwidth, while others are nearly to rebuffering but always use a smaller bandwidth. The fairness can not be guaranteed. So it is necessary for us to consider how to make a global decision to make all users have a high QoE.

We attempt to improve the overall QoE of multiple users while ensuring fairness so that each user can have a better viewing experience. To achieve it, *MUABR* algorithm consists of two parts: the ABR algorithm module deployed on the client side and the bandwidth allocation module deployed on the server side. In other word, we want to maximize the average QoE for all users as shown in Eq.1. In the equation, QoE_u represents $user_u$'s QoE, which is the sum of QoE for each chunk. b_{ui} represents the bitrate of $user_u$'s i-th chunk, and s_u is a variable related to the bandwidth allocation policy. The QoE will be determined by the adaptive algorithm of clients and the bandwidth allocation strategy of server.

$$\max_{b_{ui},s_u} \mathbb{E}[\sum_{u=1}^{U} QoE_u]. \quad (1)$$

B. Client ABR Algorithm Model

Adaptive bitrate (ABR) algorithms can select the appropriate bitrate for users according to the state of network. In the proposed algorithm, we use a convolutional neural network that can process historical information of different lengths. When the client decides bitrate, factors such as throughput and buffer occupancy will be considered. Coupled with the server-side strategy, we can improve the average QoE without affecting fairness. The QoE of $user_u$ is shown in Eq.2.

$$QoE_u = \sum_{i=1}^{N} q(b_i) - \beta_1 \sum_{i=1}^{N} \tau_i - \beta_2 \sum_{i=2}^{N} \varepsilon |b_i - b_{i-1}|. \quad (2)$$

For $user_u$, video starts from the first chunk and ends at the N-th chunk. And its QoE is mainly affected by video quality, rebuffering time and bitrate stability. b_i represents bitrate of the i-th chunk, and $q(b_i)$ is a function mapping bitrate to video quality that can be perceived by user. τ_i represents the rebuffering time of the i-th chunk. β_1, β_2 are the adjustment

factors of different factors in QoE evaluation index. ε is a coefficient to adjust bitrate switching, which describes the intensity of penalty. When the video clarity switches down, it gets larger and the penalty becomes stronger, and vice versa.

When a user begins to download the $(i+1)$-th chunk, the buffer can be concluded as Eq.3, where p refers to the play speed of the user. D_i is download time of the i-th chunk, which can be represented by average throughput over a period of time, L_i is the duration of the i-th chunk. In HAS, the duration of video chunks is same, generally 2-10s.

$$B_{i+1} = max\{max\{\frac{B_i}{p} - D_i, 0\} + L_i, 0\}. \tag{3}$$

After users feed back current state to the server, it can be allocated bandwidth. Therefore, in addition to bitrate, the ABR algorithm also needs to output priority, which can represent the urgency of users' demand for bandwidth occupancy. The algorithm judges user's status and network state, then outputs priority. As mentioned below, the smaller the buffer of user and the size of chunk to be downloaded, the more bandwidth will be allocated. In order to be consistent with the changes of these two factors, we design that the smaller user's priority value, the greater user's urgency, so the more bandwidth should be allocated. Since the client outputs one more priority, the optimization goal need to be updated. The overall optimization objective and constraint conditions can be expressed in Eq.4, where C_u is the optimization goal with urgency.

$$\begin{gathered} \max_{b_{ui}, s_u} \mathbb{E}[\textstyle\sum_{u=1}^{U} QoE_u + \alpha C_u], \\ s.t.\ QoE_u = \textstyle\sum_{i=1}^{N} q(b_i) - \beta_1 \sum_{i=1}^{N} \tau_i - \beta_2 \sum_{i=2}^{N} \varepsilon |b_i - b_{i-1}|, \\ B_{i+1} = max\{max\{\tfrac{B_i}{p} - D_i, 0\} + L_i, 0\}. \end{gathered} \tag{4}$$

In summary, the client's ABR algorithm has two outputs: the bitrate of next video chunk and priority. After deciding, the user will find out the size of the corresponding chunk through the MPD file provided by the server, then feed back it together with bitrate, its own priority, and the buffer size to the server.

C. Server Bandwidth Allocation Strategy Model

When multiple users compete for bandwidth, if users' egoistic behaviors affect the overall HAS QoE network, we'd better use a centralized control unit or based on cooperative client-agent to deal with. Only in this way can we have a fair distribution of QoE among multiple users [18]. Based on the principle, we add a bandwidth allocation strategy on the server side. It can allocate bandwidth of different sizes for users, so that we can improve QoE of users with poor status as much as possible while ensuring QoE of users with better status.

When serving multiple users, the server needs to maintain a user queue, which contains the buffer size, the chunk size, and priority of different users. The buffer is an indicator that can intuitively reflect the possibility of rebuffering. If the number of chunk in the buffer is lower than a threshold and no new chunks are downloaded to supplement, the users will be about to face rebuffering. The video size is provided by the user based on the MPD file of the server and the selected bitrate. The priority is the result of user's decision based on the buffer and historical throughput information.

The smaller the user's buffer is, the more likely it is to rebuffering, so the more bandwidth should be allocated. Considering the priority when allocating, we can more comprehensively learn user status information and avoid poor network conditions. The smaller the chunk to download is, the more bandwidth we allocate. This is because the client will decide bitrate based on the bandwidth allocated to the user historically. We allocate a larger bandwidth to users who have poor video quality and download small video chunks, so that they can gradually improve QoE in the future. It can also prevent users with high video quality from constantly requesting higher quality videos and users with low video quality from constantly requesting low quality videos. In this way, the algorithm can be operated normally and the fairness can be guaranteed.

III. METHODOLOGIES

In this section, we introduce the proposed (*MUABR*) algorithm, which includes the client ABR algorithm and the server bandwidth allocation algorithm. And we will explain the multi-agent framework and training method of the algorithm.

A. Client ABR Algorithm Design

In order to reduce burden, the ABR algorithm of each client remains independent, and users don't need to communicate with each other. According to the model introduced in the Section II, the decision of client algorithm should be affected by the state information such as throughput and buffer. We use hybrid actor-critic (*HAC*) algorithm from deep reinforcement learning. The action space contains two variables: the discrete action bitrate and the continuous action priority. Each type of action is decided by different output modules of actor, and the evaluation module shares a critic. The update formula of actor is shown in Eq.5, where π_b and π_p represent the action probability distribution of output bitrate and priority respectively. $A(s,a)$ is the advantage function of doing action a in state s. The update formula of critic is shown in Eq.6, where R is the reward function and $V(s)$ is the value of value function V in state s.

$$\begin{aligned} \theta^{\mu} = \theta^{\mu} + \alpha_1 \textstyle\sum_{n=1}^{N} \nabla_{\theta_a} (\log \pi_b\ (s_n, a_n) A(s_n, a_n) \\ +\zeta \log \pi_p\ (s_n, a_n) A(s_n, a_n\ )). \end{aligned} \tag{5}$$

$$\theta^{Q} = \theta^{Q} + \alpha_2 \textstyle\sum_{n=1}^{N} \nabla_{\theta\ Q} (R + V(s_{n+1}) - V(s_n))^2. \tag{6}$$

The reward function of each user is designed as Eq.7, including two parts: the reward calculated by QoE module and the reward SR related to bandwidth allocation. In fact, some users have higher priority, but smaller bandwidth can also meet their current needs, and their QoE will not be greatly affected. We will give extra rewards to these users who actively evade.

$$R = q(b_i) - \beta_1 \tau_i - \beta_2 \varepsilon |b_i - b_{i-1}| + \beta_3 SR. \tag{7}$$

We use convolutional neural network, and there are seven variables in the input layer, as shown in Fig. 2. The ability of one-dimensional convolution kernel to extract historical information can ensure getting all information of historical moments needed. When designing, we also consider the situation of users joining or exiting in multi-user scenarios, and the decision needs to be determined by historical information of variable length moments. We introduce *inception network* [19] structure to solve it, and design structures that "share convolutional layer and dedicate fully connected layer" to output discrete action bitrate and continuous action priority. We add *softmax* activation function to the last layer of neural

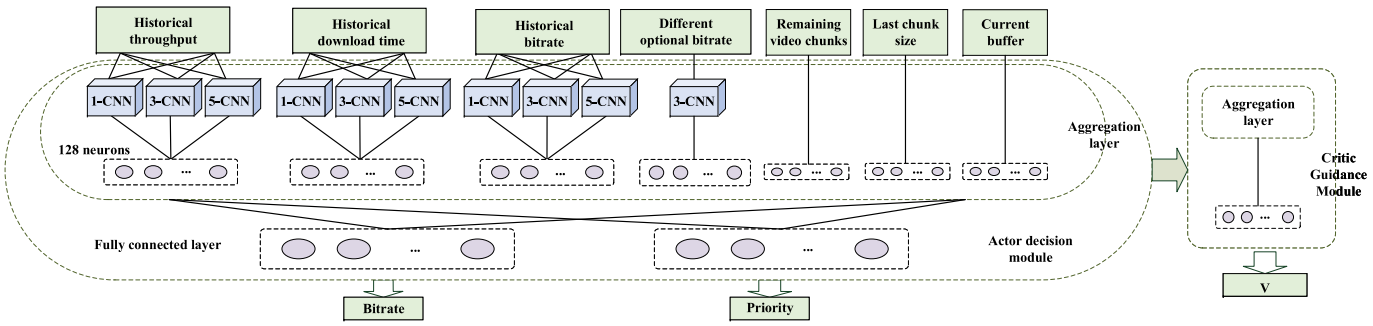


Fig. 2: Network structure of client ABR algorithm.

network that outputs bitrate to output the probability of each bitrate, then select action with the highest probability. We get the continuous action by outputting a *Gaussian* distribution and sampling. So we add *tanh* and *softplus* activation functions to the last layer of neural network that outputs priority to output mean and variance of the *Gaussian* distribution respectively. The priority range is 0-3, and we shift the position of *Gaussian* distribution during sampling to make sure. For a single user, the algorithm structure is shown in Fig. 2.

B. Bandwidth Allocation Strategy Design

When multiple clients compete for a bottleneck link, the server will allocate a reasonable bandwidth to each user according to a certain ratio. After downloading a chunk, if users do not finish watching, they will decide bitrate and priority of the next video chunk again, and the server will further reallocate bandwidth according to the updated user queue.

We linearly combine the buffer size, priority and chunk size of each user in the user queue, then calculate the allocation coefficient co. Furthermore, the server will calculate the modified allocation coefficient co_m according to co and the bandwidth distribution of each user at historical time, then we allocate bandwidth. It is worth noting that after a user finish downloading, the amount of video data to be downloaded by other users and the status information of the user are updated. So when designing the next bandwidth allocation, co and co_m also should be recalculated.

The allocation coefficient can be calculated shown in Eq.8, where η_1, η_2, and η_3 are balance coefficient of different factors. And we can know that the smaller the $buffer_{size}$, the value of priority pri and the video chunk size to be downloaded are, the larger the allocation coefficient is. In order to ensure the fairness of multiple users, the server will record the number of times x that each user is greater than the average link bandwidth in n historical bandwidth allocations. We get the correction value $\mu = (x + \lambda)/n$, where λ can ensure that μ is non-zero and can adjust bandwidth allocation strategy. Then the modified allocation coefficient can be calculated shown in Eq.9. The larger co_m is, the larger the allocated bandwidth is.

$$co = \frac{\eta_1}{buffer_{size} + 0.1} + \frac{\eta_2}{pri + 0.1} + \frac{\eta_3}{video_{size}}. \tag{8}$$

$$co_m = \frac{co}{\mu} = co \times \frac{n}{x + \lambda}. \tag{9}$$

C. Multi-agent framework and training method

Previously, we discussed the ABR algorithm of a single user and bandwidth allocation strategy on server. *MUABR* algorithm combines the two to form a multi-agent system. We regard each user as an independent agent, and the server allocates bandwidth reasonably based on the information provided by these agents. Every step here will improve QoE without affecting fairness of users as much as possible. The whole architecture is shown in Fig. 3.

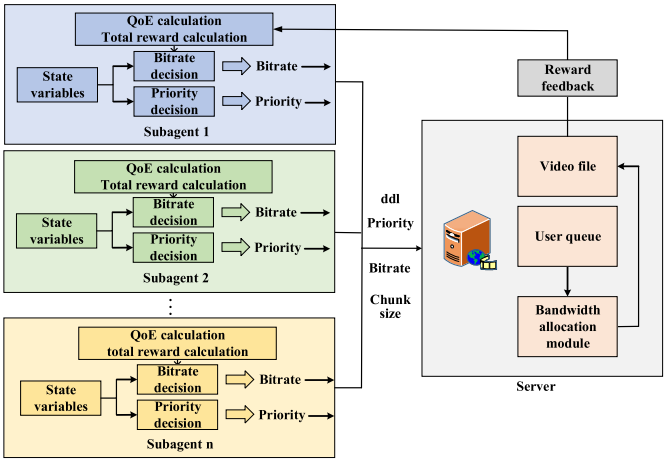


Fig. 3: Multi-user video streaming transmission architecture.

It is difficult for multi-agent to converge during training and to determine the actual number of users. In order to speed up the convergence, we firstly use single-agent model pre-training, where users can download continuously before the buffer is full. To simulate multi-user competition for bandwidth, we introduce a priority threshold value σ of 0.1. During the single-user pre-training process, if the priority output is less than σ, we will immediately transmit the next chunk. If the priority is higher, we will select a random number near the priority, and use it to generate a time. During this time, the download of the user is suspended, and it only plays video in the buffer. A bandwidth allocation reward will be given to it. Appropriately suspending the download of user is equivalent to reducing the bandwidth from the perspective of user perception. When training the single user adaptive algorithm, we use the hybrid actor-critic algorithm mentioned in section III-A.

In practical application, if there is only a single user in the network, the user directly deploys the single-user pre-training model, and the server will not use priority. If there are multiple users, each user will deploy the results trained in the multi-user scenario as shown in Alg. 1.

Algorithm 1 *MUABR* multi-agent training algorithm

Initialize parameters with pre-trained model.
repeat
  Determine the initialization bitrate and priority.
  $n = 0$
  repeat
    According to state variables to decide bitrate and priority.
    Send the decision result, buffer size and chunk size to server.
    Play video in the buffer and download the next chunk.
    Finish download and get scheduling rewards.
    $n+ = 1$
  until Update round N is reached.
  Get the model update gradient:
  $R = 0$
  for $n \in [0, N-1]$ **do**
    $R_n = r_n + \gamma R_{n+1}$
    Calculate the actor update gradient:
    $d\theta^{\mu} = \nabla_{\theta^{\mu}}(log\pi_b(s_n, a_n)(R_n - V(s_n; \theta^Q) + \gamma log\pi_p(s_n, a_n)(R_n - V(s_n; \theta^Q))$
    Calculate the critic update gradient:
    $d\theta^Q = \nabla_{\theta^Q}(R_n - V(s_n; \theta^Q)^2$
  end for
  Upload the updated gradient to the central agent and update model parameters.
until Convergence reached.

IV. EXPERIMENTS

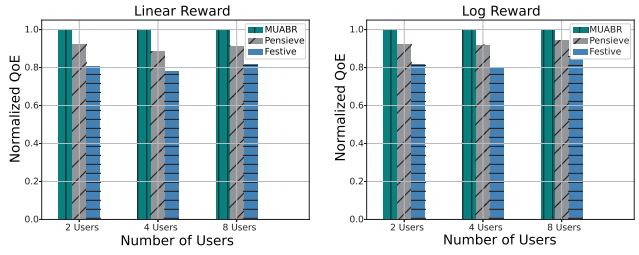
A. Experimental Setup

Pensieve [8] and *Festive* [3] are our contrast algorithms. We use *FCC* [20] and *HSDPA* [21] datasets, and the overall bandwidth of bottleneck link is the sum of each user's bandwidth at different time. The video uses *H.264/AVC* encoding. In detail, the simulation parameter settings are shown in Table I.

B. Results

In this section, we present experimental results. We observe the pros and cons of the three algorithms in QoE, bandwidth utilization, and fairness in multi-user scenarios. We will compare under the two conditions of linear reward $q(b_t) = b_t$ and logarithmic reward $q(b_t) = log(b_t)$ respectively.

QoE comparison: We calculate the QoE evaluation index using the sum of all users' QoE, as shown in the formula $QoE_u = \sum_{i=1}^{N} q(b_i) - \beta_1 \sum_{i=1}^{N} \tau_i - \beta_2 \sum_{i=2}^{N} \varepsilon |b_i - b_{i-1}|$. We compare the average QoE of the algorithms at different users, the results are shown in Fig. 4. The average QoE of *MUABR* is about 10% and 24.6% higher than *Pensieve* and *Festive* respectively in the case of linear reward. In the case of logarithmic reward, the average QoE of *MUABR* is about 7.5% and 21.4% higher than *Pensieve* and *Festive* respectively.



(a) linear reward (b) logarithmic reward

Fig. 4: QoE comparison.

Bandwidth utilization comparison: In order to compare bandwidth utilization, we select the bandwidth utilization of the test algorithms at different time and finally take the average of all the calculation results. The comparison results are shown in Fig. 5. Similarly, we evaluate the conditions of $q(b_t) = b_t$ and $q(b_t) = log(b_t)$ respectively. For *Festive*, since the algorithm has nothing to do with the linear and logarithmic rewards of the bitrate, its bandwidth utilization is the same curve.

TABLE I: Simulation parameter settings.

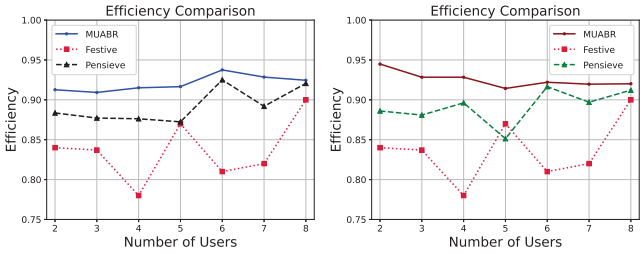
variable	parameter	value
QoE	β_1	4.3
	β_2	1
	ε	1/0.5, switch down/up
actor	ζ	1
the allocation coefficient	η_1	5
	η_2	1
	η_3	1
the modified allocation	λ	1
video playback speed	p	1
video bitrate		{300,750,1200,1850, 2850,4300} *kbps*
the length of video chunk		4s
the buffer size of client		60s

Bandwidth utilization can reflect the algorithm's utilization of link bandwidth. The simulation results shows the bandwidth utilization of the algorithms with different numbers of users. Comparing with *Pensieve* and *Festive*, the bandwidth utilization of *MUABR* has been increased by approximately 3.2% and 10% respectively in the case of linear rewards. And in the case of logarithmic rewards, the bandwidth utilization of *MUABR* has increased by approximately 3.8% and 10.6% respectively.

This is because *Festive* is difficult to dynamically change the adaption logic with the changes of network. When selecting bitrate, it is very dependent on the predicted value of network bandwidth, besides it does not make full use of the information of buffer occupancy on client. *Pensieve* is based on reinforcement learning, and the bandwidth utilization is slightly improved, but each user makes decisions independently, so there is still some room for improvement. *MUABR* adopts a combination of server bandwidth allocation strategy and client ABR algorithm. By reducing the bandwidth of users with better status appropriately, users in poor status can obtain higher bandwidth, thereby ensuring that they dare to choose a higher bitrate. Moreover, *MUABR* considers clients' information and networks when making decisions. Therefore, when the bandwidth of a good user is slightly reduced, it can still select a high bitrate based on historical throughput and sufficient buffers, which improves the bandwidth utilization of the system.

Fairness comparison: Similar to the evaluation in bandwidth utilization, in order to compare fairness, we compute the unfairness index of the test algorithm at different time, and take the average value of all the calculation results. The comparison results are shown in Fig. 6.

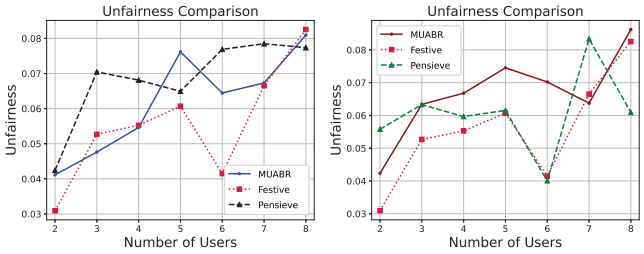
It can be seen that there is no monotonic relationship between the unfairness index and the number of users, and *Festive* performs best in terms of fairness. This is because *Festive* has specifically considered fairness of the multi-user



(a) linear reward (b) logarithmic reward

Fig. 5: Bandwidth utilization comparison.

system, and the main goal of the algorithm proposed in this paper is to maximize the overall QoE of users. When designing the bandwidth allocation algorithm and the client-side ABR algorithm, *MUABR* still improves the fairness by introducing priority, coefficient correction and other measures. Therefore, although the overall fairness performance is not as good as *Festive*, in some circumstances, *MUABR* can still perform better, so it can be seen that *MUABR* improves the video viewing experience of each user without major loss of fairness.



(a) linear reward (b) logarithmic reward

Fig. 6: Fairness comparison.

V. Conclusion

In this paper, we propose a multi-user bitrate adaptive algorithm *MUABR* based on multi-agent reinforcement learning. In order to be more adaptable to dynamic network, we mainly optimize the situation of multiple users compete for a bottleneck link. With the optimization goal of improving the overall QoE, we design a multi-agent framework and construct the client ABR algorithm and the server bandwidth allocation algorithm. Among them, the client ABR algorithm uses neural network and reinforcement learning to solve the problem of users choosing bitrate of the video to be downloaded. The bandwidth allocation algorithm of the server combine users' priority and historical state information to allocate bandwidth to solve the problem of multiple users' competition for bandwidth. After a series of experiments, we show that *MUABR* can improve bandwidth utilization and the overall QoE of users without affecting fairness much, thus achieving the purpose of improving users' viewing experience.

Acknowledgment

This work was supported by National Science Foundation of China under Grant U21A20452, U19B2044.

References

[1] Cisco, "Cisco annual internet report (2018–2023) white paper," 2020, https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html.

[2] I. Sodagar, "The mpeg-dash standard for multimedia streaming over the internet," *IEEE MultiMedia*, vol. 18, pp. 62–67, 2011.

[3] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive," vol. 22, 2014, pp. 326–340.

[4] Y. Sun, X. Yin, J. Jiang, V. Sekar, F. Lin, N. Wang, T. Liu, and B. Sinopoli, "Cs2p: Improving video bitrate selection and adaptation with data-driven throughput prediction," 2016, p. 272–285.

[5] 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 *Proceedings of the 2014 ACM Conference on SIGCOMM*, New York, NY, USA, 2014, p. 187–198.

[6] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "Bola: Near-optimal bitrate adaptation for online videos," *IEEE/ACM Transactions on Networking*, vol. 28, pp. 1698–1711, 2020.

[7] M. Claeys, S. Latré, J. Famaey, T. Wu, W. Van Leekwijck, and F. De Turck, "Design of a q-learning-based client quality selection algorithm for http adaptive video streaming," in *Adaptive and Learning Agents Workshop, part of AAMAS2013 (ALA-2013)*, 2013, pp. 30–37.

[8] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with pensieve," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, New York, NY, USA, 2017, p. 197–210.

[9] Y. Zhang, Y. Zhang, Y. Wu, Y. Tao, K. Bian, P. Zhou, L. Song, and H. Tuo, "Improving quality of experience by adaptive video streaming with super-resolution," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, 2020, pp. 1957–1966.

[10] Y. Guo, F. R. Yu, J. An, K. Yang, C. Yu, and V. C. M. Leung, "Adaptive bitrate streaming in wireless networks with transcoding at network edge using deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, pp. 3879–3892, 2020.

[11] S. Akhshabi, L. Anantakrishnan, A. C. Begen, and C. Dovrolis, "What happens when http adaptive streaming players compete for bandwidth?" in *Proceedings of the 22nd International Workshop on Network and Operating System Support for Digital Audio and Video*, New York, NY, USA, 2012, p. 9–14.

[12] L. De Cicco, G. Manfredi, S. Mascolo, and V. Palmisano, "Qoe-fair resource allocation for dash video delivery systems," in *Proceedings of the 1st International Workshop on Fairness, Accountability, and Transparency in MultiMedia*, New York, NY, USA, 2019, p. 33–39.

[13] 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*, vol. 32, pp. 719–733, 2014.

[14] H. Jin, Q. Wang, S. Li, and J. Chen, "Joint qos control and bitrate selection for video streaming based on multi-agent reinforcement learning," in *2020 IEEE 16th International Conference on Control Automation (ICCA)*, 2020, pp. 1360–1365.

[15] X. Wei, M. Zhou, S. Kwong, H. Yuan, S. Wang, G. Zhu, and J. Cao, "Reinforcement learning-based qoe-oriented dynamic adaptive streaming framework," *Information Sciences*, vol. 569, pp. 786–803, 2021.

[16] B. Wei, H. Song, S. Wang, and J. Katto, "Performance analysis of adaptive bitrate algorithms for multi-user dash video streaming," in *2021 IEEE Wireless Communications and Networking Conference (WCNC)*, 2021, pp. 1–6.

[17] S. Pham, P. Heeren, C. Schmidt, D. Silhavy, and S. Arbanowski, "Evaluation of shared resource allocation using sand for abr streaming," vol. 16, pp. 1–18, Jul. 2020.

[18] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld, and P. Tran-Gia, "A survey on quality of experience of http adaptive streaming," *IEEE Communications Surveys Tutorials*, vol. 17, pp. 469–492, 2015.

[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[20] F. C. Commission *et al.*, "Raw data-measuring broadband america.(2016)," https://www.fcc.gov/reports-research/reports/measuring-broadband-america/raw-data-measuring-broadband-america-2016.

[21] H. Riiser, P. Vigmostad, C. Griwodz, and P. Halvorsen, "Commute path bandwidth traces from 3g networks: Analysis and applications," in *Proceedings of the 4th ACM Multimedia Systems Conference*, New York, NY, USA, 2013, p. 114–118.