Proceedings Paper:
Naseh, D., Bozorgchenani, A. orcid.org/0000-0003-1360-6952 and Tarchi, D. (2025) Deep Reinforcement Learning for Edge-DASH-Based Dynamic Video Streaming. In: 2025 IEEE Wireless Communications and Networking Conference (WCNC). 2025 IEEE Wireless Communications and Networking Conference (WCNC), 24-27 Mar 2025, Milan, Italy. Institute of Electrical and Electronics Engineers (IEEE) ISBN 979-8-3503-6836-9

https://doi.org/10.1109/wcnc61545.2025.10978132

Deep Reinforcement Learning for Edge-DASH-Based Dynamic Video Streaming

David Naseh*, Arash Bozorgchenani†, Daniele Tarchi‡
*Department of Electrical, Electronic and Information Engineering "Guglielmo Marconi",
University of Bologna, 40136 Bologna, Italy, Email: david.naseh2@unibo.it
†School of Computer Science, University of Leeds, Leeds, LS2 9JT, UK, Email: a.bozorgchenani@leeds.ac.uk
‡Department of Information Engineering, University of Florence, 50139 Firenze, Italy, Email: daniele.tarchi@unifi.it

***Abstract*—Dynamic Adaptive Streaming over HTTP (DASH) is a promising solution to enhance the Quality of Experience (QoE) of mobile video services. In this paper, we consider an Edge-DASH scenario where two problems of Bitrate Allocation (BrA) and user-to-server allocation (USA) have been jointly formulated. Then, we exploit Deep Reinforcement Learning (DRL) algorithm to solve the USA problem and select the streaming point for users, which can be streaming from the Edge, Macro layer or cloud, and deliver the users the most appropriate bitrate respecting the QoE by solving the BrA problem. In the simulation results, we have demonstrated that our Deep Deterministic Policy Gradient (DDPG) outperforms the traditional solution in terms of bitrate allocation.**

***Index Terms*—DASH, multi-access edge computing, bitrate delivery, transcoding**

I. Introduction and Related Works

By 2025, mobile video streaming is expected to account for 76% of mobile data traffic [1]. With mobile subscriptions projected to reach 8.4 billion by 2029 and 5G networks covering up to 65% of the global population, ensuring Quality of Experience (QoE) for users has become a key research focus. The adoption of 5G is further driving mobile data traffic, enabling more immersive media formats. The diversity in user demands, influenced by network conditions and device capabilities, presents challenges for video service providers in maintaining optimal QoE. Adaptive Bit Rate (ABR) streaming and Dynamic Adaptive Streaming over HTTP (DASH) have emerged as solutions [2], allowing users to stream video in resolutions that suit their data rate and preferences. However, a network-only strategy is inadequate, as rate adaptation must consider user preferences and device factors like screen size and bandwidth [3]. While cloud computing supports DASH services, Cloud-DASH has drawbacks, such as high latency and core network congestion [4]. Multi-access Edge Computing (MEC) mitigates these issues by providing computation and storage resources closer to users at the network edge [5], [6]. MEC helps meet 5G's low-latency requirements, but an efficient User-to-Server Allocation (USA) mechanism is necessary to balance traffic across cloud and edge resources.

A cache hit occurs when a video chunk with the requested resolution is available, which makes ABR-aware edge caching more complex. In ABR streaming, having simply a video chunk in the cache is insufficient; the chunk must be stored at the required bit rate [7]. To address this issue, given the limited storage at the edge, we propose incorporating transcoding functionality at the Base Stations (BSs), which improves performance by eliminating the need to cache all possible bitrate levels. However, real-time video transcoding is highly computationally demanding, and transcoding multiple videos simultaneously can quickly deplete the processing capacity of MEC servers. Therefore, it is crucial to develop a bitrate delivery strategy that optimizes the use of processing resources.

Several studies have addressed the Bitrate Allocation (BrA) problem. Mehrabi et al. [8] proposed a greedy-based scheduling algorithm that periodically solves the USA and BrA problems, aiming to balance the load while considering various QoE metrics. However, their approach is entirely network-driven, overlooking client-side limitations and focusing on maximizing bitrate for all users. Bayhan et al. [9] studied BrA in WiFi Access Points to facilitate cache delivery. Their proactive approach considered a tolerable difference between the requested and delivered bitrates, using a compositional Pareto-algebraic heuristic. Although their model has some similarities with ours, they did not incorporate transcoding techniques, which play a crucial role in enhancing QoE. Some works have also considered hybrid edge-cloud architectures. Tao et al. [10] proposed an edge-cloud-assisted predictive adaptive streaming framework for mobile networks with unreliable data rate predictions, using slow fading to optimize long-term scheduling risk. Yan et al. [11] developed a hybrid edge-cloud framework to optimize client rate adaptation in cellular networks.

In this paper, we jointly address the problems of BrA and USA. First, we model the system and formulate a joint BrA-USA optimization problem. Then, we propose a Deep Reinforcement Learning (DRL) approach to solve this problem, determining the optimal streaming source for users, which can be from the edge, the Macro layer, or the cloud. Our solution ensures that users receive the most suitable bitrate while maintaining QoE by addressing the BrA problem. The main contributions of our work are as follows:

1) We introduce a reactive streaming strategy within a 4-tier network topology, where users can stream from either the edge (small cells), a Macro cell with wider coverage, or the cloud. Additionally, we incorporate transcoding capabilities in the edge layer,
2) We formulate a joint BrA-USA problem to optimize

bitrate delivery in a reactive, multi-layer network environment,

3) We develop a Deep Deterministic Policy Gradient (DDPG) method, which combines DRL and off-policy deterministic policy gradient approaches, to solve this joint problem. To the best of our knowledge, our approach is novel in this area.

II. System Model and Problem Formulation

A. System Model

We consider a macro-cellular network that covers a specific area, where a Macro Base Station (MBS) connects to the cloud via high-capacity links. The MBS contains a Macro Edge Server (MES), and within the Macro cell, S Small-cell Base Stations (SBSs) are co-located with Small-cell Edge Servers (SESs). The network provides video streaming services to Edge Clients (ECs), who request videos in varying resolutions. SESs store video content at multiple bitrates, whereas the MES holds the highest resolution versions.

Since ECs may prefer lower bitrates due to device constraints (e.g., battery life or data limits), our system adopts a reactive approach that takes user requests into account to ensure satisfactory QoE. ECs, represented by N users, connect to the closest base station, each requesting chunks from a video catalog, each chunk encoded at various bitrates. SESs can transcode video chunks to lower bitrates on requests, while the requested bitrate may differ from the delivered one.

There are N ECs scattered throughout the area, represented by indices $\mathcal{U} = \{u_1, u_2, \ldots, u_N\}$. Each EC connects to the nearest BS according to signal strength. The available videos are represented by $\mathcal{V} = \{v_1, v_2, \ldots, v_M\}$, and each video is divided into K chunks, each encoded at multiple bitrates. The SESs are capable of transcoding chunks to lower bitrates. ECs can request chunks from a range of bitrate levels, denoted as $\mathcal{O} = \{o_{\min}, o_{\max}\}$, and the bitrate of the kth chunk of the mth video is denoted as $o_{m,k}$. Each chunk has a fixed duration, typically between 2 and 10 seconds.

ECs' bitrate requests are denoted as $r_i^{o_{m,k}}(t)$, with the general notation $r_i(t)$ used for simplicity. The set of all EC requests is $\mathcal{R}(t) = \{r_1(t), \ldots, r_i(t), \ldots, r_N(t)\}$. The actual delivered content is represented by $\hat{\mathcal{R}}(t) = [\hat{r}_i(t)]_{i=1}^{N}$, where the delivered bitrate $\hat{r}_i(t)$ may differ from the requested bitrate.

The SESs provide caching and transcoding capabilities. Each SES has limited capacity, which allows it to store a certain number of videos and chunks out of the total of M videos and K chunks. Due to capacity constraints, not all bitrates can be cached, so each chunk is stored at a specific bitrate if cached at all.

At any given time t, an EC may request a video chunk at a certain bitrate. The requested chunk may be delivered from an SES, MES, or cloud, depending on availability. To maximize QoE, we prioritize delivering the requested chunk from the SES. If the SES cannot meet the request, the EC can go back to the MES or the cloud. The MES stores all videos at the highest bitrate, whereas the cloud stores all videos at all bitrates. The goal is to optimize the delivered bitrate, matching it as closely as possible to the requested bitrate while meeting QoE requirements. The following options describe the video delivery process:

1) Direct Edge hit: The requested chunk is cached at the SES in the requested bitrate.
2) Transcoding Edge hit: The chunk is cached at a higher bitrate, but is transcoded to the requested lower bitrate.
3) MES hit: The EC may access the MES if the chunk is either not cached at the SES or cached at a lower bitrate, or if MES provides better QoE or lower cost as MES stores all videos only with the highest bitrate.
4) Streaming from the cloud: If none of the above options is feasible, or if the cloud offers better QoE or lower cost, the EC streams from the cloud. This is an option; however, interaction with the cloud is preferred to be limited due to the traffic burden on the backhaul.

The ability of SESs to transcode is restricted by a shortage of computational resources. Let η_o^s represent the computing resources required for transcoding a chunk at a certain cached bitrate to a lower bitrate o. Transcoding to lower bitrates consumes more resources, i.e., $\eta_o^s < \eta_{o^-}^s$ if $o^- < o$. The indicator function $\mathbb{R}_i^o$ is equal to 1 if the request of EC i requires transcoding to bitrate o. The total computing resource constraint on SESs for transcoding is defined as:

$$\sum_{i=1}^{N_s} \sum_{o=1}^{O} \eta_o^s \cdot \mathbb{R}_i^o \leq \Omega^s \quad \forall s \tag{1}$$

where Ω^s is the maximum available computing resource for transcoding at an SES and N_s is the number of ECs covered by an SES. We assume that the cloud does not have any limitation in computation capacity. The cost of transcoding a chunk is measured by CPU usage on the cache servers.

B. Problem Formulation

Our objective is to maximize EC satisfaction by delivering the appropriate bitrate using optimal BrA and USA mechanisms, while considering edge resource constraints. Since each EC has multiple streaming sources (SBS, MBS, or cloud), the USA decision is made independently for each request. A key metric for effective BrA is avoiding stalling during streaming, which directly impacts QoE. However, USA is also tied to BrA since the choice of streaming source can depend on whether the requested chunk needs transcoding. Given the limited edge resources, all streaming options must be included in the optimization of bitrate and server allocation. In some cases, delivering a lower bitrate than requested may be preferable to prevent stalling and meet the USA requirements. Thus, the optimal USA decision may sometimes come at the cost of lower bitrate delivery. Both BrA and USA decisions are also influenced by the content cached in SESs. An effective caching strategy can improve streaming by increasing direct edge hits or reducing transcoding resource usage. However, due to the cache capacity limitations of each SES, only a limited

number of chunks at specific bitrates can be stored. Since this work does not focus on caching, we assume a random caching strategy at the edge.

To define the cost functions, we use the *Weber–Fechner* law, which explains the relationship between the actual changes in stimuli and human perception. This law has been shown to model user satisfaction in communication systems and multimedia applications, particularly following a logarithmic relationship for QoE [12], [13].

In the USA problem, each EC is assigned to a single server (SES, MES, or cloud) for streaming. Let $a_{i,j}$ be a variable that denotes whether EC u_i streams from the jth node out of SES, MES, or the cloud. The USA matrix $A \in \mathbb{R}^{N\times 3}$ shows the allocation for all ECs, with $\sum_{j=1}^{3} a_{i,j} = 1$ for each EC. Given the reactive approach to BrA, where ECs are served based on their requested bitrates, and the joint optimization of BrA and USA, the joint cost function for each EC request is defined as:

$$\Upsilon_{\phi_k^m}\left(\hat{r}_i(t), r_i(t)\right) = \alpha \log \frac{\max\left(\hat{r}_i(t), r_i(t)\right)}{\min\left(\hat{r}_i(t), r_i(t)\right)} \tag{2}$$

where α is a positive constant that reflects the significance of the cost, and $\hat{r}_i(t)$ is dependent on the value of $a_{i,j}$ as it identifies the streaming source, which is why it is a joint problem. The difference between requested and delivered bitrates at lower levels has a more significant impact on the cost compared to higher levels. This encourages the algorithm to allocate losses, if unavoidable, to higher bitrate levels, where user dissatisfaction will be less pronounced. Additionally, as the gap between the requested and delivered bitrates grows, so does the cost. Therefore, when $\hat{r}_i(t) = r_i(t)$, the cost is zero. We define the total joint BrA-USA cost function for all ECs:

$$\hat{\Upsilon}\left(\hat{\mathcal{R}}(t), \mathcal{R}(t)\right) = \sum_{i=1}^{N} \Upsilon_{\phi_k^m}\left(\hat{r}_i(t), r_i(t)\right) \tag{3}$$

We define the optimization problem as

$$\mathbf{P1}: \underset{\hat{\mathcal{R}}, A}{\text{minimize}} \left\{ \sum_{t=1}^{T} \sum_{i=1}^{N} \left(\Upsilon_{\phi_k^m}\left(\hat{r}_i(t), r_i(t)\right)\right) \right\} \tag{4}$$

subject to

$$\mathbf{C1}: \text{Eq. (1)} \quad \forall s \tag{5}$$

$$\mathbf{C2}: \sum_{j=1}^{3} a_{i,j} = 1, \quad \forall u_i \in \mathcal{U} \tag{6}$$

In (4), the goal is to minimize the difference between the requested bitrate and the delivered bitrate for all ECs over the time horizon T. The optimization focuses on the delivered bitrate vector $\hat{\mathcal{R}}$ and the server assigned for streaming, A. The transcoding computing constraint for each SES is represented in (5), while the USA condition, ensuring that each EC streams from either the SES, MES, or cloud, is shown in (6).

III. Proposed Solution

A. Preliminaries

We assume that each EC is associated with the SES that provides the highest SINR. In each time slot, some ECs make a request, while others are in playback mode, having made previous requests. Let $\mathcal{U}(t)$ represent the ECs connected to SES with no request and $\mathcal{U}(t)$ those with new requests. The total number of ECs in slot t is given by $|\mathcal{U}(t)| + |\mathcal{U}(t)| = N$.

We design a DRL-based algorithm located in the MES that generates BrA decisions for the ECs. Each small cell includes an environment E, states $\mathcal{S}$, and actions $\mathcal{A}$, with a reward function $r : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$. At each step t, the SES observes the state $s_t \in \mathcal{S}$, selects an action $a_t \in \mathcal{A}$ using policy π, and receives a scalar reward $r_t = r(s_t, a_t) \in \mathcal{R}$ proportional to the QoE. The agent transitions to the next state $s_{t+1} \in \mathcal{S}$ with probability $p(s_{t+1}|s_t, a_t)$. The actor's objective is to find the optimal policy π^* that maximizes the long-term expected reward:

$$R_t = \sum_{i=t}^{T} \gamma^{(i-t)} \cdot r(s_i, a_i), \tag{7}$$

where $\gamma \in [0, 1]$ is the discount factor.

The DRL framework, based on the Wolpertinger Policy [14], includes three components:

1) ***The actor***: The actor network finds a proto-action $\hat{a} \in \mathcal{A}$ based on the decision policy, updated after each step. The actor is parameterized by θ^μ and maps states $\mathcal{S}$ to actions $\mathcal{A}$, providing a proto-action $\hat{a}$ for the current state $\mu(s|\theta^\mu) = \hat{a}$.
2) ***K-Nearest Neighbors (KNN)***: The KNN maps the proto-action $\hat{a}$ to a set of valid actions $\mathcal{A}_k$ to simplify action selection in large spaces:

$$\mathcal{A}_k = g_k(\hat{a}_t), \tag{8}$$

where

$$g_k = \underset{a\in\mathcal{A}}{\overset{k}{\text{argmin}}} |a - \hat{a}|_2. \tag{9}$$

g_k is a k-nearest-neighbor mapping from a continuous space to a discrete set, and it returns the k actions in $\mathcal{A}$ that are closest to $\hat{a}$ by L_2 distance, i.e., $|a - \hat{a}|_2$.
3) ***The critic***: The critic evaluates the expanded actions from KNN and selects the one with the highest Q-value:

$$a_t = \arg\max_{a_j \in \mathcal{A}_k} Q(s_t, a_j). \tag{10}$$

The deterministic target policy for the critic is:

$$Q(s_t, a_j|\theta^Q) = \mathbb{E}_{r_t, s_{t+1}}\left[r(s_t, a_j) + \gamma Q(s_{t+1}, a_{t+1}|\theta^Q)\right], \tag{11}$$

where θ^Q are the critic network parameters. The action a_t that maximizes the Q-value is:

$$a_t = \arg\max_{a_j \in \mathcal{A}_k} Q(s_t, a_j|\theta^Q). \tag{12}$$

B. DRL-based Solution for BrA-USA

We introduce a DRL-based method located in the MES to solve $\mathbf{P1}$. Using the DDPG algorithm, the MES learns a dynamic BrA policy, selecting bitrate actions for ECs based on the observed environment states. The agent has no prior knowledge of the environment, which means it does not know the number of ECs, or bitrate demand, making the learning process model-free. The critic network $V(x)$ and the actor $\pi_{\theta_i}(o)$ are parameterized by $\theta = \{\theta_c, \theta_s\}$.

The DRL takes the demand profiles from N ECs, $\mathcal{R}(t) = \{r_1(t), \ldots, r_N(t)\}$, and outputs the BrA decision vector $\hat{\mathcal{R}}(t) = \{\hat{r}_1(t), \ldots, \hat{r}_N(t)\}$. Only $|\mathcal{U}(t)|$ ECs have new demand in slot t, while the others have an entry of 0. The DRL-based BrA-USA DDPG is defined as follows:

State Space: Agent's state is determined by the full system observation includes ECs' buffer length ($\mathbf{B}(t)$), and transcoding capacity at t, i.e. the agent's state is $s_t = [\mathbf{B}(t), \bar{\Omega}]$.

Action Space: Agent finds an action matrix $\bar{\mathcal{R}}(t) \in \mathbb{R}^{N \times \nu}$, where $\nu = 2 + |\mathcal{O}|$, representing all streaming options from SES (considering all transcoding options) plus the two options of MES and Cloud. The matrix is converted to vector $\hat{\mathcal{R}}(t) \in \mathbb{R}^{1 \times N}$ by assigning the highest probable bitrate using Softmax. The action space becomes $(N)^{\nu}$, and is continuous, optimized by DDPG.

Reward Function: The agent aims to minimize $\mathbf{P1}$ while meeting the EC constraints. The reward function r_t accounts for the allocated bitrate and the penalty on transcoding for all SESs:

$$r_t = -\Bigg(w_1 \cdot \sum_{i=1}^{N} \Upsilon_{\phi_k^m}\left(\hat{r}_i(t), r_i(t)\right) + w_2 \cdot \sum_{\forall s \in S} p_s^{\text{trans}}\left(\hat{\mathcal{R}}^s(t)\right) \Bigg) \quad (13)$$

where $\hat{\mathcal{R}}^s(t)$ is the vector of delivered bitrate to the users of the s-th SES. To address constraint violations in $\mathbf{P1}$, we define penalty function p_s^{trans} for transcoding resource constraints:

$$p_s^{\text{trans}}(\hat{\mathcal{R}}^s(t)) = max\Big(0, \sum_{i=1}^{N_s} \sum_{o=1}^{O} \eta_o^s \mathbb{R}_i^o - \Omega^s\Big). \quad (14)$$

We use a centralized critic-actors architecture. After selecting an action and receiving feedback (reward and next state), the critic updates the temporal difference (TD) error:

$$\delta^{\pi,\theta} = r_t + \gamma V(\mathbf{x}_{t+1}) - V(\mathbf{x}_t), \quad (15)$$

The critic is updated by minimizing the TD difference:

$$V^* = \arg\min_V (\delta^{\pi\theta})^2, \quad (16)$$

Actors are updated using policy gradients:

$$\nabla_\theta J(\theta) = E_{\pi\theta}\Big[\nabla_\theta \log \pi\theta(o,a)\delta^{\pi\theta}\Big], \quad (17)$$

$$\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi\theta(o,a)\delta^{\pi\theta}, \quad (18)$$

Fig. 1 illustrates our proposed architecture. The agent finds proto-actors, expands actions via KNN, and selects the highest Q-value actions for execution. The critic network evaluates K possible combinations of actions and updates the networks.

The training stage (Algorithm 1) initializes the state of the agent and ends at T_{max}. Experience tuples (s_t, a_t, r_t, s_{t+1}) are stored in buffer $\mathcal{M}$. After training for E_{max} episodes, the agent learns the BrA policy.

In testing, the agent loads trained parameters, interacts with the environment, and selects actions based on the output of the actor network.

Algorithm 1 Training stages of the DRL-based Solution

Input: $R(t)$ and $\Phi^{\text{SES}}(t)$, and $\forall i \in N$
Initialize the actor and critic networks' parameters randomly.
Initialize target networks $\theta^{\mu'} \leftarrow \theta^{\mu}$ and $\theta^{Q'} \leftarrow \theta^{Q}$.
Initialize an empty experience memory $\mathcal{M}$.
for each episode $e = 1, 2, ..., E_{max}$ **do**
  Generate an initial state s_1 randomly.
  for each step $t = 1, 2, ..., T_{max}$ **do**
    Determine the BrA action a_t given the demand of the EC, using the current policy network θ^{μ} and the exploration noise ϵ_μ.
    Execute action a_t, receive the reward r_t and observe the next state s_{t+1}.
    Store the tuple (s_t, a_t, r_t, s_{t+1}) in replay memory $\mathcal{M}$.
    Sample a mini-batch of K tuples from $\mathcal{M}$.
    Update the critic network by minimizing the loss L with the samples: $L = \frac{1}{K}\sum_{i=1}^{K}(r_i + \max_{a \in \mathcal{A}} Q(s_i', a|\theta^{Q'}) - Q(s_i, a_i|\theta^{Q}))^2$
    Update the actor network using the sampled policy gradient: $\nabla_{\theta^\mu} J = \frac{1}{K}\sum_{i=1}^{K} \nabla_a Q(s_i, a|\theta^Q)|_{a=a_i} \nabla_{\theta^\mu}\mu(s_i|\theta^\mu)$
  end for
  if t mod $\delta = 0$ **then**
    Update the target networks by $\theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'}$ and $\theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'}$.
  end if
end for

IV. Numerical Results

In this section, we evaluate the performance of the DRL-based algorithm through computer simulations. The simulation setup and results are described in the following subsections.

A. Simulation Setup

A single MBS is located in the center of a circular area with a radius of 500 m, and 5 SBSs are located within the area, each with a radius of 200 m. ECs are distributed in an area with a Poisson distribution with an expected density of S/N ECs per cell. The buffer length of each EC is considered 90 s [10]. The cache capacity at MES is enough to cache only the highest resolution for all videos (each video will have approximately 4.8 Gb).

We consider 10 videos each having an equal length of 10 minutes [15]. We also consider that different videos have different chunk sizes in the range [4 6] seconds. We select the six most popular resolutions for traditional systems, i.e., 240P, 360P, 480P, 720P, 1080P, and 1440P, each with a bitrate range which is obtained from YouTube similar to [3].

We characterize the popularity of videos using a Zipf-like distribution and sort the videos in $\mathcal{V}$ in descending order of their popularity $P_m = \{p_1, \ldots p_i \ldots, p_M\}, \sum_{i=1}^{M} p_i = 1$, where p_m

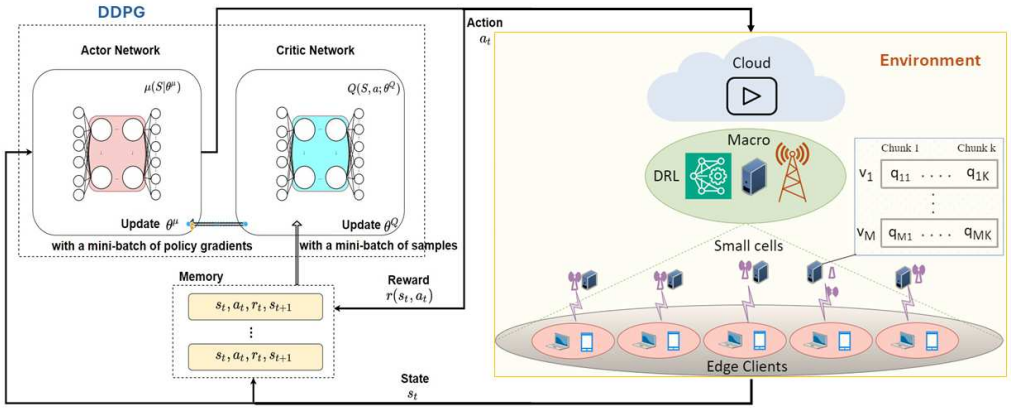

Fig. 1. Our considered architecture

represents the popularity of the ith rank video. Each EC selects a video based on its popularity. In a heterogeneous environment, devices have different interests in terms of resolution selection. Hence, we consider each EC uniformly at random selects one resolution level for its videos, however with a %5 probability for each time slot, the selected resolution downgrades to a lower resolution due to various reasons (e.g., interference, etc.). We also assume ECs with a probability of %10 start streaming from the first chunk, and with a %90 probability select a chunk randomly from the middle of the video in order to show real-world-like user behavior. ECs download the chunks until the play-out buffer is full; such an aggressive approach is also exploited on YouTube [16].

After the USA and the allocation of the bitrates, bandwidth is fairly allocated to the ECs. The chunks are added to the buffer after they are downloaded and the buffer decreases as the chunks are being displayed. For processing capacity, we set $\Omega^s = 25$GHz, which is the maximum number of CPU cycles per second. We also set the number of CPU cycles per Byte 5900 [15].

In the DRL simulation, we use the DDPG agent consisting of two networks: (1) the Actor network has three layers with dimensions 400, 300, and the action dimension, and (2) the Critic network takes as input both the state and action, with layers 400, 300. Both networks use ReLU activations, with Tanh activation in the output of the actor network.

The agent updates its parameters using Adam optimizers, with learning rates of 1×10^{-4} for the actor and 1×10^{-3} for the critic. It uses a replay buffer of size 1×10^{6} to store experiences and sample mini-batches of size 64 for training. The discount factor (γ) is set to 0.99, and the soft update coefficient (τ) for the target networks is 0.005. The agent's action selection is deterministic and is bounded by the maximum action value.

B. Simulation Results

To assess the convergence of the DRL-based approach, we conducted an experiment spanning 100 episodes. As illustrated in Fig.2, the rewards for both BrA and transcoding converge after approximately 25 episodes of exploration. Similarly, the joint reward (as defined in (13)) reaches its maximum value around the same point. Please note that as defined in (2), BrA is positive but has been negated since it represents a cost.

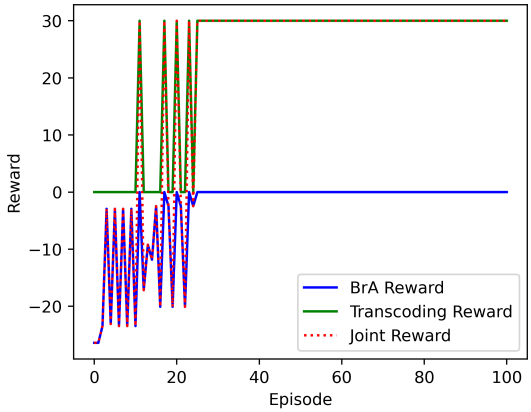

Fig. 2. Bitrate allocation, transcoding and joint rewards vs number of episodes

Fig.3.a shows the bitrate error over several episodes, representing the difference between the requested and allocated bitrates. As observed, the error is initially high but drops to zero after approximately 25 episodes, demonstrating the learning and improvement in bitrate allocation over time, which is the primary objective of this study, as defined in **P1**. This trend is further illustrated in Fig.3.b, which displays the Root Mean Square Bitrate Error (RMSBRE), highlighting the reduction in bitrate delivery errors for video chunks over time. In both figures, the proposed solution is compared with a randomized bitrate delivery for clearer contrast.

In order to better understand the source of delivery/streaming and the amount of edge transcoding, we have conducted two experiments, and the results are shown in Figs.4 and 5. As seen, almost all of the ECs count on streaming from the edge in the first episodes. However, since this violates the transcoding constraint **C1**, the agent learns this through time and reduces the edge streaming; as a result, edge transcoding, and instead streams more from the MBS and Cloud.

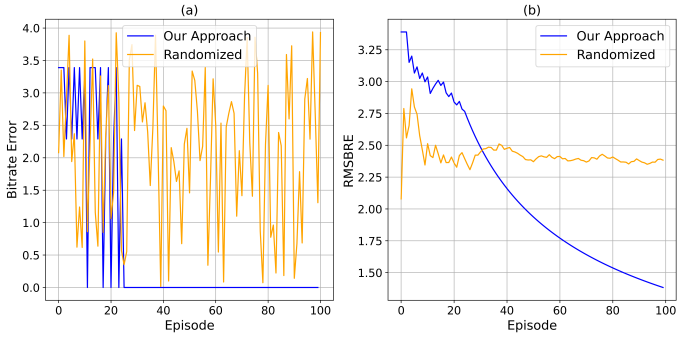

Fig. 3. (a) Bitrate error and (b) Root Mean Square of Bitrate error vs episodes

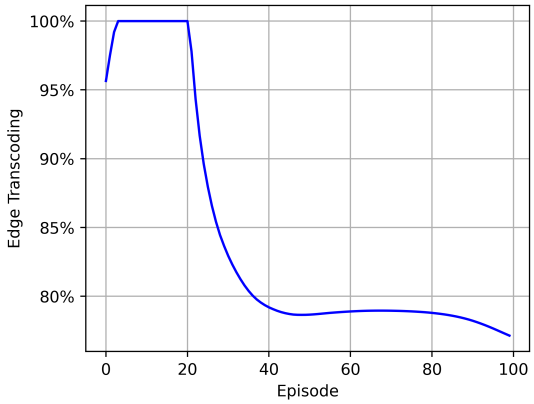

Fig. 4. Transcoding percentage over the edge vs number of episodes

To conclude, the agent initially struggles to deliver the correct bitrate (as shown in Figs. 2 and 3) and exceeds its transcoding capacity, resulting in lower rewards (having a low reward in Fig. 2). Moreover, as depicted in Figs.4 and 5 the agent transcodes for most ECs. However, this does not lead to a lower bitrate because the agent transcodes at an incorrect level. Over time, the agent learns to adjust its transcoding process (both when and how much to transcode), reducing the edge transcoding and flexibly utilizing all tiers for efficient streaming.

V. Conclusion

In this paper, we proposed a DRL-based solution for the joint optimization of BrA and USA in an Edge-DASH environment. Our DDPG approach significantly improves the system's ability to effectively allocate bitrates while maintaining high QoE standards. The simulation results validate the approach, showing that the agent improves over time, reducing bitrate errors and transcoding violations by selecting appropriate streaming sources, thus balancing the use of edge, macro, and cloud resources. The proposed method demonstrates superior performance compared to traditional solutions, offering a promising strategy for managing video streaming in 5G-enabled multi-tier networks.

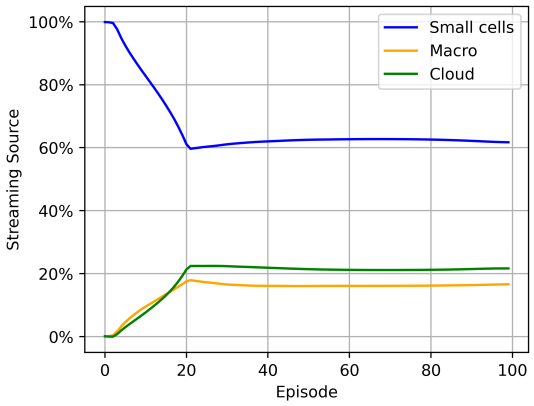

Fig. 5. Streaming source vs number of episodes

References

[1] *Ericsson Mobility Reports*, Ericsson, Nov 2021.

[2] S. Kumar, D. S. Vineeth, and A. F. A, "Edge assisted DASH video caching mechanism for multi-access edge computing," in *2018 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, Indore, India, 2018, pp. 1–6.

[3] D. Wang, Y. Peng, X. Ma, W. Ding, H. Jiang, F. Chen, and J. Liu, "Adaptive wireless video streaming based on edge computing: Opportunities and approaches," *IEEE Trans. Serv. Comput.*, vol. 12, no. 5, pp. 685–697, 2019.

[4] M. Nafeh, A. Bozorgchenani, and D. Tarchi, "Joint scalable video coding and transcoding solutions for fog-computing-assisted DASH video applications," *Future Internet*, vol. 14, no. 9, Sep. 2022.

[5] A. Bozorgchenani, D. Tarchi, and G. E. Corazza, "Centralized and distributed architectures for energy and delay efficient fog network-based edge computing services," *IEEE Trans. Green Commun. Netw.*, vol. 3, no. 1, pp. 250–263, 2019.

[6] A. Bozorgchenani, F. Mashhadi, D. Tarchi, and S. A. Salinas Monroy, "Multi-objective computation sharing in energy and delay constrained mobile edge computing environments," *IEEE Trans. Mobile Comput.*, vol. 20, no. 10, pp. 2992–3005, 2021.

[7] H. A. Pedersen and S. Dey, "Enhancing mobile video capacity and quality using rate adaptation, RAN caching and processing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 2, pp. 996–1010, 2016.

[8] A. Mehrabi, M. Siekkinen, and A. Ylä-Jääski, "Edge computing assisted adaptive mobile video streaming," *IEEE Trans. Mobile Comput.*, vol. 18, no. 4, pp. 787–800, 2019.

[9] S. Bayhan, S. Maghsudi, and A. Zubow, "EdgeDASH: Exploiting network-assisted adaptive video streaming for edge caching," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 2, pp. 1732–1745, 2021.

[10] L. Tao, Y. Gong, S. Jin, and J. Zhao, "Energy-efficient predictive HTTP adaptive streaming in mobile cellular networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 11 069–11 083, 2018.

[11] Z. Yan, J. Xue, and C. W. Chen, "Prius: Hybrid edge cloud and client adaptation for HTTP adaptive streaming in cellular networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 1, pp. 209–222, 2017.

[12] W. Zhang, Y. Wen, Z. Chen, and A. Khisti, "QoE-driven cache management for HTTP adaptive bit rate streaming over wireless networks," *IEEE Trans. Multimedia*, vol. 15, no. 6, pp. 1431–1445, 2013.

[13] P. Reichl, B. Tuffin, and R. Schatz, "Logarithmic laws in service quality perception: where microeconomics meets psychophysics and quality of experience," *Telecommunication Systems*, vol. 52, no. 2, pp. 587–600, 2013.

[14] G. Dulac-Arnold, R. Evans, H. van Hasselt, P. Sunehag, T. Lillicrap, J. Hunt, T. Mann, T. Weber, T. Degris, and B. Coppin, "Deep reinforcement learning in large discrete action spaces," *arXiv preprint arXiv:1512.07679*, 2015.

[15] S. Rezvani, S. Parsaeefard, N. Mokari, M. R. Javan, and H. Yanikomeroglu, "Cooperative multi-bitrate video caching and transcoding in multicarrier NOMA-assisted heterogeneous virtualized MEC networks," *IEEE Access*, vol. 7, pp. 93 511–93 536, 2019.

[16] H. W. Barz and G. A. Bassett, *Multimedia Networks: Protocols, Design and Applications*. Wiley, 2016.