Distributed Precoding for eMBB and URLLC Traffic in Cell-free O-RAN: A Multi-agent Reinforcement Learning Framework

Mohammad Hossein Shokouhi and Vincent W.S. Wong
Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, Canada
email: {mhshokouhi, vincentw}@ece.ubc.ca

***Abstract*—The integration of cell-free multiple-input multiple-output (MIMO) technology within the open radio access network (O-RAN) architecture addresses the growing need for decentralized, scalable, and high-capacity networks that can support different applications and use cases. In this paper, we propose a distributed precoding framework to support enhanced mobile broadband (eMBB) and ultra-reliable low-latency communications (URLLC) traffic in cell-free O-RANs, where each user is served by multiple open radio units (O-RUs). We consider short packet transmission in order to satisfy the latency requirements of URLLC traffic. We formulate a precoding optimization problem to maximize the aggregate throughput of eMBB users subject to the latency constraint of URLLC users. We propose a multi-agent deep reinforcement learning (DRL) algorithm to solve the formulated problem in a distributed manner. In particular, an actor-critic DRL agent is assigned to each O-RU. The actor determines the precoding matrices. The critic evaluates the actor's policy. The critics have global knowledge of all agents' policies, which stabilizes training and enables collaboration among agents. Simulation results show that the proposed algorithm provides an aggregate eMBB throughput improvement by up to 55.4% when compared with three state-of-the-art baseline schemes.**

I. Introduction

With the emergence of new technologies such as cell-free multiple-input multiple-output (MIMO) and artificial intelligence (AI), effective resource management of wireless networks requires open solutions that provide access to data and analytics and enable data-driven optimization. To address this need, the open radio access network (O-RAN) paradigm has been proposed in the literature [1]. O-RAN leverages disaggregated and virtualized components that are interconnected through open interfaces. The O-RAN architecture splits base station functions into three components [1]: the open central unit (O-CU), the open distributed unit (O-DU), and the open radio unit (O-RU), collectively known as E2 nodes. O-RAN also features two RAN intelligent controllers (RICs): the near-realtime (RT) RIC, which manages the network in a near-realtime (10 ms to 1 second) time scale, and the non-RT RIC, which operates at a non-realtime (over 1 second) time scale. The near-RT RIC consists of multiple applications called xApps which support optimization routines and machine learning (ML) workflows. In [2], dApps are introduced that run on O-DUs to support realtime (below 1 ms) control loops.

Mobile wireless networks need to support different mobile applications and services with diverse quality-of-service requirements. These applications fall into three categories: ultra-reliable low-latency communications (URLLC), enhanced mobile broadband (eMBB), and massive machine-type communications (mMTC). URLLC applications are delay-sensitive and transmit short packets sporadically, while eMBB requires high throughput. mMTC supports a massive number of Internet of things (IoT) devices that transmit data at a low data rate [3].

Various algorithms have been proposed in the literature to support eMBB and URLLC traffic on shared network resources. In [4], an xApp is deployed in the near-RT RIC which uses stochastic network calculus technique to allocate physical resource blocks (PRBs) to URLLC users. In [5], the URLLC requirements are satisfied by using the puncturing technique that schedules URLLC packets over eMBB transmissions. Some recent works utilized deep reinforcement learning (DRL) algorithms for resource allocation. In [6], a DRL algorithm is proposed to determine the numerology, bandwidth, and transmit power for eMBB and URLLC slices. In [3], each O-RU has a DRL agent in near-RT RIC for PRB and power allocation. A global model is trained in non-RT RIC using the historical information from all agents. The model is then sent to the agents for local execution. The aforementioned works [3]–[6] consider single-input single-output (SISO) communications and employ puncturing technique to serve eMBB and URLLC users using shared resources. Additionally, they assume that the scheduling algorithm module has prior information about the URLLC traffic for the entire time slot in advance, which may not align with the stochastic nature of URLLC traffic.

In recent years, cell-free MIMO has emerged as a promising wireless technology, where multiple O-RUs serve each user cooperatively. Compared with traditional cellular MIMO, cell-free MIMO networks achieve more uniform data rates across the coverage area due to macro diversity offered by distributed O-RUs. Precoding is a crucial step in the operation of cell-free networks where O-RUs steer their signal beams towards the intended users while minimizing interference to others. Some recent works have leveraged the advantages of cell-free networks to support eMBB and URLLC traffic. In [7], an optimization problem is formulated to determine the precoding matrices that maximize the long-term energy efficiency while satisfying URLLC queuing delay constraints. In [8], an algorithm is proposed to determine the PRB allocation and

precoding matrices that maximize eMBB throughput while satisfying URLLC constraints. The aforementioned works use a centralized approach and may not scale well with the number of users and antenna elements.

In this paper, we propose a distributed framework to support eMBB and URLLC traffic in a cell-free O-RAN. Both the O-RUs and user devices are equipped with multiple antennas. A PRB can be utilized by multiple users simultaneously via spatial multiplexing. We formulate an optimization problem to maximize the aggregate eMBB throughput subject to the URLLC constraints. We propose a multi-agent DRL algorithm to solve the formulated problem. The contributions of this paper are summarized as follows:

- We propose a distributed precoding algorithm using multi-agent actor-critic DRL to support eMBB and URLLC traffic in cell-free O-RAN. Each O-RU has a DRL agent, where the actor determines local precoding matrices and the critic evaluates the policy. The critics have global knowledge of the policies of all agents. The proposed approach enables collaboration between agents.
- Unlike prior approaches, we make no assumptions about the incoming URLLC traffic. We consider short packet transmission to satisfy the latency requirements of URLLC users, where each time slot is divided into short transmission time intervals (sTTIs). At the beginning of each sTTI, we monitor the URLLC buffer and use the multi-agent DRL algorithm to determine the precoding matrices. Given the short duration of sTTIs, realtime decision-making is crucial. To achieve this, actors are deployed as dApps at the O-DU, allowing them to determine the precoding matrices in under 1 ms. Meanwhile, critics are deployed as xApps within the near-RT RIC.
- We compare the performance of the proposed algorithm with another multi-agent actor-critic DRL algorithm as well as two additional DRL baselines across various numbers of eMBB users and different URLLC traffic demands. Simulation results show that our proposed algorithm outperforms other baseline schemes in terms of aggregate eMBB throughput by up to 55.4%.

The rest of this paper is organized as follows. In Section II, we introduce the system model and formulate the optimization problem. In Section III, we propose a multi-agent DRL algorithm to solve the formulated problem. Performance evaluation is presented in Section IV. Conclusion is given in Section V.

Notations: In this paper, we use $\mathbb{C}$ to denote the set of complex numbers. We use boldface upper-case letters (e.g., $\mathbf{X}$) to denote matrices and boldface lower-case letters (e.g., $\mathbf{x}$) to denote vectors. $\mathbf{I}_N$ represents an $N \times N$ identity matrix. $(.)^H$ denotes the conjugate transpose of a matrix. $\mathrm{tr}(.)$ and $\det(.)$ denote the trace and determinant of a matrix, respectively. $\dim(.)$ denotes the dimension of a vector space.

II. System Model and Problem Formulation

We consider downlink transmission in a cell-free O-RAN as shown in Fig. 1. The system supports eMBB and URLLC users. In cell-free O-RAN, each user can be served by multiple

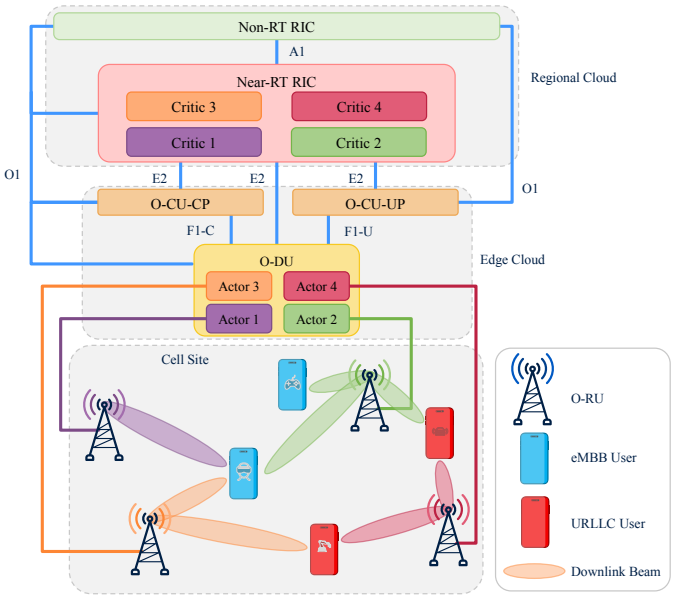

Fig. 1. The considered system model. The actor module for each O-RU is deployed as a dApp in the O-DU. The critic module is deployed as an xApp in the near-RT RIC.

O-RUs. The set of O-RUs is denoted by $\mathcal{L} = \{1, 2, \ldots, L\}$. Each O-RU is equipped with N_t antennas and each user equipment has N_r antennas. The O-RUs are connected to an O-DU using open-fronthaul links. The near-RT RIC uses the E2 termination to collect key performance measurements (KPMs) from E2 nodes and push control actions to them [1]. The network has K users. Let $\mathcal{K} = \{1, 2, \ldots, K\}$ denote the set of users. Let $\mathcal{K}_\mathrm{e}$ and $\mathcal{K}_\mathrm{u}$ denote the set of eMBB users and URLLC users, respectively. We have $\mathcal{K} = \mathcal{K}_\mathrm{e} \cup \mathcal{K}_\mathrm{u}$ and $\mathcal{K}_\mathrm{e} \cap \mathcal{K}_\mathrm{u} = \emptyset$. Let $K_\mathrm{e} = |\mathcal{K}_\mathrm{e}|$ and $K_\mathrm{u} = |\mathcal{K}_\mathrm{u}|$ denote the number of eMBB users and URLLC users, respectively. The resource pool consists of W PRBs, denoted by set $\mathcal{W} = \{1, 2, \ldots, W\}$. Each PRB spans a time slot of duration T seconds and a bandwidth of B Hz. Each time slot is further divided into η sTTIs for short packet transmissions. In each sTTI, q_k packets of size ζ need to be transmitted to URLLC user $k \in \mathcal{K}_\mathrm{u}$ to empty the URLLC buffer. We assume that the O-RUs have perfect channel state information (CSI).

Let $\mathbf{H}_{k,l,\omega} \in \mathbb{C}^{N_\mathrm{r} \times N_\mathrm{t}}$ denote the downlink channel gain between user $k \in \mathcal{K}$ and O-RU $l \in \mathcal{L}$ over PRB $\omega \in \mathcal{W}$. Let $\beta_{k,l}$ denote the large-scale fading coefficient from O-RU l to user k. In cell-free MIMO, each user is served by a subset of O-RUs to enhance scalability. The serving O-RUs are determined on a user-centric basis [9]. Let $c_{k,l}$ denote the clustering variable, where $c_{k,l} = 1$ if O-RU l is serving user k and is equal to 0 otherwise. Let $\mathcal{K}_l = \{k \in \mathcal{K} \mid c_{k,l} = 1\}$ denote the set of users served by O-RU l and $\mathcal{L}_k = \{l \in \mathcal{L} \mid c_{k,l} = 1\}$ denote the set of O-RUs that serve user k. In this paper, we use a heuristic to determine the serving clusters. For each user k, we sort the large-scale fading coefficients $\beta_{k,l}$ and choose the O-RUs with the highest $\beta_{k,l}$ until $\sum_{l \in \mathcal{L}} c_{k,l} \beta_{k,l} \geq \delta \sum_{l \in \mathcal{L}} \beta_{k,l}$ is satisfied, where δ is a threshold in the range $0 < \delta < 1$. By setting δ close to 1, this heuristic ensures that each user is served by

those O-RUs with high channel gain.

The precoding matrix for O-RU l serving user k over PRB ω is denoted by $\mathbf{V}_{k,l,\omega} \in \mathbb{C}^{N_\mathrm{t} \times \Upsilon}$, where $\Upsilon = \min(N_\mathrm{t}, N_\mathrm{r})$ is the number of data streams. The achievable data rate of eMBB user $k \in \mathcal{K}_\mathrm{e}$ at sTTI t is

$$r_k(t) = \sum_{\omega \in \mathcal{W}} B \log_2 \det \left(\mathbf{\Gamma}_{k,\omega} \right), \tag{1}$$

where $\mathbf{\Gamma}_{k,\omega} \in \mathbb{C}^{N_\mathrm{r} \times N_\mathrm{r}}$ is given by [10]

$$\mathbf{\Gamma}_{k,\omega} = \mathbf{I}_{N_\mathrm{r}} + \mathbf{\Psi}_{k,k,\omega} \left(\sum_{i \in \mathcal{K} \setminus \{k\}} \mathbf{\Psi}_{k,i,\omega} + \sigma^2 \mathbf{I}_{N_\mathrm{r}} \right)^{-1}. \tag{2}$$

In (2), σ^2 is the variance of the additive white Gaussian noise. The matrix $\mathbf{\Psi}_{k,i,\omega} \in \mathbb{C}^{N_\mathrm{r} \times N_\mathrm{r}}$ is given by

$$\mathbf{\Psi}_{k,i,\omega} = \left(\sum_{l \in \mathcal{L}_i} \mathbf{H}_{k,l,\omega} \mathbf{V}_{i,l,\omega} \right) \left(\sum_{l \in \mathcal{L}_i} \mathbf{V}_{i,l,\omega}^H \mathbf{H}_{k,l,\omega}^H \right). \tag{3}$$

We consider short packet transmissions for URLLC users to achieve low latency. The achievable data rate of URLLC user $k \in \mathcal{K}_\mathrm{u}$ can be determined in the finite blocklength regime as

$$r_k(t) = \sum_{\omega \in \mathcal{W}} B \left(\log_2 \det \left(\mathbf{\Gamma}_{k,\omega} \right) - \log_2 e \, Q^{-1}(\epsilon) \sqrt{\frac{D_{k,\omega}}{L}} \right), \tag{4}$$

where L is the codeword blocklength, $Q^{-1}(.)$ is the inverse of the Gaussian Q-function, and ϵ is the transmission error probability. $D_{k,\omega}$ represents the channel dispersion of URLLC user $k \in \mathcal{K}_\mathrm{u}$ in PRB $\omega \in \mathcal{W}$ and is given by $D_{k,\omega} = \mathrm{tr} \left[\mathbf{I}_{N_\mathrm{r}} - \mathbf{\Gamma}_{k,\omega}^{-2} \right]$ [11].

We aim to maximize the aggregate eMBB throughput while satisfying the latency requirements of URLLC users. The precoding optimization problem can be formulated as

$$\underset{\substack{\mathbf{V}_{k,l,\omega}, k \in \mathcal{K}, \\ l \in \mathcal{L}, \omega \in \mathcal{W}}}{\text{maximize}} \quad \sum_{k \in \mathcal{K}_\mathrm{e}} r_k(t) \tag{5a}$$

$$\text{subject to} \quad r_k(t) \geq \frac{q_k(t) \zeta \eta}{T}, \; k \in \mathcal{K}_\mathrm{u} \tag{5b}$$

$$\sum_{k \in \mathcal{K}_l} \sum_{\omega \in \mathcal{W}} \mathrm{tr} \left(\mathbf{V}_{k,l,\omega} \mathbf{V}_{k,l,\omega}^H \right) \leq P^{\max}, \; l \in \mathcal{L}. \tag{5c}$$

At each sTTI t, the above optimization problem needs to be solved to determine the precoding matrices for eMBB and URLLC users. Constraint (5b) guarantees the minimum data rate requirements of URLLC users. Constraint (5c) limits the total transmit power allocated to users by each O-RU to $P^{\max}$. Problem (5) is nonconvex due to the nonconvexity of the objective function (5a) and constraint (5b). In the next section, we propose a multi-agent DRL algorithm to solve the problem in a distributed manner.

III. Proposed Algorithm

In a conventional O-RAN, a DRL algorithm resides in the near-RT RIC as an xApp and performs control loops in the near-realtime time scale. However, the precoding matrices need to be determined within a realtime time scale (less than 1 ms) to satisfy the latency requirements of URLLC users. A centralized DRL algorithm may also have convergence issues due to its large state and action spaces. Moreover, deploying a centralized xApp in the near-RT RIC to determine the precoding matrices for all O-RUs may not be scalable. To resolve these issues, we reformulate problem (5) as a Markov game, where a DRL agent is assigned to each O-RU $l \in \mathcal{L}$ to locally determine the precoding matrices for users $k \in \mathcal{K}_l$. A Markov game is a generalization of the Markov decision process (MDP) to multiple agents. It models decision-making in environments where multiple agents interact and influence the state of the environment through their actions.

We define the Markov game as a set of state spaces $\mathcal{S} = \{\mathcal{S}_1, \ldots, \mathcal{S}_L\}$, a set of action spaces $\mathcal{A} = \{\mathcal{A}_1, \ldots, \mathcal{A}_L\}$, a set of policies $\pi = \{\pi_1, \ldots, \pi_L\}$, and a set of reward functions $R = \{R_1, \ldots, R_L\}$. Each episode of the Markov game corresponds to a time slot in cell-free O-RAN, and each step t within that episode corresponds to an sTTI. At each step t, each agent $l \in \mathcal{L}$ uses its policy $\pi_l : \mathcal{S}_l \mapsto \mathcal{P}(\mathcal{A}_l)$ to map its observation $s_l(t) \in \mathcal{S}_l$ to a distribution over its action space. Based on the current state and the sampled action, agent l receives a reward $R_l(t) : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{R}$ and the state transits to $s_l(t+1)$. The reward function for agent l should consider the eMBB throughput and URLLC requirements of users $k \in \mathcal{K}_l$. Thus, we define the reward function for agent l as

$$R_l(t) = \sum_{k \in \mathcal{K}_\mathrm{e} \cap \mathcal{K}_l} r_k(t) - \phi_l(t) \sum_{k \in \mathcal{K}_\mathrm{u} \cap \mathcal{K}_l} \left(\frac{q_k(t) \zeta \eta}{T} - r_k(t) \right), \tag{6}$$

where the first term represents the aggregate eMBB throughput and the second term penalizes the reward if the URLLC rate constraint is violated. $\phi_l(t)$ is the penalty coefficient at step t. It is updated as follows [3]:

$$\phi_l(t+1) = \max \left\{ \phi_l(t) + \sum_{k \in \mathcal{K}_\mathrm{u} \cap \mathcal{K}_l} \left(\frac{q_k(t) \zeta \eta}{T} - r_k(t) \right), 0 \right\}. \tag{7}$$

At the beginning of each episode, we initialize ϕ_l to zero. According to (7), if the URLLC constraint is violated at a step of the episode, ϕ_l is increased to place more emphasis on the URLLC term in (6) at the next step of the episode. Conversely, if the URLLC constraint is consistently satisfied across all steps of the episode, $\phi_l(t)$ remains at zero. This helps the algorithm adapt to dynamic environments. For example, if the URLLC traffic demand suddenly increases, $\phi_l(t)$ will increase as well. We include $\phi_l(t)$ in the state space of agent l. Thus, the policy network will notice the change and increase the data rate of URLLC users. The state $s_l(t) \in \mathcal{S}_l$ of agent l at step t consists of the channel gains for both eMBB and URLLC users, along with the number of URLLC packets and the URLLC penalty coefficient. It is defined as $s_l(t) = \{\mathbf{H}_{k,l,\omega}(t), q_k(t), \phi_l(t), \; k \in \mathcal{K}_l, \omega \in \mathcal{W}\}$. The action $a_l(t) \in \mathcal{A}_l$ taken by agent l at step t is defined as $a_l(t) = \{\mathbf{V}_{k,l,\omega}(t), \; k \in \mathcal{K}_l, \omega \in \mathcal{W}\}$.

At each step t, each agent l aims to maximize its own total discounted reward $G_l(t) = \sum_{i=t}^{\eta-1} \gamma^{i-t} R_l(t)$, where γ

is a discount factor. However, according to (1) and (4), the achievable data rate $r_k(t)$ in the reward function $R_l(t)$ of agent l is determined by the precoding matrices of all O-RUs. In other words, the reward of each agent depends on the actions of all agents. Independent decision making in such an environment may result in convergence issues. Thus, there should be some degree of collaboration between agents. The state-action value function Q_l for agent l at step t is defined as follows:

$$Q_l(\mathbf{s}(t), \mathbf{a}(t)) = \mathbb{E}_\pi \left[G_l(t) \mid \mathbf{s}(t), \mathbf{a}(t) \right], \tag{8}$$

where $\mathbf{s}(t) = (s_1(t), \ldots, s_L(t))$ and $\mathbf{a}(t) = (a_1(t), \ldots, a_L(t))$ denote the states and actions of all agents at step t, respectively. Using the Bellman equations [12], Q_l can be recursively written as

$$Q_l(\mathbf{s}(t), \mathbf{a}(t)) = R_l(t) + \gamma \mathbb{E}_{\mathbf{s}} \big[\mathbb{E}_{\mathbf{a} \sim \pi(\cdot|\mathbf{s})} [Q_l(\mathbf{s}(t+1), \mathbf{a}(t+1))] \big]. \tag{9}$$

In this paper, we propose a multi-agent extension of the soft actor-critic (SAC) algorithm introduced in [12] to solve the Markov game. SAC is an off-policy actor-critic algorithm that maximizes a combination of the reward and policy entropy to encourage exploration. The overall objective function of agent l with policy π_l is

$$J(\pi_l) = \sum_{t=0}^{\eta-1} \mathbb{E}_{(s_l, a_l) \sim \pi_l} \left[\gamma^t \left(R_l(t) + \alpha_l \mathcal{H}(\pi_l(\cdot|s_l(t))) \right) \right], \tag{10}$$

where $\mathcal{H}(\pi_l(\cdot|s_l(t))) = -\mathbb{E}_{a_l \sim \pi_l(\cdot|s_l)}[\log \pi_l(a_l(t)|s_l(t))]$ is the entropy of the policy and α_l is the temperature parameter that controls the trade-off between the reward and entropy. The entropy term encourages exploration by keeping the policy's behavior stochastic. It prevents premature convergence to suboptimal policies. π_l is referred to as the actor for agent l. It is approximated using a deep neural network (DNN) with parameters θ_l^π deployed as a dApp at the O-DU. The actor objective function of agent l at step t is defined as

$$J_{\pi_l}(\theta_l^\pi) = \mathbb{E}_{s_l \sim \mathcal{B}} \big[\mathbb{E}_{a_l \sim \pi_l(\cdot|s_l)} [\alpha_l \log \pi_l(a_l(t)|s_l(t)) - \min_{i=1,2} Q_{l,i}(\mathbf{s}(t), \mathbf{a}(t))] \big], \tag{11}$$

where $\mathcal{B}$ is the experience replay buffer that contains the tuples $(\mathbf{s}(t), \mathbf{a}(t), \mathbf{R}(t), \mathbf{s}(t+1))$, recording the experiences of all agents throughout training. $\mathbf{R}(t) = (R_1(t), \ldots, R_L(t))$ denotes the rewards of all agents at step t. To mitigate overestimation bias, SAC trains two independent Q-functions $Q_{l,1}$ and $Q_{l,2}$ using DNNs with parameters $\theta_{l,1}^Q$ and $\theta_{l,2}^Q$, respectively, and uses their minimum value as the Q-value estimate. This is called the critic for agent l, deployed as an xApp in the near-RT RIC. It takes the collective states and actions of all agents as input and outputs the Q-value for agent l. The critic loss functions of agent l at step t are defined as

$$J_{Q_{l,i}}(\theta_{l,i}^Q) = \mathbb{E}_{(\mathbf{s}, \mathbf{a}, \mathbf{R}) \sim \mathcal{B}} \left[\frac{1}{2} \left(Q_{l,i}(\mathbf{s}(t), \mathbf{a}(t)) - y_l(t) \right)^2 \right], \quad i \in \{1, 2\}, \tag{12}$$

Algorithm 1: Training procedure for the distributed precoding algorithm in cell-free O-RAN

```
Initialize parameters θ_l^π, θ_{l,1}^Q, θ_{l,2}^Q, θ̂_{l,1}^Q, and θ̂_{l,2}^Q
for iteration := 1 to M_iter do
    Set m_frames := 0
    while m_frames ≤ M_frames do
        Observe initial state s(0)
        for t := 0 to η − 1 do
            For each agent l, sample action a_l(t) ~ π_l(·|s_l(t))
            Execute actions a(t) and observe reward R(t) and new states s(t+1)
            Store (s(t), a(t), R(t), s(t+1)) in B
            m_frames := m_frames + 1
    for optimizer_step := 1 to M_opt do
        Sample a batch of M_batch samples from B
        For each agent l, update the critic by minimizing the loss in (12)
        For each agent l, update the actor by minimizing the loss in (11)
        For each agent l, update the temperature parameter by minimizing the loss in (15)
        For each agent l, update the target network parameters using (14)
```

$$y_l(t) = R_l(t) + \gamma \mathbb{E}_{a_l \sim \pi_l(\cdot|s_l)} \Big[\min_{i=1,2} \hat{Q}_{l,i}(\mathbf{s}(t+1), \mathbf{a}(t+1)) - \alpha_l \log \pi_l(a_l(t+1)|s_l(t+1)) \Big], \tag{13}$$

where $\hat{Q}_{l,1}$ and $\hat{Q}_{l,2}$ are the target Q-functions of agent l parameterized by $\hat{\theta}_{l,1}^Q$ and $\hat{\theta}_{l,2}^Q$, respectively. The target network parameters are updated gradually using soft updates by:

$$\hat{\theta}_{l,i}^Q(t) = \tau \theta_{l,i}^Q(t) + (1 - \tau) \hat{\theta}_{l,i}^Q(t-1), \quad i \in \{1, 2\}, \tag{14}$$

where τ is the soft update rate. The temperature parameter α_l in (13) is dynamically adjusted throughout training to keep the entropy close to a target entropy $\mathcal{H}_{\text{target}}$. The temperature loss is defined as

$$J_{\alpha_l}(\alpha_l) = \mathbb{E}_{s_l \sim \mathcal{B}} \big[\mathbb{E}_{a_l \sim \pi_l(\cdot|s_l)} [-\alpha_l \log \pi_l(a_l(t)|s_l(t)) - \alpha_l \mathcal{H}_{\text{target}}] \big]. \tag{15}$$

A common choice for the target entropy is $\mathcal{H}_{\text{target}} = -\prod_{l \in \mathcal{L}} \dim(\mathcal{A}_l)$. During training, the critic evaluates the actor's actions by estimating the Q-value, while the actor updates its policy by minimizing the loss function in (11). When the training is complete, the actors use their local CSI to determine the precoding matrices. The precoding matrices determined by the actor of agent l may violate the maximum transmit power constraint (5c). To enforce this constraint, agent l uses the following scaling method to map the solutions back into the feasible constraint set [9, Section 7.1.2]:

$$\mathbf{V}'_{k,l,\omega} = \sqrt{\min \left\{ \frac{P^{\max}}{P_l^{\text{current}}}, 1 \right\}} \mathbf{V}_{k,l,\omega}, \; k \in \mathcal{K}_l, \omega \in \mathcal{W}, \tag{16}$$

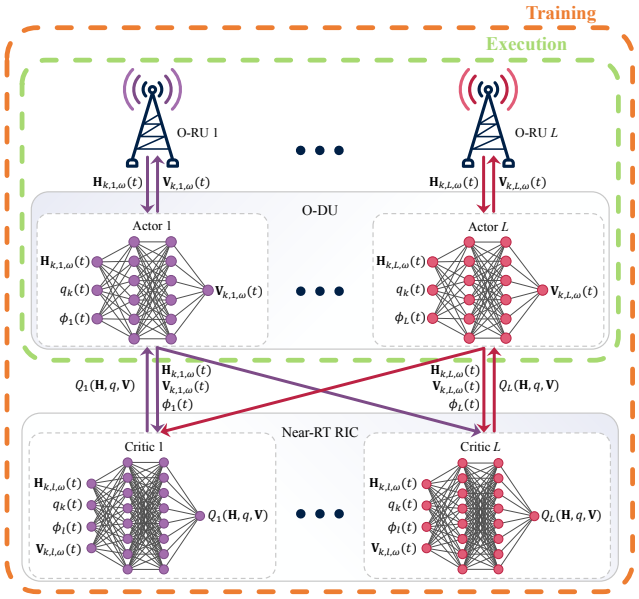

Fig. 2. Workflow of the multi-agent actor-critic DRL algorithm in cell-free O-RAN during training and execution.

$$P_l^{\text{current}} = \sum_{k\in\mathcal{K}_l}\sum_{\omega\in\mathcal{W}} \text{tr}\left(\mathbf{V}_{k,l,\omega}\mathbf{V}_{k,l,\omega}^H\right), \tag{17}$$

where the precoding matrices are only scaled if the current total power P_l^{current} exceeds $P^{\max}$. The training procedure for the proposed algorithm is summarized in Algorithm 1. Fig. 2 shows the actor-critic interactions in the proposed algorithm.

In the proposed algorithm, the centralized critic takes the global state-action pair as input. Thus, the value network has an input size of $\mathcal{O}\left(N_{\text{t}}N_{\text{r}}\sum_{l\in\mathcal{L}}|\mathcal{K}_l|\right)$, which grows substantially with the number of users and O-RUs. As a result, training the centralized critic can become a bottleneck in dense deployments. A possible workaround is to divide O-RUs into groups, where each group's critic is only aware of the policies of agents within that group. We leave the exploration of such solutions for future work.

IV. Performance Evaluation

In this section, we present the simulation results to evaluate the performance of the proposed algorithm. We consider a cell-free O-RAN consisting of $L = 16$ O-RUs placed in a 4×4 grid with a total area of 600 m^2. Each O-RU is positioned 10 m above the ground. The maximum transmit power of each O-RU $P^{\max}$ and the noise power σ^2 are set to 30 dBm and -114 dBm, respectively. We use a wrap-around topology to mimic a large network deployment. The O-RUs collaboratively serve $K_{\text{e}} = 5$ eMBB users and $K_{\text{u}} = 5$ URLLC users. For URLLC users, we set $\epsilon = 10^{-6}$ and $\zeta = 100$ bytes and assume that q_k follows a Poisson process with arrival rate $\lambda = 5$ packets per sTTI. The number of antennas of each O-RU, N_{t}, is equal to 4. The number of antennas of each user device, N_{r}, is equal to 2. We use the uncorrelated Rayleigh fading channel model, where $\mathbf{H}_{k,l,\omega} = \beta_{k,l}^{1/2}\mathbf{G}_{k,l,\omega}$. The small-scale fading coefficients $\mathbf{G}_{k,l,\omega}$ for each user k, O-RU l and PRB ω are generated from the complex Gaussian distribution $\mathcal{CN}\left(0, \mathbf{I}_{N_{\text{t}}N_{\text{r}}}\right)$. The large-scale fading coefficients are generated according to [9]. We consider $W = 25$ PRBs, and the bandwidth of each PRB $B = 360$ kHz. We set $T = 1$ ms and $\eta = 8$. The threshold δ is set to 0.9.

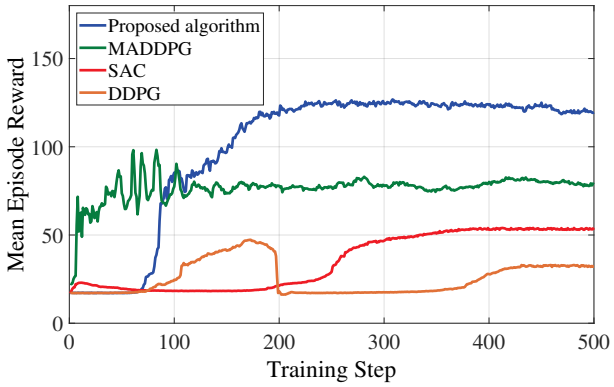

Fig. 3. Mean episode reward vs. the training step.

We use the BenchMARL library introduced in [13] to implement the proposed algorithm in PyTorch. The discount factor γ is 0.9. The learning rate is 5×10^{-5} and the epsilon parameter in the Adam optimizer is 10^{-6}. For Algorithm 1, we set $M_{\text{iter}} = 24000$, $M_{\text{frames}} = 6000$, $M_{\text{opt}} = 100$, and $M_{\text{batch}} = 1024$. We use soft updates with $\tau = 0.005$. The temperature parameter is initialized as $\alpha_l = 1$.

For performance comparison, we consider the following DRL algorithms as baselines:

1) **SAC [12]:** In conventional SAC, each agent operates independently and does not coordinate with other agents. The critic is unaware of the policies of other agents.
2) **Deep deterministic policy gradient (DDPG) [14]:** DDPG is an off-policy actor-critic algorithm that aims to maximize the expected reward. It adds a random noise to the actions during training to encourage exploration.
3) **Multi-agent DDPG (MADDPG) [15]:** MADDPG is a multi-agent extension of DDPG, where the critic for each agent is aware of the policies of all agents.

In Fig. 3, we compare the mean episode reward between our proposed algorithm and the DRL baselines throughout training. Our proposed algorithm outperforms MADDPG since it uses a stochastic policy and maximizes the entropy, which enhances exploration and reduces the chances of converging to suboptimal policies. Meanwhile, MADDPG outperforms DDPG and SAC but still converges to a local optimum due to limited exploration; adding random noise to actions may not be effective in high-dimensional environments with multiple local optima. Additionally, the Q-function in MADDPG tends to suffer from overestimation bias. SAC and DDPG demonstrate the worst performance. This is because in those two algorithms, each agent independently interacts with the environment. Since all agents update their policies simultaneously without cooperation, the environment appears to be non-stationary to each agent, which violates the Markov assumptions required for convergence.

In Fig. 4, we compare the aggregate eMBB throughput

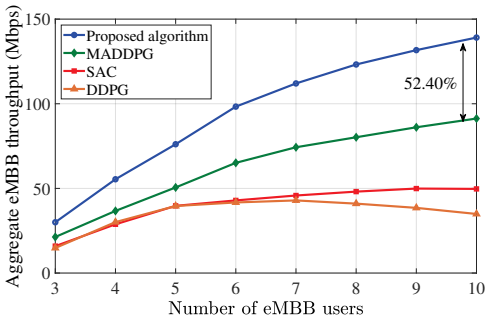

Fig. 4. Aggregate throughput of eMBB users vs. the number of eMBB users.

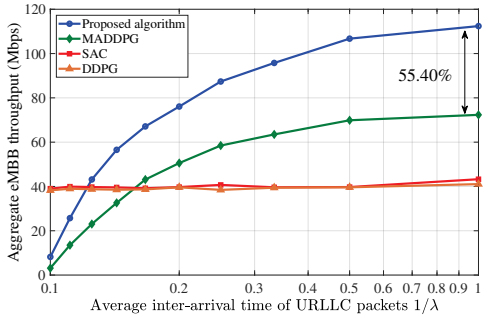

Fig. 5. Aggregate throughput of eMBB users vs. the average inter-arrival time of URLLC packets $1/\lambda$.

between our proposed algorithm and the DRL baselines under different number of eMBB users. We observe that our proposed algorithm achieves up to 52.4% higher aggregate eMBB throughput than MADDPG due to its enhanced exploration. MADDPG outperforms DDPG and SAC due to the collaboration between agents throughout training. The aggregate eMBB throughput for DDPG degrades after a certain point as the number of eMBB users increases. This is because DDPG fails to learn an effective precoding method to mitigate inter-user interference. As the number of users increases, the interference becomes more significant, eventually reaching a point where adding a new user degrades the overall system performance.

Fig. 5 shows the aggregate eMBB throughput as a function of the average inter-arrival time of URLLC packets $1/\lambda$ for the proposed algorithm and DRL baselines. As the average inter-arrival time $1/\lambda$ increases, URLLC packets arrive less frequently. Therefore, the aggregate eMBB throughput increases since less transmission power is allocated to URLLC users to meet their latency constraint. The proposed algorithm consistently outperforms MADDPG under different URLLC traffic demands. When $1/\lambda$ is equal to 1 sTTI (i.e., under light URLLC traffic), the proposed algorithm achieves 55.4% higher aggregate eMBB throughput than MADDPG.

V. Conclusion

In this paper, we proposed a distributed precoding algorithm to support eMBB and URLLC traffic in cell-free O-RAN. We formulated the precoding optimization problem to maximize the aggregate eMBB throughput while meeting URLLC latency requirements and O-RU power budget. To solve the formulated problem, we proposed a multi-agent DRL algorithm. Each O-RU is assigned an actor-critic DRL agent. The actor, located at the O-DU, determines the O-RU's precoding matrices according to its local CSI. The centralized critic, implemented as an xApp in the near-RT RIC, is aware of the states and actions of all agents and evaluates the actor's policies during training. Simulation results showed that the proposed algorithm achieves up to 55.4% higher aggregate eMBB throughput compared with three baseline schemes. For future work, we plan to deploy our proposed framework on a real-world testbed and consider the impact of channel estimation errors on its performance.

References

[1] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Understanding O-RAN: Architecture, interfaces, algorithms, security, and research challenges," *IEEE Commun. Surveys & Tuts.*, vol. 25, no. 2, pp. 1376–1411, second quarter 2023.

[2] A. Lacava, L. Bonati, N. Mohamadi, R. Gangula, F. Kaltenberger, P. Johari, S. D'Oro, F. Cuomo, M. Polese, and T. Melodia, "dApps: Enabling real-time AI-based open RAN control," *arXiv preprint arXiv:2501.16502*, pp. 1–32, Jan. 2025.

[3] M. Alsenwi, E. Lagunas, and S. Chatzinotas, "Coexistence of eMBB and URLLC in open radio access networks: A distributed learning framework," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Rio de Janeiro, Brazil, Dec. 2022.

[4] O. Adamuz-Hinojosa, L. Zanzi, V. Sciancalepore, A. Garcia-Saavedra, and X. Costa-Pérez, "ORANUS: Latency-tailored orchestration via stochastic network calculus in 6G O-RAN," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Vancouver, Canada, May 2024.

[5] A. Anand, G. De Veciana, and S. Shakkottai, "Joint scheduling of URLLC and eMBB traffic in 5G wireless networks," *IEEE/ACM Trans. Netw.*, vol. 28, no. 2, pp. 477–490, Apr. 2020.

[6] M. Setayesh, S. Bahrami, and V. W.S. Wong, "Resource slicing for eMBB and URLLC services in radio access network using hierarchical deep learning," *IEEE Trans. Wireless Commun.*, vol. 21, no. 11, pp. 8950–8966, Nov. 2022.

[7] Y. Yin, B. Liu, P. Zhu, X. Lyu, and Y. Wang, "Joint long-term energy efficient scheduling and beamforming design for URLLC in cell-free MIMO systems," *IEEE Wireless Commun. Letters*, vol. 13, no. 1, pp. 118–122, Jan. 2024.

[8] Z. Wang and V. W.S. Wong, "Joint resource block allocation and beamforming with mixed-numerology for eMBB and URLLC use cases," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Madrid, Spain, Dec. 2021.

[9] Ö. T. Demir, E. Björnson, and L. Sanguinetti, "Foundations of user-centric cell-free massive MIMO," *Found. Trends Signal Process.*, vol. 14, no. 3-4, pp. 162–472, 2021.

[10] A. Mehrabian and V. W.S. Wong, "Joint spectrum, precoding, and phase shifts design for RIS-aided multiuser MIMO THz systems," *IEEE Trans. Commun.*, vol. 72, no. 8, pp. 5087–5101, Aug. 2024.

[11] X. You, B. Sheng, Y. Huang, W. Xu, C. Zhang, D. Wang, P. Zhu, and C. Ji, "Closed-form approximation for performance bound of finite blocklength massive MIMO transmission," *IEEE Trans. Commun.*, vol. 71, no. 12, pp. 6939–6951, Dec. 2023.

[12] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Stockholm, Sweden, Jul. 2018.

[13] M. Bettini, A. Prorok, and V. Moens, "BenchMARL: Benchmarking multi-agent reinforcement learning," *J. Mach. Learn. Res.*, vol. 25, no. 217, pp. 1–10, Jul. 2024.

[14] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, San Juan, PR, May 2016.

[15] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, Long Beach, CA, Dec. 2017.