Cooperative Edge Caching via Multi Agent Reinforcement Learning in Fog Radio Access Networks

Qi Chang[1], Yanxiang Jiang[1,2,*], Fu-Chun Zheng[1,2], Mehdi Bennis[3], and Xiaohu You[1]

[1]National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China

[2]School of Electronic and Information Engineering, Harbin Institute of Technology, Shenzhen 518055, China

[3]Centre for Wireless Communications, University of Oulu, Oulu 90014, Finland

E-mail: {220200705@seu.edu.cn, yxjiang@seu.edu.cn, fzheng@ieee.org, mehdi.bennis@oulu.fi, xhyu@seu.edu.cn}

Abstract—In this paper, the cooperative edge caching problem in fog radio access networks (F-RANs) is investigated. To minimize the content transmission delay, we formulate the cooperative caching optimization problem to find the globally optimal caching strategy.By considering the non-deterministic polynomial hard (NP-hard) property of this problem, a Multi Agent Reinforcement Learning (MARL)-based cooperative caching scheme is proposed.Our proposed scheme applies double deep Q-network (DDQN) in every fog access point (F-AP), and introduces the communication process in multi-agent system. Every F-AP records the historical caching strategies of its associated F-APs as the observations of communication procedure.By exchanging the observations, F-APs can leverage the cooperation and make the globally optimal caching strategy.Simulation results show that the proposed MARL-based cooperative caching scheme has remarkable performance compared with the benchmark schemes in minimizing the content transmission delay.

Index Terms—Fog radio access networks, cooperative edge caching, multi agent reinforcement learning, double deep Q-network.

I. INTRODUCTION

With the rapid advancement of wireless network technologies and the tremendous amount of data information, the global mobile data traffic generated by portable devices grows continuously in these years. Fog radio access network (F-RAN) has been proposed as a promising paradigm for improving spectral efficiency and optimizing legacy networks for mobile cellular communications systems [1] [2]. In F-RANs, edge caching can be regarded as a key component to relax the traffic burden at backhaul links by edge devices, e.g., fog access points (F-APs) [3]. Due to the finite cache capacity and communications resources of F-APs, the caching strategy should be designed comprehensively. In this regard, cooperative edge caching has become an efficient way to alleviate data traffic and decrease transmission delay.

There is a variety of research works focused on cooperative edge caching. In [4], an improved pigeon inspired optimization based cooperative edge caching scheme was proposed, which utilized Cauchy perturbation and self-adaptive factor to avoid premature convergence and achieve a better search performance. In [5], the authors proposed a brain storm optimization approach which utilized the penalty-based fitness function in individuals evaluation to meet the storage capacity constraint and the genetic algorithm in new individuals generation to meet the integer constraint, respectively. Specifically, with the maturation of reinforcement learning (RL), extensive works take RL into the optimization of cooperative edge caching. In [6], the authors deployed a distributed Q-learning based content replacement strategy, which created a Q-table to store the Q-value of every action. In [7], a learning automata based Q-learning algorithm for cooperative caching was proposed, which was invoked to obtain an optimal action selection at a random and stationary environment. In [8], a delay-aware cache update policy was proposed in F-RANs with the dueling deep Q-network (DQN). In [9], the authors proposed a double DQN based distributed edge caching algorithm to find the optimal caching policy with content recommendation. In [10], the cooperative caching problem was formulated by two potential recurrent neural networks, i.e., the echo state network and long short-term memory network, to determine which content to cache and where to cache. By considering the leakage of sensitive users' data and additive waste of resources in training process, a cooperative caching method based on federated deep reinforcement learning framework was proposed to find the optimal caching policy in [11]. In [12], the authors extended Q-learning into multi-agent learning to solve the content transmission delay problem, which generally required complex computation for finding Nash-Q equilibrium. Most of the aforementioned methods utilize RL to find the optimal caching strategy. However, these RL-based methods generally neglect the influence of environment by other agents when a particular agent learns from the environment independently.

According to the above discussions, a cooperative edge caching scheme based on Multi Agent Reinforcement Learning (MARL) is proposed in F-RANs to find the globally optimal caching strategy. Firstly, the cooperative edge caching optimization problem is formulated to minimize the average transmission delay under the cache capacity and integer constraints. Then, double deep Q-network (DDQN) is utilized by each F-AP to learn how to coordinate their caching strategies in the multi-agent system. Finally, every F-AP keeps its his-
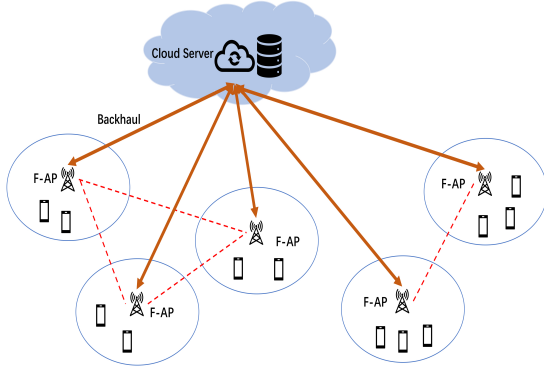
Fig. 1: The cooperative caching scenario in F-RANs.

torical caching strategy as the observation of communication procedure. Through the iterative communications among F-APs, the average transmission delay can be reduced and the optimization problem is tackled dynamically.

The rest of this paper is organized as follows. Section II introduces our system model and problem formulation. Section III describes the proposed MARL-based cooperative caching scheme. The simulation results are shown in Section IV. Finally, conclusions are drawn in Section V.

II. SYSTEM MODEL AND PROBLEM FORMULATION

A. System Model

The cooperative caching scenario in F-RANs is illustrated in Fig. 1, where a cloud server is connected with multiple F-APs via backhaul links and multiple users are under the serving region of each F-AP. The continuous time is divided into discrete time slots $\mathcal{T} = \{1, 2, ..., t, ..., T\}$. The set of F-APs is denoted by $\mathcal{N} = \{1, 2, ..., n, ..., N\}$ and the set of all the considered users is denoted by $\mathcal{U} = \{1, 2, ..., u, ..., U\}$. The set of users in the serving region of F-AP n is denoted by $\mathcal{U}_n^t = \{1, 2, ..., u_n, ..., U_n\}$. We assume that user u_n is only served by F-AP n during time slot t.

Suppose that the library, denoted by $\mathcal{F} = \{1, 2, ..., f, ..., F\}$, is located at the cloud server far away from users, which can be accessed by F-APs via backhaul links. Furthermore, we assume that every file has the same size Q. The content popularity distribution in the serving region of F-AP n is denoted by $\mathcal{P}_n^t = \left\{ P_{n,1}^t, P_{n,2}^t, ..., P_{n,f}^t, ..., P_{n,F}^t \right\}$. Let $p_{u,f}^t$ denote the file preference of user u for file f, which can be viewed as content popularity indicator and predicted via some learning procedure [13]. We assume that the user's file preference $p_{u,f}^t$ satisfies the Mandelbrot-Zipf distribution [14] as follows:

$$p_{u,f}^t = \frac{\phi_u^t(f)^{-\tau_t}}{\sum_{i=1}^{F} i^{-\tau_t}}, \forall u \in \mathcal{U}, \tag{1}$$

where $\phi_u^t(f) \in \oplus_u^t = \{\phi_u^t(1), \phi_u^t(2), ..., \phi_u^t(f), ..., \phi_u^t(F)\}$, $\oplus_u^t$ is a random permutation of content library $\mathcal{F}$ for user u during time slot t, and τ_t is the time-varying skewness factor.

The content popularity in F-AP n generally depends on the file preference of its serving users $u_n \in \mathcal{U}_n$, and it can be calculated by:

$$P_{n,f}^t = \mathbb{E}_u\left[\sum_{u \in \mathcal{U}_n} p_{u,f}^t\right], \tag{2}$$

where $\mathbb{E}[\cdot]$ denotes the operation of mathematical expectation. We also assume that all F-APs have the same cache capacity S. Let the binary variable $x_{n,f}$ indicate whether F-AP n has cached file f. $x_{n,f} = 1$ if file f has been cached at F-AP n, and otherwise $x_{n,f} = 0$. The caching variable $x_{n,f}$ should be determined collaboratively by all F-APs and the cooperative caching strategy, denoted by $\boldsymbol{X} = [x_{n,f}]_{N \times F}$, should be designed carefully to make file requests from all users respond quickly and accurately.

B. Transmission Mode

At the network edge, some F-APs can deliver the requested file via backhaul links [15]. The connectivity among F-APs can be denoted by an $N \times N$ matrix $\boldsymbol{Y}$, where every binary element $y_{n,m}$ indicates whether F-AP n can associate with F-AP m. $y_{n,m} = 1$ if F-AP n can establish connection with F-AP m, and otherwise $y_{n,m} = 0$. Therefore, the set of the associated F-APs for F-AP n can be denoted by $\mathcal{N}_n = \{m | \forall m \in \mathcal{N}, y_{n,m} = 1, m \neq n\}$.

When user u_n requests file f, the serving F-AP n checks its own caching strategy $[x_{n,1}, ..., x_{n,f}, ..., x_{n,F}]$ to decide how to transmit the requested file f to user u_n. Some specific transmission modes are applied to deliver the file for the requesting user. In the following, we discuss the transmission delay with different transmission modes, when the requested file is cached in the serving F-AP, its associated F-APs or the cloud server.

1) F-AP-to-User: If the requested file is cached in the serving F-AP, it can directly deliver the file to the requesting user. Let $R_{n,u,f}^t$ denote the delivery rate of file f from F-AP n to user u_n during time slot t. Assume that efficient interference management schemes are applied and interference power is constrained by a fixed value P_I [13]. Then, the file delivery rate in wireless transmission stage can be expressed as:

$$R_{n,u,f}^t = B \log \left(1 + |h_{n,u}^t|^2 l_{n,u}^t \frac{P_n}{N_0 B + P_I}\right), \tag{3}$$

where B is the channel bandwidth, P is the transmit power, N_0 is the power spectral density of noise, $h_{n,u}^t$ denotes the channel coefficient between F-AP n and user u during time slot t, and $l_{n,u}^t$ is the distance between F-AP n and user u during time slot t. Thus, the corresponding transmission delay can be defined as:

$$Z_{1,n,u,f}^t = Q / R_{n,u,f}^t. \tag{4}$$

2) F-AP-to-F-AP: If the requested file is not cached in the serving F-AP, the requesting user can obtain the requested file from the associated F-APs that have cached the file. And the transmission process can be divided into two parts: the transmission delay from F-AP to the requesting user,

i.e., $Z_{1,n,u,f}^t$ and the transmission delay between F-APs, i.e., $Z_{2,n,f}^t$. Then, we have:

$$Z_{2,n,f}^t = Q\left(\sum_{m\in\mathcal{N}_n} \frac{x_{m,f}}{R_{n,m,f}^t}\right), \tag{5}$$

where $R_{n,m,f}^t$ is the transmit rate between F-APs. When there exist multiple associated F-APs that have stored the requested file, these associated F-APs can transmit the requested file cooperatively to improve the transmission performance [5].

3) Cloud-Server-to-F-AP: If the requested file is cached neither in the serving F-AP nor in its associated F-APs, the requested file can only be fetched from the cloud server. And the transmission process can also be divided into two parts: the transmission delay from F-AP to the requesting user, i.e., $Z_{1,n,u,f}^t$ and the transmission delay from the cloud server to F-AP, i.e., $Z_{3,n,f}^t$. And the transmission delay in backhaul link can be defined as:

$$Z_{3,n,f}^t = C/R_{n,0,f}^t, \tag{6}$$

where $R_{n,0,f}^t$ is the transmit rate from the cloud server to F-AP.

Based on the above discussions, the transmission delay for the requested file f in three transmission modes can be expressed as:

$$
\begin{aligned}
d_{n,f}^t(\boldsymbol{X}) =\ & x_{n,f} Z_{1,n,u,f}^t \\
& + (1-x_{n,f})\left(1 - \prod_{m\in\mathcal{N}_n}(1-x_{m,f})\right)\left(Z_{1,n,u,f}^t + Z_{2,n,f}^t\right) \\
& + (1-x_{n,f})\prod_{m\in\mathcal{N}_n}(1-x_{m,f})\left(Z_{1,n,u,f}^t + Z_{3,n,f}^t\right).
\end{aligned}
\tag{7}
$$

Without loss of generality, $Z_{1,n,u,f}^t < Z_{2,n,f}^t \ll Z_{3,n,f}^t$ is assumed. If $x_{n,f}=1$, the requested file can be directly fetched from the serving F-AP. If $x_{n,f}=0$ and $\prod_{m\in\mathcal{N}_n}(1-x_{m,f})=0$, the requested file can be fetched from the associated F-APs. And if $x_{n,f}=0$ and $\prod_{m\in\mathcal{N}_n}(1-x_{m,f})=1$, the requested file can be fetched from the cloud server.

C. Problem Formulation

By considering time-varying channel state, diverse content preference of user and cooperation among F-APs, our work aims at finding the globally optimal caching strategy $\boldsymbol{X}^*$ to minimize the average transmission delay of the entire system. According to the transmission delay given by (7), the cooperative caching problem can be formulated as follows:

$$\min_{x_{n,f}} \quad \bar{D}(\boldsymbol{X}) = \frac{1}{T}\sum_{t=1}^T\sum_{f=1}^F\sum_{n=1}^N P_{n,f}^t \cdot d_{n,f}^t(\boldsymbol{X}) \tag{8}$$

$$\text{s.t.} \quad \begin{cases} \sum_{f=1}^F x_{n,f} \le S, \forall n\in\mathcal{N}, & (8a) \\ x_{n,f}\in\{0,1\}, \forall n\in\mathcal{N}, \forall f\in\mathcal{F}, & (8b) \end{cases}$$

where the constraint (8a) implies that each F-AP is allowed to cache at most S files, and the constraint (8b) implies that the caching strategy variable is binary.

III. Proposed MARL-based Cooperative Caching Scheme

The optimization problem in (8) is a constrained integer programming problem and non-deterministic polynomial hard (NP-hard), which generally requires exponential computational complexity for traditional simple searching approaches to obtain the globally optimal solution [5]. To solve the problem with low computational complexity, we propose an MARL-based cooperative caching scheme. We briefly introduce the DDQN in every F-AP to minimize the local transmission delay. However, individual training in the DDQN neglects the interaction among F-APs and cannot guarantee the minimum average transmission delay of the entire system. We then resort to MARL to build a communication procedure to leverage the cooperation among F-APs. By the joint learning of agents, the maximum global reward function is achieved and the average transmission delay of the entire system is minimized.

A. Reinforcement Learning Framework

We model the local transmission process in single F-AP as a Markov Decision Process (MDP) with state space, action space and reward function. In detail, agent n observes a state $\boldsymbol{s}_n^t$ from the environment and executes an action a_n^t during time slot t. Then, the environment feeds back a reward $r_n^t = r(\boldsymbol{s}_n^t, a_n^t)$ and the new state $\boldsymbol{s}_n^{t+1}$ to the agent. To employ the RL framework, the critical elements in MDP are identified as follows:

1) State Space: The state $\boldsymbol{s}_n^t \in \mathcal{S}_n$ indicates the cache status information of the n-th agent during time slot t and the cache status can be denoted by $\boldsymbol{s}_n^t=\{\boldsymbol{q}_n^t, f_n^t\}$. The former element $\boldsymbol{q}_n^t=\{q_{n,1}^t, q_{n,2}^t,...,q_{n,S}^t\}$ collects the indexes of cached files in agent n, which corresponds to the local caching strategy of F-AP n. The latter element $f_n^t \in \mathcal{F}$ is the requested file from the requesting user in the region of F-AP n.

2) Action Space: The objective of an agent is to map the space of states to the space of actions. The action of agent n is denoted by $a_n^t \in \mathcal{A}_n$. Let $a_n^t=0,1,...,S$, where $a_n^t=s (s\neq 0)$ means that the s-th cached content in F-AP n will be replaced by the requested file f_n^t, and $a_n^t=0$ means that the requested file f_n^t should not be cached. Then, the agent can update its own caching strategy according to the selected action.

3) Reward Function: When agent n selects an action a_n^t under the state $\boldsymbol{s}_n^t$, a reward function r_n^t is determined. The objective of RL is to obtain the minimum local transmission delay of F-AP n and to achieve the maximum reward. Thus, the reward function is designed as follows:

$$r_n^t(\boldsymbol{X}) = \sum_{f=1}^F P_{n,f}^t e^{-\lambda(d_{n,f}^t(\boldsymbol{X}) - Z_{1,n,u,f}^t)}, \tag{9}$$

where the exponential function is used to keep the reward function bigger than 0, and $\lambda (0 < \lambda \le 1)$ guarantees that the reward function is normalized.

Besides, the optimal action-value function $Q_n^t(\boldsymbol{s}_n^t, a_n^t)$ in agent n can be defined as follows:

$$\begin{aligned} Q_n^t(\boldsymbol{s}_n^t, a_n^t) \leftarrow Q_n^t(\boldsymbol{s}_n^t, a_n^t) + \alpha[r_n^{t+1} \\ + \gamma \max_{a_n^{t+1}} Q_n^t(\boldsymbol{s}_n^{t+1}, a_n^{t+1}) - Q_n^t(\boldsymbol{s}_n^t, a_n^t)], \end{aligned} \quad (10)$$

where α and γ denote learning rate and reward decay respectively.

B. Double Deep Q-Network

RL techniques such as DQN and DDQN are applied as the effective approaches to tackle the curse of dimensionality and achieve the maximum reward. In addition, compared with DQN algorithm, DDQN can decouple the action selection from the calculation in (10) to prevent the overoptimistic value estimates [16]. Correspondingly, DDQN based on RL is utilized to find the optimal strategy. In the architecture of DDQN, there are two separate neural networks, a current Q-network and a target Q-network. The current Q-network $Q_n^t(\boldsymbol{s}_n^{t+1}, a_n^{t+1}|\theta_n)$ with the network parameter θ_n is utilized for approximating $Q_n^t(\boldsymbol{s}_n^t, a_n^t)$ in (10). And the target Q-network $\hat{Q}_n^t(\boldsymbol{s}_n^t, a_n^t|\hat{\theta}_n)$ with the network parameter $\hat{\theta}_n$ is utilized for computing the target Q-value. It can be expressed as follows:

$$\hat{Q}_n^t(\boldsymbol{s}_n^t, a_n^t|\hat{\theta}_n) = r_n^{t+1} + \gamma \hat{Q}_n^t(\boldsymbol{s}_n^t, a'|\hat{\theta}_n), \quad (11)$$

where $a' = \text{argmax}_{a_n^{t+1}} Q_n^t(\boldsymbol{s}_n^{t+1}, a_n^{t+1}|\theta)$ is an action chosen from the current Q-network to maintain the current Q-value under the state $\boldsymbol{s}_n^{t+1}$, and $\hat{\theta}_n$ is the weight of the n-th target Q-network.

Instead of updating the network parameters of the target Q-network iteratively, they are copied from the current Q-network at intervals, i.e., delayed update, which reduces the correlation between the target Q-value and the current Q-value. The loss function in the network is updated via a gradient descent approach as follows [11]:

$$L(\theta_n) = (\hat{Q}_n^t(\boldsymbol{s}_n^t, a_n^t|\hat{\theta}_n) - Q_n^t(\boldsymbol{s}_n^t, a_n^t|\theta_n))^2, \quad (12)$$

where the current Q-network parameters θ_n can be obtained according to (12), and the target Q-network parameter $\hat{\theta}_n$ will copy θ_n from the current Q-network $Q_n^t(\boldsymbol{s}_n^t, a_n^t|\theta_n)$ every ν steps.

C. Proposed MARL-based Cooperative Caching Scheme

In the above work, we have utilized the DDQN in single F-AP. In order to leverage the cooperation among F-APs, we extend DDQN to multi-agent system and introduce the communication procedure among F-APs, which is illustrated in Fig. 2.

The global caching strategy can be formulated as Stochastic Game (SG) [12]. The SG model can be defined as $\{N, S_1 \times ... \times S_N, A_1 \times ... \times A_N, R^t\}$, where S_n is the state space of the n-th agent, A_n is the action space of the n-th agent, and R^t is the global reward function. So the joint action space is $\mathcal{A} = A_1 \times ... \times A_N$ and the joint state space is
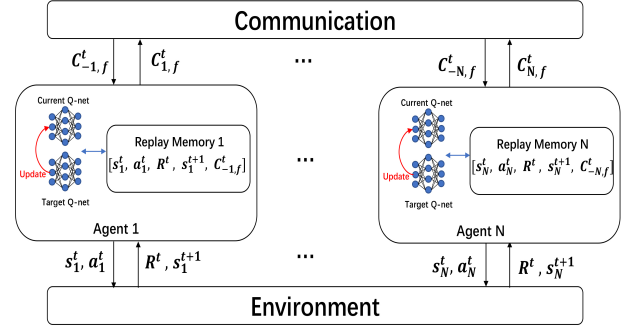

Fig. 2: Schematic of the MARL framework.

$\mathcal{S} = S_1 \times ... \times S_N$. Since every agent's action has an impact both on the local reward as well as on the global reward, all agents are expected to work cooperatively to find the globally optimal strategy that maximizes the global reward. By considering the reward function $r_n^t(\boldsymbol{X})$, the global reward function R^t can be defined as:

$$R^t(\boldsymbol{X}) = \sum_{n=1}^{N} r_n^t(\boldsymbol{X}). \quad (13)$$

The maximum reward function in (9) only indicates the minimum local transmission delay in single agent. To further optimize the caching strategy, we employ the global reward function in (13) instead of the local reward function in (9).

Next, we will use the joint learning of all agents to find the globally optimal caching strategy $\boldsymbol{X}^*$. Every agent updates its target Q-values according to the observation from communication procedure. Then, every agent and its associated agents jointly update their DDQNs by sampling from experience replies.

1) Communication Procedure: As the global reward function in (13) depends on the caching strategies of all agents, every agent should observe the historical caching strategies of its associated agents to adjust its own caching strategy. Thus, MARL introduces a communication procedure among agents. Each agent $n \in \mathcal{N}$ caches files in accordance with the current caching strategy of its associated agent $m \in \mathcal{N}_n$. We assume that agent n treats the relative observation of its associated agent m as the indicator of agent n's caching strategy. Let $C_{n,f}^t$ denote the number of times that the requested file f has been cached in agent n until time slot t. Agent n records $C_{n,f}^t$ according to its chosen action a_n^t. Then, we have $C_{-n,f}^t = \mathbb{E}_m[\sum_{m \in \mathcal{N}_n} C_{m,f}^t]/t$. In the communication procedure, agent n collects the relative observation $C_{-n,f}^t$ and stores in the experience reply $\mathcal{D}_n$ for updating its DDQN.

2) Update Target Q-values: When file f is requested in agent n, agent n observes the historical caching strategies of its associated agents and updates its own DDQN. For maximizing the global reward function, we rewrite the target Q-value in (11) as follows:

$$\hat{Q}_n^t(\boldsymbol{s}_n^t, a_n^t|\hat{\theta}_n) = \frac{1}{C_{-n,f}^t + 1}(R^t(\boldsymbol{X}) + \gamma \hat{Q}_n^t(\boldsymbol{s}_n^t, a'|\hat{\theta}_n)), \quad (14)$$

Algorithm 1 The MARL based cooperative caching scheme

1: Initialize the reply memories $\mathcal{D}_1, ..., \mathcal{D}_N$;
2: Initialize the current Q-network Q with the weight θ, and the target Q-network $\hat{Q}$ with the weight $\hat{\theta} = \theta$;
3: Initialize the count $C_{n,f} = 0, n \in \mathcal{N}, f \in \mathcal{F}$;
4: **for** time slot $t = 1, 2, ..., T$ **do**
5:    **for** F-AP $n = 1, 2, ..., N$ **do**
6:       Collect the requested file f from users in $\mathcal{U}_n$;
7:       Observe the state $\boldsymbol{s}_n^t = \left\{ q_{n,1}^t, q_{n,2}^t, ..., q_{n,S}^t, f \right\}$;
8:       Choose an action $a_n^t = \text{argmax}_a Q_n^t(s, a)$ using the ϵ-greedy policy under the current state $\boldsymbol{s}_n^t$;
9:       **if** action $a_n^t \neq 0$ **then**
10:          Update $C_{n,f} = C_{n,f} + 1$;
11:          Update $C_{n,a_n^t} = 0$;
12:          Execute the action a_n^t, and replace the a_n^t-th stored file in F-AP n with file f;
13:       **end if**
14:       Compute $C_{-n,f}^t = \mathbb{E}_m[\sum_{m \in \mathcal{N}_n} C_{m,f}^t]/t$;
15:       Save $\left[\boldsymbol{s}_n^t, a_n^t, R^t, \boldsymbol{s}_n^{t+1}, C_{-n,f}^t \right]$ in $\mathcal{D}_n$;
16:       **while** F-AP $m \in \mathcal{N}_n \cup \{n\}$ **do**
17:          Obtain the reward $R^t(\boldsymbol{X})$ according to (13);
18:          Randomly sample a mini-batch of experiences from $\mathcal{D}_m$;
19:          Update the target Q-values $\hat{Q}_m^t(\boldsymbol{s}_m^t, a_m^t | \hat{\theta}_m)$ according to (14);
20:          Update the weight θ_m by the loss function $L(\theta_m)$ according to (12);
21:          Reset $\hat{\theta}_m = \theta_m$ every ν time slots;
22:       **end while**
23:    **end for**
24:    Obtain the caching strategy $\boldsymbol{X}^*$ according to the joint state space $\mathcal{S} = S_1 \times ... \times S_N$;
25: **end for**

where $C_{-n,f}^t$ is an observation from which agent n observes the historical caching strategies of its associated agents during time slot t.

3) Joint Learning: Every agent and its associated agents jointly update their own DDQNs. Single agent n chooses the optimal action and stores the experience data $\left[\boldsymbol{s}_n^t, a_n^t, R^t, \boldsymbol{s}_n^{t+1}, C_{-n,f}^t \right]$ in reply memory $\mathcal{D}_n$. Based on MARL, agent n and its associated agents select randomly small batches of data from their own reply memories for updating their own DDQNs.

During each time slot, every agent learns from the interactions with environment and observes the historical caching strategies of its associated agents to choose the optimal action. After the joint learning, we can collect the joint caching space to obtain the globally optimal caching strategy. The detail of the proposed MARL based cooperative caching scheme is presented in Algorithm 1.

IV. SIMULATION RESULTS

The performance of the proposed MARL-based cooperative caching scheme is evaluated via simulations. The users' file preference follows the Mandelbrot-Zipf distribution with the skewness factor $\tau_t = 1.1$. The small-scale channel gain $|h_{a,b}^t|^2$ follows standard exponential distribution. The bandwidth B
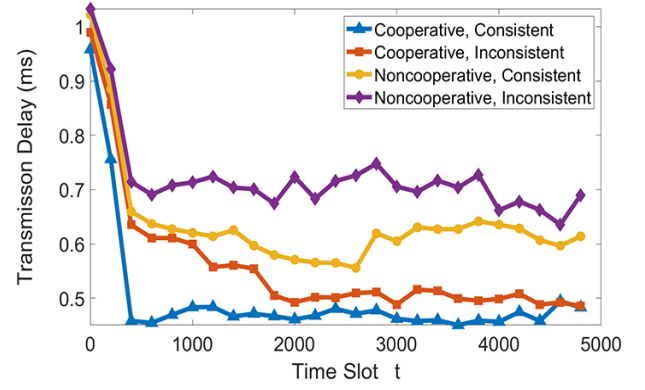

Fig. 3: Transmission delay versus different caching and different user preference.

is set to 100MHz [13]. Each F-AP serves the users in a circular cell with a radius of 100m. Assume that no inter-cell interference is induced. The file size is set to 1Mbits. For simplification, the transmission rate in backhaul link is set to $R =$100Mbps. The learning rate α is set to 0.001 and the reward decay γ is set to 0.9. Unless otherwise stated, we set $U = 50, F = 500, N = 5$. In the simulations, the traditional scheme (Least Recently Used (LRU)) and the learning schemes (DQN and Independent Q-learning (IQL)) are chosen as the benchmark schemes.

In Fig. 3, we show the delay performance of different caching and different user preference[1] based on MARL. It can be observed that the four schemes can approach their stable transmission delay as time slot increases. The noncooperative caching schemes have higher transmission delay than the cooperative caching schemes. The reason is that F-APs need to fetch more files from the cloud server in noncooperative caching schemes. It can also be observed that the transmission delay has the lowest value in the cooperative caching and consistent user preference scheme. That is because our proposed scheme can learn the user preference and get the content popularity of every F-AP.

In Fig. 4, we show the convergence performance of our proposed scheme in comparison with the three benchmark schemes. It can be observed that our proposed scheme converges to a relatively stable value when time slot t is larger than 2000. Compared with the benchmark schemes, our proposed scheme has lower convergence speed and better delay performance. The reason is that our proposed scheme has few records about the historical caching strategies at the beginning of the training. With the continuous caching updates, our proposed scheme can gradually leverage the cooperation among F-APs and find the globally optimal caching scheme. Meanwhile, LRU has the highest transmission delay as no learning is adopted. IQL and DQN have the close delay performances since they neglect the interactions among agents.

---

[1]For consistent user preference, we set the random permutation $\oplus_u^t$ as a constant. And for inconsistent user preference, we set the random permutation $\oplus_u^t$ as a time-varying random permutation of $\mathcal{F}$.
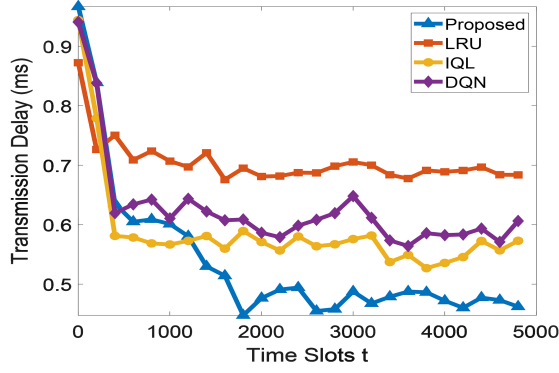
Fig. 4: Transmission delay versus time slot for the proposed scheme and three benchmark schemes.
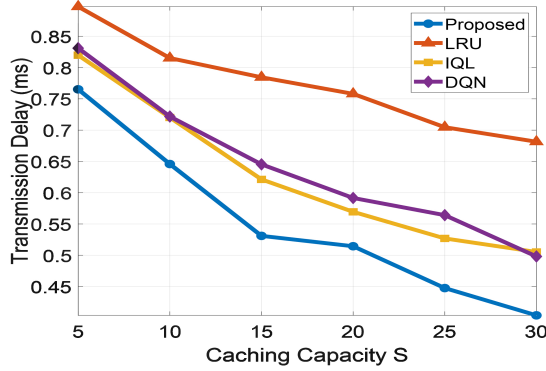

Fig. 5: Transmission delay versus cache capacity for the proposed scheme and three benchmark schemes.

In Fig. 5, we show the transmission delay of our proposed scheme and the benchmark schemes while varying the F-AP caching capacity. It can be observed that the transmission delay reduces as the caching capacity increases. It can also be observed that the transmission delay of our proposed scheme is always lower than that of the benchmark schemes. That is reasonable because larger caching capacity enables F-APs to cache more popular files simultaneously and our proposed scheme can utilize the communication among F-APs to reduce the average transmission delay.

V. CONCLUSIONS

In this paper, we have proposed an MARL-based cooperative caching scheme in F-RANs. In each F-AP, the DDQN has been utilized to meet the integer and cache capacity constraints. In addition, MARL has introduced the communication procedure to leverage the cooperation among F-APs. By recording the historical strategies of the associated F-APs, our proposed scheme has made agents communicate with other agents to maximize the global reward function and reduce the average transmission delay further. Simulation results have shown that our proposed scheme achieves a significant performance improvement compared with the benchmark schemes.

REFERENCES

[1] M. A. Habibi, M. Nasimi, B. Han, and H. D. Schotten, "A comprehensive survey of RAN architectures toward 5G mobile communication system," *IEEE Access*, vol. 7, pp. 70 371–70 421, May 2019.

[2] X. Wang, S. Leng, and K. Yang, "Social-aware edge caching in fog radio access networks," *IEEE Access*, vol. 5, pp. 8492–8501, Apr. 2017.

[3] M. Peng, S. Yan, K. Zhang, and C. Wang, "Fog-computing-based radio access networks: Issues and challenges," *IEEE Network*, vol. 30, no. 4, pp. 46–53, Jul. 2016.

[4] C. Xia, Y. Jiang, M. Peng, F.-C. Zheng, M. Bennis, and X. You, "Cooperative edge caching in fog radio access networks: A pigeon inspired optimization approach," in *2019 IEEE Global Communications Conference (GLOBECOM)*, Feb. 2019, pp. 1–6.

[5] Y. Jiang, X. Chen, F.-C. Zheng, D. Niyato, and X. You, "Brain storm optimization-based edge caching in fog radio access networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 2, pp. 1807–1820, Jan. 2021.

[6] C. Wang, S. Wang, D. Li, X. Wang, X. Li, and V. C. M. Leung, "Q-learning based edge caching optimization for D2D enabled hierarchical wireless networks," in *2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, Oct. 2018, pp. 55–63.

[7] Z. Yang, Y. Liu, Y. Chen, and L. Jiao, "Learning automata based Q-Learning for content placement in cooperative caching," *IEEE Transactions on Communications*, vol. 68, no. 6, pp. 3667–3680, Mar. 2020.

[8] B. Guo, X. Zhang, Q. Sheng, and H. Yang, "Dueling deep-Q-network based delay-aware cache update policy for mobile users in fog radio access networks," *IEEE Access*, vol. 8, pp. 7131–7141, Jan. 2020.

[9] J. Yan, Y. Jiang, F. Zheng, F. R. Yu, X. Gao, and X. You, "Distributed edge caching with content recommendation in fog-rans via deep reinforcement learning," in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, Jul. 2020, pp. 1–6.

[10] L. Li, Y. Xu, J. Yin, W. Liang, X. Li, W. Chen, and Z. Han, "Deep reinforcement learning approaches for content caching in cache-enabled D2D networks," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 544–557, Nov. 2020.

[11] M. Zhang, Y. Jiang, F.-C. Zheng, M. Bennis, and X. You, "Cooperative edge caching via federated deep reinforcement learning in fog-rans," in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, Jul. 2021, pp. 1–6.

[12] K. Jiang, H. Zhou, D. Zeng, and J. Wu, "Multi-agent reinforcement learning for cooperative edge caching in internet of vehicles," in *2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, Dec. 2020, pp. 455–463.

[13] J. Liu, B. Bai, J. Zhang, and K. B. Letaief, "Cache placement in fog-rans: From centralized to distributed algorithms," *IEEE Transactions on Wireless Communications*, vol. 16, no. 11, pp. 7039–7051, Aug. 2017.

[14] Z. Silagadze, "Citations and the Zipf-Mandelbrot law," *COMPLEX SYSTEMS -CHAMPAIGN-*, vol. 11, no. 6, pp. 487–500, Sep. 1997.

[15] Y. Jiang, Y. Hu, M. Bennis, F.-C. Zheng, and X. You, "A mean field game-based distributed edge caching in fog radio access networks," *IEEE Transactions on Communications*, vol. 68, no. 3, pp. 1567–1580, Dec. 2020.

[16] H. van Hasselt, A. Guez, and D. Silver, "Deep Reinforcement Learning with Double Q-learning," *arXiv e-prints*, p. arXiv:1509.06461, Sep. 2015.