Non-episodic and Heterogeneous Environment in Distributed Multi-agent Reinforcement Learning

Fenghe Hu, Yansha Deng, and A. Hamid Aghvami

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10000672

***Abstract*—Reinforcement learning (RL) is a efficient intelligent algorithm when solving radio resource management problems in the wireless communication network. However, for large-scale networks with limited centralization (i.e., high latency connection to center-server or capacity-limited backbone), it is not realistic to employ a centralized RL algorithm to perform joint real-time decision-making for the entire network, which calls for scalable algorithm designs. Multi-agent RL, which allows separate local execution of policy, has been applied to large-scale wireless communication areas. However, it has performance issue which largely varies with different system settings. In this paper, we study a multi-agent algorithm for a coordinate multipoint (CoMP) scenario, which requires cooperation between base stations. We show that the common settings of user distribution, the design of reward, and episodic in the environment can significantly ease the learning of the algorithm and obtain beautiful converge results. However, these settings are not realistic in wireless communication. By validating the performance difference between these settings with our algorithm in a coordinate multipoint (CoMP) scenario, we introduce several possible solutions and highlight the necessity of further study in this area.**

I. Introduction

As a definite developing direction in future wireless communication networks, a self-organized network should automatically make decisions (radio resource management, routing, edge computing resource management, etc.) based on diverse service requirements via intelligent algorithms such as reinforcement learning (RL). One common solution is employing a centralized RL controller to make allocation or routing decisions. Although RL algorithms are capable of learning from the environment experience. The commonly used centralized RL requires accessing information throughout the network to make a suitable decision. Unfortunately, when being applied to a large-scale network, such a centralized structure can cause significant overhead in backbone transmission and computation complexity, which makes it unacceptable [1], [2]. That is a so-called high-dimensional or scalability problem limiting RL's large-scale wireless network application. Existing solutions dealing with the large-scale network with the aforementioned scalability problem mainly use static, greedy, or game-theorem algorithms, which are not sensitive to the network scale but usually lead to sub-optimal solutions [3]. To fully realize the benefit of RL in large-scale networks, researchers have tried to distribute the action decisions into entities inside the network, i.e., multi-agent RL, which is recognized as the enabling technology for the large-scale network. Such new learning algorithms allow each network entity (e.g., access-points) to make its own decisions distributively while still optimizing the common global target. However, unlike single-agent RL which has successfully solved many real-world problems with decent performance, multi-agent RL still suffers from poor performance and instability due to the non-stationary environment caused by unknown information from other network entities, i.e., agents.

F. Hu, Y. Deng, and A. H. Aghvami are with King's College London, UK (E-mail:fenghe.hu, hamid.aghvami@kcl.ac.uk).

This work was supported by Engineering and Physical Sciences Research Council (EPSRC), UK., under Grant EP/W004348/1.

Among multi-agent RL algorithms in [2] solving wireless communication problems, i.e. radio resource management (RRM) [4], unmanned aerial vehicle (UAV) route management [5], mobile edge computing (MEC) resource management [6]. Centralized-training-decentralized-execution (CTDE) is the most widely used scheme, whose performance has theoretical proof. In CTDE, each agent holds a policy trained via a central server by collecting observations from agents. It is reasonable for these works to assume homogeneous settings, for example, the users follow Poisson Point Process (PPP) [7] or the elements in MIMO channel metric [8]. But in some cases, the users' positions' distribution can highly vary due to the existence of hot points. Besides, some works also use a well-defined fine-tuned quality-of-service (QoS) function as a reward, which requires prior knowledge of the task to design [6]. In practice, we may find it only possible to obtain the signal-to-interference-noise (SINR) ratio and modify it without prior knowledge. Besides, we also notice that all aforementioned works in the communication area are episodic, allowing the environment to reset or have a limited horizon. However, the interaction is a continuous procedure in the real world, which means no reset is possible, and the horizon is infinite. This setting is recognized as non-episodic, which is critical for RL in continuous control in real-time, for example, robot arm. The optimization target changes from the maximization of accumulative reward to average reward in this case, which is shown as a significant performance difference in the performance of the RL algorithm [9], [10]. There are limited discussions on the influence of these environment settings on the performance of multi-agent RL algorithms.

Inspired by previous works, we apply a multi-agent RL algorithm with actor-critic agents trained via CTDE. We discuss the influence of different environment settings and show the performance with the CoMP environment. The contributions of this paper include:

- We highlight the use of the homogeneous distribution,

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 26,2023 at 04:53:42 UTC from IEEE Xplore. Restrictions apply.

i.e., PPP can largely bring up the algorithm's performance, compared to heterogeneous distribution, i.e., Poisson Cluster Process (PCP).
- We show that the direct use of SINR value as the reward can decrease the algorithm's performance due to its low variance, while the well-modified reward can lead to a converged result.
- We present the algorithms under both episodic and non-episodic settings and show the convergence under non-episodic is more complicated than that under the episodic setting.
- We introduce the idea of applying transfer learning to allow personalization and improve the algorithm's cooperation performance with PCP distribution.

II. Distributed Model Wireless Networks

This section presents the system model and basic information of a joint-transmission coordinated multipoint (CoMP) in a large-scale multi-cell network.

We consider the joint-transmission CoMP (JT-CoMP) for downlink transmission with a set of access points (APs), denoted by $\mathcal{B}$. For simplicity, each AP is located in a hexagonal grid and equipped with one omnidirectional antenna for downlink transmission. All APs are connected via fibre links, which allow data sharing through a central unit. A set of users, denoted by $\mathcal{U}$, are located in the serving area following the Poisson point process (PPP) or Poisson cluster process (PCP). To enhance the quality-of-service (QoS) for the cell-edge users, the neighbouring APs seek to form cooperative clusters [11], where the signals are transmitted and enhanced by cooperative APs using the same frequency band. Through joint transmission, the CoMP technology enhances the cell-edge users' QoS at the cost of backhaul overhead and frequency resource. The larger the cluster size, the more effective cooperation among APs, and the higher backhaul capacity and frequency resource requirements. Due to this trade-off, we consider a maximum cluster size of $B_{\max}$ [12]. The cluster association or cooperation decisions are re-calculated after a certain time.

Serving by such a network, we consider a traffic model where users request a certain amount of data D from APs. The request is deemed to be failed if it is not satisfied within a particular time. Then, we qualify the performance of the network with a certain cooperation scheme in the t time slot via a QoS function, which is a function of the resulting users' signal-to-interference-noise-ratio:

$$r_t = \sum_{u \in \mathcal{U}} \log_2(1 + \underbrace{\frac{\sum_{i \in \mathcal{B}^u} P_b |w_{b,u}\beta_{b,u}|^2 d_{b,u}^{-\alpha}}{\sum_{i \in \mathcal{B}^{\mathcal{U}/u}} P_b |w_{i,u'}\beta_{i,u}|^2 d_{i,u}^{-\alpha} + \sigma^2}}_{\text{SINR}_u}), \quad (1)$$

where P_b is the transmit signal power from i-th agent, $\beta_{b,u}$ is the small-scale fading factor, $w_{b,u}$ is the beamforming vector, $d_{b,u}^{-\alpha}$ denotes the free-space large-scale fading that depends on the distance and path loss factor α, $\mathcal{B}^u$ is the set of associated APs (include the cooperation APs) of user u, $\mathcal{B}^{\mathcal{U}/u}$ are the set of APs which do not with user u, σ^2 is the noise power.

Then, we define a policy π to decide which APs should belong to the same cluster dynamically over time. In this way, the optimization problem for our example CoMP clustering scenario can be written as

$$\max_{\pi} \sum_{t}^{\infty} [r_t | \{\mathcal{B}^u\}_{u \in \mathcal{U}} \sim \pi], \qquad s.t. \quad |\mathcal{B}^u| \le B_{\max} \quad (2)$$

where r_t is defined in Eq.(1), $\mathcal{B}^u$ denotes the set of APs serving uth user cooperatively, decided by π, and $B_{\max}$ is the maximum cluster size, which is usually around 3 [11].

III. Value Decomposition in Multi-agent Reinforcement Learning

In this section, we define the considered CoMP problem as a stochastic game, which is then solved via our multi-agent algorithm.

A. Stochastic Game Definition

To solve our considered problem with RL methods, we first define our problem as a networked stochastic game, which can be characterized by a tuple of $< \mathcal{S}, \mathcal{B}, \{\mathcal{O}^b\}, P, \{\mathcal{A}\}, \{\mathcal{A}^b\}, \{\mathcal{R}^b\}, \Omega >$. We define each component of this tuple notations as

- $\mathcal{S}$ is a set of joint state ($s \in \mathcal{S}$), and $\mathcal{S}^b$ presents the set of local state of agent i ($s^b \in \mathcal{S}^b$). S_t is the state at time t, which includes users' position, SINR, neighbouring AP cooperation state, and AP's transmit power, etc.
- $\mathcal{B}$ is the set of agents ($b \in \mathcal{B}$), which are co-located with each AP,
- $\mathcal{O}^b$ is a set of local observations of the i-th agent ($\mathcal{O}$), which contains users' location, neighbouring AP's location and request status,
- P is a transition probability function which maps the state-action to the next state, i.e. $P(s'|s, a) : \mathcal{S} \times \mathcal{A} \times \mathcal{S}$,
- Ω is the observation function which maps the local state of agent i to its observation, i.e. $\Omega(s_b) : \mathcal{S}^b \times \Omega \to [0, 1]$, which is decided by APs' sensors' capability. As the limited sensor capability, the observation at the AP usually contains less information than the original state. Thus, we assume that the mapping function Ω is stationary and contractive. In this way, the problem can be analysed as a fully-observable problem with Ω added.
- $\mathcal{A}$ is the set of joint actions of agents ($a \in \mathcal{A}$) of all APs. The action a^b of ith AP is the cooperation decision which reflects its request to cooperate with a certain number of the neighbouring APs. By exchanging the requests among APs, a cooperation cluster is formed when both APs achieve an agreement in cooperation. The local action set of the ith AP is given as $\mathcal{A}^b$ ($a^b \in \mathcal{A}^b$). In our considered scenario, the size of local action space is defined as $|\mathcal{A}^b| = \sum_{c=0}^{C} |\mathcal{B}^{-b}|!/(c!(|\mathcal{B}^{-b}| - c)!)$, where C is the maximum cluster size. It is worth mentioning that the joint action space increase exponentially with the number

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 26,2023 at 04:53:42 UTC from IEEE Xplore. Restrictions apply.

of cooperative APs, i.e $|\mathcal{A}| = (|\mathcal{A}^b|^{|\mathcal{B}|})$, which results in the scalability problem,

- The QoS r^b (given in Eq.(1)) is used as the reward function for the ith agent, i.e. $r^b(s,a) : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$. r is the overall sum reward for all agents.

In each round of decisions, each AP observes its surrounding environment in-state S_t. Then, each AP chooses its action according to its local policy π_{θ^b} defined by a set of parameters θ^b, where the probability of choosing action a^b with observation o^b is represented as $\pi_{\theta^b} : \mathcal{O}^b \times \mathcal{A}^b \rightarrow [0,1]$. With a set of new cooperation decisions, the cooperation clusters form after the users are served under current cooperation decisions for a certain time slot. Before generating the new cooperation decision, each AP observes a local reward, which measures the efficiency of the current cooperation decision. The reward is then used to update the local policy. Then, the considered system shifts to a new state S_{t+1}, while all APs observe their observation O^b_{t+1} from state S^b_{t+1} based on function Ω. Then the aforementioned process is repeated, and the state-action pair forms a trajectory τ.

IV. Algorithm

We show the detailed parameter update steps of our considered actor-critic multi-agent RL algorithm for our considered problem in Algorithm.1, where the Q-function is estimated in the critic part and guides the update of the policy generated by the actor part. We define the estimator in i-th agent for Q-function with parameter ω^b and value function with δ^b, respectively. Without the actual information of $a^{b'}$, the estimator minimizes the error:

$$\epsilon(\omega^b) = Q_{\theta^b}(\bar{s}_b, a_b) - Q_{\omega^b}(s_b, a_b) \tag{3}$$

For episodic task, we define the parameters for actor and critic in agent i at time t as θ^b_t and ω^b_t. With joint state S_t and action A_t, the update procedure in i-th agent for critic network with the TD-error at time instant t follows

$$\begin{aligned}\overline{\omega}^b_{t+1} \leftarrow \omega^b_t + \alpha^\omega_t \nabla_\omega Q_{\omega^b_t}(S^b_t, A^b_t, A^{-b}_t)(r^b_{t+1} \\ + Q_{\omega^b_t}(S^b_{t+1}, A^b_{t+1}, A^{-b}_t) - Q_{\omega^b_t}(S^b_t, A^b_t, A^{-b}_t)),\end{aligned} \tag{4}$$

where α^ω_t is the step size for the critic network. With distributional RL, the TD error is the cross-entropy loss of the KL divergence between the current return and the estimated distribution of the return following the categorical algorithm in [13, Algorithm. 1]. The actor is updated as follows

$$\overline{\theta}^b_{t+1} \leftarrow \theta^b_t + \alpha^\theta_t \nabla_\theta \log \pi_{\theta^b}(O^b_t|S^b_t) Q_{\theta^b_t}(S^b_{t+1}, A^b_{t+1}, A^{-b}_t), \tag{5}$$

where α^θ_t is the step size for the critic network.

For non-episodic task, we introduce average-reward as $r(\pi_\theta) = \mathbb{E}_{(s,a)\sim\mathbb{P}_\theta(s,a)}[r(s,a)] = \sum_s d_\theta(s) \sum_a \pi_\theta(a|s) r(s,a)$. And the algorithm fit the differential return between rewards and the average reward. The network update with TD-error and estimated average

Algorithm 1: Actor-critic Multi-agent Reinforcement Learning Algorithm.

```
1  Initiate environment $Env$, state $s_0$, and the initial
     values of the parameters $\{\theta^b\}_{i\in\mathcal{B}}$ and $\{\omega^b\}_{i\in\mathcal{B}}$.
2  repeat
3      if Game end then
4          Reset $Env$ and $t = 0$, obtain new $S_0$
5      for $i \in \mathcal{B}$ do
6          Obtain $O^b_t$ from $S_t$
7          Select an action $A^b_t \sim \pi_{\theta^b_t}(O^b_t)$
8      Forms joint action $a_t = (A^b_t)_{i\in\mathcal{B}}$, the environment
         move to $S_{t+1}$
9      for $i \in \mathcal{B}$ do
10         Observe local reward $r^b_t$ from $S_{t+1}$
11     Update actor's and critic's parameters following
         Eq.(4) and Eq.(5) or categorical algorithm
12 until Performance Not Improved
```

reward $\hat{r}^b_t$ at i-th agent following:

$$\begin{aligned}\overline{\omega}^b_{t+1} \leftarrow& \omega^b_t + \alpha^\omega_t \nabla_\omega Q_{\omega^b_t}(S^b_t, A^b_t, A^{-b}_t)(r^b_{t+1} \\ &- \hat{r}^b_t + Q_{\omega^b_t}(S^b_{t+1}, A^b_{t+1}, A^{-b}_t) - Q_{\omega^b_t}(S^b_t, A^b_t, A^{-b}_t)).\end{aligned} \tag{6}$$

The average reward is updated via:

$$\begin{aligned}\hat{r}^b_{t+1} \leftarrow& \hat{r}^b_t + \alpha^r(r^b_{t+1} \\ &- \hat{r}^b_t + Q_{\omega^b_t}(S^b_{t+1}, A^b_{t+1}, A^{-b}_t) - Q_{\omega^b_t}(S^b_t, A^b_t, A^{-b}_t)),\end{aligned} \tag{7}$$

where α^r is the reward update parameter.

V. Influence of Environment Setting

Before, we put this relatively direct distributed multi-agent learning algorithm into the environment. We notice that several environment settings may significantly influence the algorithm's performance.

A. User Location Distribution

It is common to assume that the users are located following PPP. As the points are randomly distributed, PPP is a homogeneous distribution where the characteristics of users' distribution are identical for all APs, i.e., the distribution of users' locations is independent and equal (IID). Then, the experience from each AP can be shared entirely and re-used by all other APs, which enhances the performance of the CTDE method. However, this is not the case in practice, where local hot points or terrains can largely affect users' locations. For example, the existence of a river or building limits the mobility of users, which leads to the different geometrical characteristics of each APs. Non-IID data is recognized as one of the significant problems for the CTDE method. In our simulation environment, the AP at the edge of the network has a minimal choice of cooperation, whose experience is highly personalized and not suitable to be entirely accepted by other agents. The centralized trained model may not always be generalized enough to handle the heterogeneity environment. A certain agent's policy's sub-optimality or local characteristic

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 26,2023 at 04:53:42 UTC from IEEE Xplore. Restrictions apply.

can propagate through a federated process and negatively affect other agents' performance.

It is possible to solve this problem by allowing each AP to keep part of its penalization while performing CTDE. Instead of associating the trained model directly back to each AP after training, it is possible to transfer the global knowledge to APs while keeping the local characteristics. In this way, this method keeps the advantages of the CTDE method, i.e., fast converging, while avoiding the disadvantages.

B. Reward Design

The average SINR is the dedicated and easiest performance metric for radio resource management problems. However, most algorithms' decisions have very limited influence on average SINR. In our considered CoMP scenario, the cooperation can only benefit the cell-edge users. When summed and averaged as average SINR defined by Eq.(1), there is only around 5% performance gain. Furthermore, due to the random nature of the wireless signal, the variance of the average SINR is large. Even a worse cooperation decision can sometimes obtain a higher averaged SINR than some "better" decisions. It has been shown such a small reward variance between different policies is hard to be captured by reinforcement learning algorithms and can lead to significant performance loss, or un-converged results [14]. Such reward characteristics also mess up the algorithm and make it hard to evaluate and select the correct decision. The phenomenon highlights the necessity of reward shaping. It is common to apply a linear normalized multiplier [15]. The new reward is defined as

$$r_t^{\text{scale}} = C_{\text{scale}} r_t. \tag{8}$$

However, the common re-scale or normalization methods cannot separate the good/bad cooperation decisions. One of the possible ways is manually tuning the reward function based on some prior knowledge of the task, i.e. removing the cell-central users whose SINR are meaningless to our optimization target. This increases the performance gap between good/bad cooperation decisions and makes it easier for the algorithm to distinguish. But this method introduces extra complexity and requires prior knowledge to perform the such modification. The reward function which excludes the SINR from cell-central users is written as

$$r_t^{\text{exclude}} = \sum_{u \in \mathcal{U}} r_t^u, \text{ where } d_{b,u} \leq d_{\text{central}}, \tag{9}$$

where r_t^u is the QoS function from uth user, $d_{b,u} \leq d_{\text{central}}$ is the distance constrain where the ith AP and uth user must be far enough to be considered as a cell-edge user.

C. Episodic and Non-episodic Environment

Most reinforcement learning approaches in the communication area work on the episodic task, where the environment is reset between trails and the initial environment with a well-shaped reward, which is common in learning algorithms for chess or other small games. However, in practice, the algorithms should learn from continued interactions with the natural world without the possibility of resetting the environment. Reinforcement learning in a continuous environment without a reset, which is known as non-episodic learning, has been shown difficult to converge [9]. In a non-episodic task, the algorithm is not optimizing the accumulated future reward but averaged reward, which is shown in (6). One of the major challenges for non-episodic learning is that the algorithm can be trapped into irreversible states and stay at local optimal. Fortunately, irreversible states are seldom considered in communication scenarios, where the service can fail and disappear.

The optimization target for non-episodic tasks changes from maximising the time-accumulated reward to averaged reward. For non-episodic task ($T = \infty$), the average-reward as $r(\pi_\theta) = \mathbb{E}_{(s,a)\sim\mathbb{P}_\theta(s,a)}[r(s,a)] = \sum_s d_\theta(s) \sum_a \pi_\theta(a|s) r(s,a)$, and the critic network fit the differential return between rewards and the average reward. In this way, the network selects the action with the highest average reward expectation. During the learning process, the average reward is updated via

$$\begin{aligned} \hat{r}_{t+1}^b \leftarrow & \hat{r}_t^b + \alpha^r (r_{t+1}^b \\ & - \hat{r}_t^b + Q_{\omega_t^b}(S_{t+1}^b, A_{t+1}^b, A_t^{-b}) - Q_{\omega_t^b}(S_t^b, A_t^b, A_t^{-b})), \end{aligned} \tag{10}$$

where α^r is the reward update parameter.

VI. Result

In this section, we provide simulation results to show the effectiveness of our design and verify several architecture designs with our CoMP example case, which requires super high flexibility and scalability. In this example, we use this example to show the learning algorithms under our architecture can effectively solve the problem with decent scalability.

We consider a $182m \times 168m$ serving area with $|\mathcal{U}| = 160$ PCP distributed users in 10 clusters with largest radius of $40\,\text{m}$. The new users' position is generated every time slot, and the old users stay on the grid for two time slots before the service time-out. The users are served by $|\mathcal{B}| = 5 \times 4 = 20$ APs. The APs are distributed in cellular with six neighbours. The gap between neighbouring APs is $44.3m$ or $52m$. The number of possible cooperation actions is $|\mathcal{A} = 12|$, as the maximum size of the cluster is considered as 3. The APs choose one or two of its neighbour to cooperate with. The cooperation is considered established if both neighbouring APs select each other to cooperate, i.e., hand-shake. In this way, the cooperation decision can be made for all APs within a round-trip time between the current AP and its neighbours without exchanging information between the central server.

For the interaction between environment and learning algorithm, all agents observe the users inside its observation range and make decisions. The decisions are exchanged between neighbors, and the cooperation scheme is decided on the exchanged decisions. Then, the network serves users with the cooperation scheme and receives a reward. The reward is set as the summed rate of cell-edge users (decided by range), which benefits the most from the CoMP. The aforementioned proce-

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 26,2023 at 04:53:42 UTC from IEEE Xplore. Restrictions apply.

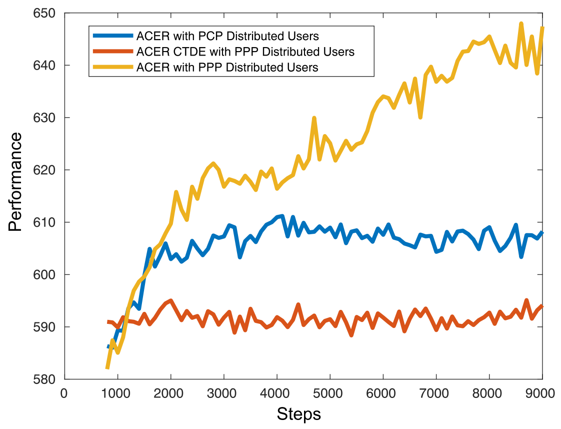

Figure 1. Performance of ACER algorithm with PCP and PPP distributed users.

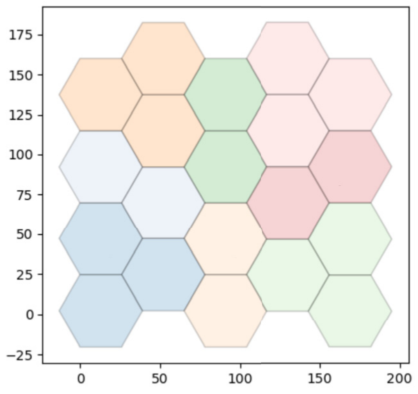

Figure 2. Averaged cooperation decision made by ACER algorithm with PPP distributed algorithm.

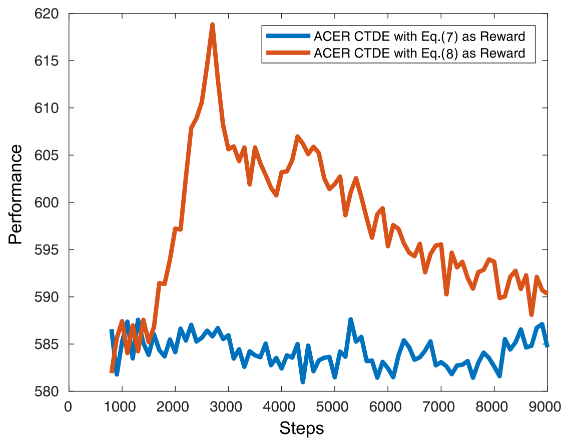

Figure 3. Performance of ACER algorithm with different reward functions.

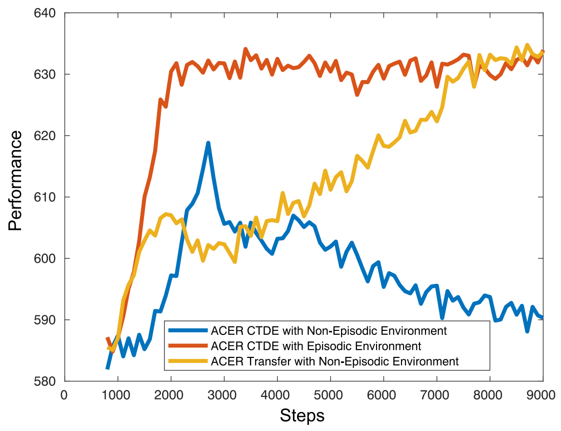

Figure 4. Performance of ACER in Episodic and Non-episodic Environment.

dures are performed without resetting to some default states, i.e., non-episodic. As we illustrated before, the RL algorithms have significantly different performances in episodic and non-episodic environments. Empirically, the episodic environment eases the learning difficulty, which is shown in Fig.1. There is limited analysis for a reason for this phenomenon [9], [10].

For the neural network design, we adopt a three-layer convolutional neural network to capture the geometry correlation between APs and users. The network takes a picture whose pixel value presents neighbor APs or users in the corresponding location. The value is added and normalized if multiple users are located in the same pixel. In this way, the network takes the users' geometry information with constant input size. After the input is processed as a hidden vector, the hidden vector is processed as the final policy output by a distributional RL structure with three linear layers [13]. The linear layers add noise to the result for state-based exploration [16]. For the RL algorithms, we show the performance of different RL approaches with our architecture in an example CoMP environment. We consider actor-critic with experience relay (ACER) in our simulation with centralized-training-decentralized-execution (ACER CDTE), federated averaging (ACER FedAvg), and the introducing transfer learning methods (ACER Transfer).

The size of this association problem in our defined environment is 12^{20}, which is over $4e22$ and impossible to be captured by any existing centralized learning approach. By leveraging the advantage of the communication environment, mean-field theory, and graph neural network, our architecture decomposes the problem geometrically and degrades the system complexity from $\mathcal{O}(|\mathcal{A}|^{|\mathcal{B}|})$ to $\sum_{|\mathcal{B}|}\mathcal{O}(|\mathcal{A}|)$ and can be applied in computation resource and memory limited devices. These designs significantly reduce the complexity of cooperative algorithms in large-scale networks. The capability of solving this problem already proves the scalability of our introduced architecture.

In Fig.1, we show the performance of algorithms in the non-episodic environment with different user distributions. The ACER algorithm converged and obtained good performance with PPP distributed users. We plot the averaged cooperation decision made by the ACER algorithm with PPP distributed users in Fig.2, which shows a policy maximizes the number of

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 26,2023 at 04:53:42 UTC from IEEE Xplore. Restrictions apply.

cooperating APs. It is pretty reasonable with PPP distributed users, as the cooperation can purely bring performance gain in our considered QoS function. On another side, as shown in Fig.1, the ACER performance can be largely brought down with PCP distributed users. That's because the heterogeneous environment is challenging for APs to find an acceptable cooperation policy. The opponent's environment might have completely unknown characteristics, which is impossible to predict without neighbourhood information. It also should be highlighted that the ACER CTDE algorithm brings the performance even down with the biased experience from different APs. The performance of ACER CTDE is worse than that of ACER. The ACER Transfer can solve this problem and leads to a converged result, which is shown in Fig.4. That's because transfer learning can effectively share problem-solving knowledge between agents without violent the local bias.

In Fig.3, we show the performance of the ACER CTDE algorithm with different reward functions in the non-episodic environment with PCP distributed users. The algorithm fails to converge with Eq.(8). The reason is a low difference and high variance SINR value falls in guiding the algorithm to converge, and the convergence result is very typical in multi-agent cases [17]. The difference between the fixed scheme and random action selection is only 5%. However, the reward is normalized and re-scaled. The reward variance is still high, as the wireless environment is highly random. The algorithm may get a higher reward with a worse policy. Thus, the algorithm cannot directly get which policy is better in most cases. Meanwhile, the reward in Eq.(9) can effectively support the ACER CTDE algorithm and achieve a good performance. By removing the users in the cell central area, the difference between policies is raised, and the variance is lowered. But the prior information is required to decide the cell-central area. It raises a problem: which prior information is suitable for reward. It also introduces a trade-off between information collection and algorithm performance. Since it is not applicable for APs to collect various information and tune rewards for each application or algorithm separately based on their special design.

In Fig.4, we plot the performance of ACER CTDE, and ACER Transfer algorithms with PCP distributed users in episodic and non-episodic environments. The transfer learning methods significantly enhance the convergence result because it allows each agent to keep its personalization information based on its local environment characteristics, while the CTDE method uses the same policy in all agents. We observe that the episodic environment significantly eases the convergence and leads to stable and good performance, while the performance is still limited and unstable in a non-episodic environment. It should be highlighted that the transfer method can achieve good performance in this case, which lacks further analysis.

VII. Conclusion

In this paper, we investigate the influence of a non-episodic and heterogeneous environment on the performance of a multi-agent reinforcement learning algorithm, which is critical when solving scalability problems in wireless communication. We also explain the reason for reward design which can potentially have a significant influence on the performance of the algorithms. We examine our results with a coordinated multipoint scenario and reinforcement learning algorithms. Our result shows that both non-episodic and heterogeneous environments can largely bring down the performance of the algorithms. We introduce the transfer learning method into a multi-agent reinforcement learning algorithm to allow personal characteristics in each agent, which can greatly enhance the performance in a heterogeneous environment. We believe our results can call for attention and inspire research in this area.

References

[1] F. B. Mismar and B. L. Evans, "Machine Learning in Downlink Coordinated Multipoint in Heterogeneous Networks," *arXiv e-prints*, Aug. 2016.

[2] Y. Al-Eryani, M. Akrout, and E. Hossain, "Multiple Access in Cell-Free Networks: Outage Performance, Dynamic Clustering, and Deep Reinforcement Learning-Based Design," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 4, pp. 1028–1042, Aug. 2021.

[3] S. Bassoy, H. Farooq, M. A. Imran, and A. Imran, "Coordinated Multi-Point Clustering Schemes: A Survey," *IEEE Communications Surveys Tutorials*, vol. 19, no. 2, pp. 743–764, Feb. 2017.

[4] Y. S. Nasir and D. Guo, "Multi-Agent Deep Reinforcement Learning for Dynamic Power Allocation in Wireless Networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2239–2250, Aug. 2019.

[5] H. Shiri, J. Park, and M. Bennis, "Communication-Efficient Massive UAV Online Path Control: Federated Learning Meets Mean-Field Game Theory," *arXiv e-prints*, Mar. 2020.

[6] H. Peng and X. Shen, "Multi-Agent Reinforcement Learning Based Resource Management in MEC- and UAV-Assisted Vehicular Networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 131–141, Nov. 2021.

[7] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J. Wang, "Mean Field Multi-Agent Reinforcement Learning," *arXiv e-prints*, Feb. 2018.

[8] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "Graph Neural Networks for Scalable Radio Resource Management: Architecture Design and Theoretical Analysis," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 101–115, Nov. 2021.

[9] J. D. Co-Reyes, S. Sanjeev, G. Berseth, A. Gupta, and S. Levine, "Ecological Reinforcement Learning," *arXiv e-prints*, p. arXiv:2006.12478, Jun. 2020. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2020arXiv200612478C

[10] A. Naik, R. Shariff, N. Yasui, H. Yao, and R. S. Sutton, "Discounted Reinforcement Learning Is Not an Optimization Problem," *arXiv e-prints*, p. arXiv:1910.02140, Oct. 2019. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2019arXiv191002140N

[11] P. Georgakopoulos, T. Akhtar, I. Politis, C. Tselios, E. Markakis, and S. Kotsopoulos, "Coordination Multipoint Enabled Small Cells for Coalition-Game-Based Radio Resource Management," *IEEE Netw.*, vol. 33, no. 4, pp. 63–69, Jul. 2019.

[12] F. Guidolin, L. Badia, and M. Zorzi, "A Distributed Clustering Algorithm for Coordinated Multipoint in LTE Networks," *IEEE Wireless Commun. Lett.*, vol. 3, no. 5, pp. 517–520, Jul. 2014.

[13] M. G. Bellemare, W. Dabney, and R. Munos, "A Distributional Perspective on Reinforcement Learning," *arXiv e-prints*, Jul. 2017.

[14] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep Reinforcement Learning that Matters," *arXiv e-prints*, Sep. 2017.

[15] N. Jiang, Y. Deng, A. Nallanathan, and J. A. Chambers, "Reinforcement Learning for Real-Time Optimization in NB-IoT Networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1424–1440, Mar. 2019.

[16] M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining Improvements in Deep Reinforcement Learning," *arXiv e-prints*, Oct. 2017.

[17] Y. Zhang, Q. Yang, D. An, and C. Zhang, "Coordination between individual agents in multi-agent reinforcement learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 13, pp. 11 387–11 394, May 2021. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/17357

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 26,2023 at 04:53:42 UTC from IEEE Xplore. Restrictions apply.