2022 International Conference on Computing, Communication, Perception and Quantum Technology (CCPQT) | 978-1-6654-7020-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/CCPQT56151.2022.00050

MEC-enabled Federated Learning for Network Slicing

1st Ruijie Ou
School of Computer Science and Engineering
University of Electronic Science and Technology of China, UESTC, Chengdu, China.
ORJ@uestc.edu.cn

2nd Daniel Ayepah-Mensah*
School of Computer Science and Engineering
University of Electronic Science and Technology of China, UESTC, Chengdu, China.
ayeps2000@gmail.com

3rd Guisong Liu
School of Computing and Artificial Intelligence
Southwestern University of Finance and Economics
SWUFE, Chengdu, China
gliu@swufe.edu.cn

***Abstract*—Network slicing divides wireless networks into multiple logical networks to support different applications with different performance requirements. In recent times, centralized slice controllers based on Deep Learning have been utilized to gain insight from base stations to facilitate the dynamic network slicing process. However, centralized controllers suffer from high data communication overhead due to a large amount of user data, and most network slices are unwilling to share private network dataAs a means of achieving scalable and privacy for network slices, we propose a multi-access edge-based federated learning approach for network slicing through which distributed base stations can dynamically allocate resources across multiple slices without having to share any personal or network data with a central orchestrator. The experimental results show that the proposed dynamic network slicing algorithm can dynamically allocate resources for multiple slices and satisfy the corresponding quality of service requirements.**

Keywords- Network slicing; Deep Reinforcement Learning; Resource allocation; Federated Learning; Mobile edge Computing; Privacy protection.

I. INTRODUCTION

The introduction of the fifth-generation (5G) mobile network has caused mobile data traffic growth to skyrocket. Radio Access Network (RAN) slicing is the answer to the heterogeneity and exponential growth of data transmitted over wireless cellular networks. This unprecedented growth is accompanied by an ever-increasing end-user demand for improved and faster data transmission. Network slicing enables wireless networks to adapt more quickly to different applications [1]. Different applications require efficient and dynamic resource allocation for network slices.

Deep Reinforcement learning (DRL) algorithms have been introduced to deal with the uncertain behavior of user equipment and channel conditions in wireless networks, as they can handle nonconvex and NP -hard problems [2]. In [3], the authors designed a DRL-based slice manager to organize radio resource management. In [4], the authors proposed a dynamic allocation framework where the infrastructure provider (InP) allocates unused resources to slices based on the quality of service (QoS), and the slices autonomously manage their resource allocation using DRL. Considering real-time slice requirements and spatio-temporal dynamics of traffic density in wireless networks, the authors in [5] proposed a model to predict the capacity allocated to each slice using a Deep Q-network (DQN). However, as we turn to new classes of traffic, such as autonomous vehicles, network slices tend to generate many data sets that create an overwhelming signaling overhead. The classic centralized DRL agent will inevitably face data congestion and communication delays.

Mobile edge computing, also known as MEC, has emerged as an essential part of the process known as network slicing. Because MEC is able to provide devices at the edge of a wireless network with additional computing power to meet the extensive resource requirements of base stations [6]. MEC has the ability to serve as a platform for implementing intelligent and complex resource management algorithms for RAN. Despite the fact that the MEC server is connected to the core cloud to synchronize data with the global application, the overall goal of MEC computing is to provide cloud computing capabilities to the periphery of wireless access networks. However, there are privacy concerns as some base stations are not willing to upload private and sensitive data to the cloud server.

Recently, a novel concept called federated learning (FL) has been introduced, which aims to construct intelligent systems that improve privacy through the process of federated learning. FL is a distributed collaborative approach to artificial intelligence that enables data training by coordinating numerous devices with a cloud server [7]. This is achieved without sharing the actual datasets. FL is able to address privacy concerns because each client trains its own local model based on a local training dataset. This means that the local training dataset is never sent to the cloud server, which means that FL can address privacy concerns. The authors of this paper have developed a device assignment strategy for RAN based on a hybrid FL reinforcement learning framework. The goal of this assignment scheme is to improve network throughput while increasing data security. However, the current network slicing framework does not address privacy issues associated with network slices and their users. Motivated by the scalability of MEC and privacy protection of FL, we propose an efficient and scalable MEC platform to enable a distributed resource allocation solution for network slices that jointly optimize QoS satisfaction and resource utilization of the slices. We also investigate how the number of MEC servers and the aggregation frequency of the FL model affect the QoS satisfaction and resource utilization of individual slices. The main contributions of this paper is a distributed MEC-enabled RAN slicing framework. We formulate the RAN slicing problem to optimize QoS

978-1-6654-7020-9/22/$31.00 ©2022 IEEE
DOI 10.1109/CCPQT56151.2022.00050

satisfaction and resource utilization. Furthermore, we propose a Federated DRL algorithm (FDRL) to adaptively determine the optimal fraction of RB for each slice while offering stronger privacy guarantees for BSs. Finally, we verify the effectiveness of the proposed FDRL algorithm through numerical simulations.

The rest of this paper is organized in the following manner. As part of Section II, we present the system model and the problem formulation. A proposal for FDRL RAN slicing is presented in Section III, which includes MEC-enabled FDRL RAN slicing. We present the results of the simulations in Section IV of this paper. The conclusion of this paper is presented in section IV, which concludes this paper as a whole.

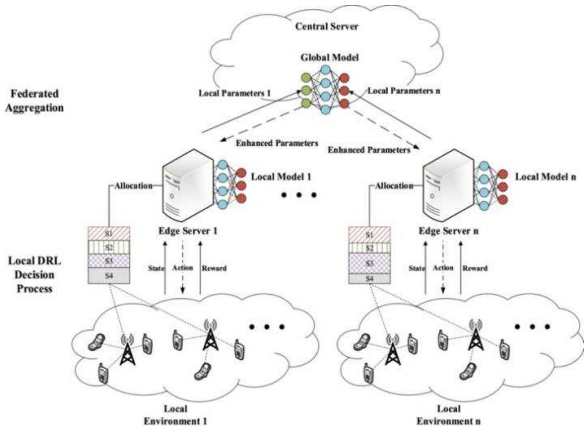


Figure 1. System Model.

II. System Model And Problem Formulation

A. Network Model

In this section, we consider a beyond 5G RAN topology under the centralized cloud server, which includes physical base stations equipped with MEC servers as illustrated in Fig. 1. Our study considers a downlink cellular network with a number of base stations $\mathcal{K} = \{1,2,3,4 \dots K\}$ in order to study its behavior. The base stations are managed by an InP . Each base station is connected to a set of users, $\mathcal{I} = \{1,2,3,4, \dots I\}$. The cellular network is split into a set $\mathcal{J} = \{1,2, \dots, J\}$ of network slices. The time is partitioned into time slots, with each time slot denoted as t. Physical Resource Blocks (PRBs) are used to denote the bandwidth of the network. Let $r_{i,t}^{j,k}$ be the transmission rate of user i which is served by slice j via base station k during time slot t, and $w_{i,t}^{j,k}$ be the number of resources i.e., bandwidth available at base station k allocates to user i which is served by slice j during time slot t. In order to calculate the transmission rate, we use Shannon's theory [9] as a guideline:

$$r_{i,t}^{j,k} = w_{i,t}^{j,k} \log_2 \left(1 + \frac{p_{i,t}^{k} \left|h_{i,t}^{k}\right|^2}{N_0}\right) \quad (1)$$

where $p_{i,t}^{k}$ is the transmit power on base station k at the t-th TTI $h_{i,t}^{k}$ is the channel coefficient. The channel coefficient is made up of path loss, shadowing effects, Rayleigh fast fading between two users, as well as path loss, shadowing and shadowing effects. N_0 is the power of additive white Gaussian noise (AWGN). The Path Loss (PL) is defined as $\text{PL (dB)} = 20\log_{10}(d) + 20\log_{10}(f) - 27.55$ where f (in MHz) and d (in meters) representing user to base station channel frequency and distance respectively [10]. Different slices can be created based on the user's requirements in order to cater to different types of traffic generated by the users. Each slice type has different QoS requirements, measured through a satisfaction function [10]. In this paper, we consider a sigmoid-based utility function such that $U\left(r_{i,t}^{j,k}\right) = \left(1 + e^{-\varphi\left(r_i - r_{i,t}^{j,k}\right)}\right)^{-1}$, where r_i is the rate requirement user i. Hence, the utility of the j th network slice at time t can be defined as:

$$\mathrm{U}_t^{j,k} = \sum_{i \in \mathcal{I}_j} w_{i,t}^{j,k} \mathrm{U}_{i,t}^{j,k}\left(r_{j,t}^{j,k}\right) \quad (2)$$

where $w_t^{j,k} = \sum_{i \in \mathcal{I}_j} w_{i,t}^{j,k}$ is the amount of RBs allocated to the slice. In addition, we measure the utilization of a slice as the ratio of used resources $w_t^{j,k}$ and the resource allocated RBs $\varphi_t^{j,k}$ to the slice as follows:

$$\Psi_t^{j,k} = \frac{w_t^{j,k}}{\varphi_t^{j,k}} \quad (3)$$

B. Federated Learning Model

Each base station has a set of local training data, denoted as $\mathcal{D}_k = \left\{x_j, y_j\right\}_{j=1}^{|\mathcal{D}_k|}, k \in \mathcal{K}$, where x_j is the data sample and y_j is the corresponding label associated with each slice on the base station. The whole training dataset is denoted as $\mathcal{D}$. A shared DRL model θ is trained by the agents on the MEC servers. Let $f\left(x_j, y_j; \theta\right)$ be the loss function associated with the data sample $\left(x_j, y_j\right)$ based on the model θ. Hence, the objective of the slicing problem can be defined as:

$$min\left\{F(\theta) \triangleq \sum_{k \in \mathcal{K}} \frac{|\mathcal{D}_k|}{|\mathcal{D}|} F_k(\theta)\right\} \quad (4)$$

where $F_k(\theta) \triangleq \frac{1}{|\mathcal{D}_k|} \sum_{j \in \mathcal{D}_k} f\left(x_j, y_j; \theta\right)$. As shown in Fig. 1, each round of the FL process starts with the cloud server sharing the global model θ, followed by the learning process at base station to obtain θ_k.

C. Problem Formulation

The dynamic network slicing problem can be defined as maximizing the utility of slices on the base station. The network slicing problem is expressed as:

$$\begin{aligned} & min \sum_{j \in J_k} \mathrm{U}_t^{j,k}\left(w_t^{j,k}\right) + \Psi_t^{j,k}\left(w_t^{j,k}\right) \\ \text{s.t.} \quad & \sum_{j \in J_k} w_t^{j,k} \leq \kappa,\ w_t^{j,k} > 0 \end{aligned} \quad (5)$$

where κ is the maximum amount of resources available on the base station, $\sum_{j\in J_k} w_t^{j,k} \leq \kappa$ indicates that the amount of the resources allocated to slice should not exceed the of total resources available on the base station and $w_t^{j,k} > 0$ is the constraints that ensure that of RB s allocated to a slice j is always positive. It should come as no surprise that resolving this issue involves solving a challenging nonlinear NP problem. As the scale of the problem grows due to the enormous dataset and the dynamic nature of 5G network slices, traditional optimization approaches for solving nonlinear programming problems will become more challenging to implement and will no longer be applicable. We propose a distributed strategy to solve the problem based on DRL and federated learning to ensure that it can be scaled.

III. MEC-ENABLED FDRL NETWORK SLICING

In this section, we transform the network slicing problem at each base station into a Markov decision problem (MDP) and design an FL approach that allows the MEC servers to collaboratively learn the network slicing decisions.

A. MDP Formulation for Network Slicing

In our MEC-enabled FL slicing system, the DRL agent deployed on the MEC server interacts environment to find an action for a state using a policy as shown in Fig. 1. To implement FDRL-based algorithms, we first have to define the state, the action, and the reward of the proposed model in order to develop the FDRL-based algorithm.

1) States: Let, s_t^k denote the state slices on base station k at time t, which is represented by the tuple $\{w_t^{j,k}, U_t^{j,k}, \Psi_t^{j,k}\}$ such that $w_t^{j,k}$, $\mathcal{U}_t^{j,k}$, and $\Psi_t^{j,k}$ denotes the resource allocation, utility, and QoS utility of each slice on the base station, respectively.

2) Actions: The action taken by the agent for the slice j is defined as a set of percentages $a_t^{j,k} = \{-0.6, -0.5 - 0.4 - 0.3, -0.2, 0, 0.2, 0.3, 0.4, 0.5, 0.6\}$. Selecting a negative action indicates a decrease in the resource and vice versa. Zero means that the slice's resource has remained the same.

3) Reward: After the state transition, the learning agent would gain rewards w.r.t. current state $s_t^{j,k}$ and actions $a_t^{j,k}$. The reward function of the agent can be defined as:

$$\mathcal{R}_t^k(s,a) = \alpha \cdot \mathrm{U}_t^{j,k}(s,a) + \beta \cdot \Psi_t^{j,k}(s,a) \quad (6)$$

where $\mathrm{U}_t^{j,k}$ and $\Psi_t^{j,k}$ respectively, denote the utility and resource utilization of the slices in the base station. α and β are the importance of the utility and resource utilization functions, respectively [11]. The ultimate objective of the MEC agent is to find the optimal slicing strategy (policy) π^*.

B. FDRL-Based Slicing Strategy

The proposed FDRL-Based slicing strategy is assumed to have T iterations for each policy update. The FDRL-based slicing strategy consist of the following: 1) local model training, 2) local model updating, and 3) Global model aggregation.

While the local model training and updates are performed on the MEC server, the central cloud is responsible for model aggregation. The aggregation period for the global model is referred to as τ.

1) .MEC local model training: In this paper, we use DQN to perform the local training [3]. As the global model is updated in the cloud, the MEC server associated with each base station will receive a copy of it so that they can update their local model as well as to evaluate the greedy policy based on weights θ_t^k as:

$$Q^*(s_t, a: \theta_t^k) := \mathbb{E}_{s_{t+1}}\left[R_t^k + \gamma \max_a Q^*(s_{t+1}, a: \theta_t^k) \mid s_t, a\right] (7)$$

Hence, the agent learns the optimal policy by minimizing the mean-squared error (MSE) loss function, which is defined as:

$$\mathcal{L}_k(\theta_t^k) := \left(R_t^k + \gamma \max_a \mathrm{Q}(s_{t+1}, a; \theta_t) - Q(s_t, a; \theta_t^k)\right)^2 (8)$$

2) Local Model Update: In the t-th iteration, the mini-batch stochastic gradient algorithm is used in each MEC agent to perform a model update in accordance with its local data, and is based on the following expression:

$$\theta_t^{(k)} \leftarrow \theta_{t-1}^{(k)} - \eta \nabla f_k\left(\theta_{t-1}^{(k)}\right) \quad (9)$$

where η is the learning rate, and where $f_k(\theta) = \frac{1}{|D_k|}\sum \mathcal{L}_k(\theta^k)$ is the loss function for the agent k, $\theta_{t-1}^{(k)}$ is the local model available at kth BS at the start of the t-th iteration.

3) Model Aggregation: Each MEC server uploads the latest local model to the central cloud server in the t-th iteration, and the cloud server aggregates the local models from each MEC server according to the following expression as the t-th iteration is completed:

$$\theta \leftarrow \sum_{k\in\mathcal{K}} \frac{|\mathcal{D}_k|}{|\mathcal{D}|}\theta_t^{(k)} \quad (10)$$

where θ denotes the global model at the cloud server. After that, the MEC servers will download θ. The perform an update on θ an $\theta_t^{(k)} \leftarrow \theta$ which will initialize the global model as the local model. Based on the global model, the MEC agent for the base station determines the percentage of RB for each slice decision epoch. The dynamic network slicing algorithm is summarized in Algorithm 1, where the complexity is $\mathcal{O}(\tau(|\mathcal{K}|))$ because each aggregation period includes the computation of local model updates on the base station, where τ and $|\mathcal{K}|$ are the number of aggregation periods and number of base stations, respectively.

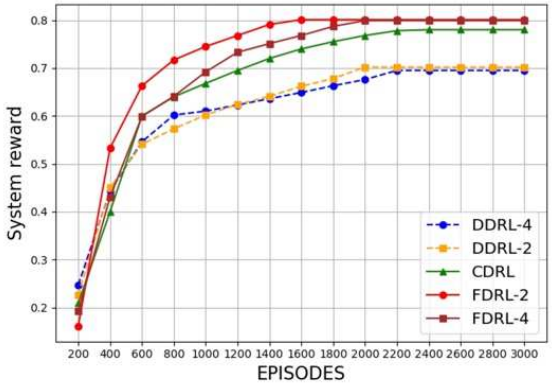
IV. SIMULATION RESULTS

This section aims to validate the proposed FDRL method for network slicing. Based on [10], the simulation setup has been developed. Network resources are distributed over a grid of OFDM-based resource blocks with 50 RBs. The carrier frequency is 1 GHz, and the total bandwidth of the system is 5 MHz. Assume a coverage area of approximately 150 meters with four uniformly distributed base stations. Each base station is capable of transmitting a maximum of 30 dBm. It is estimated that the spectral density of additive white Gaussian noise (AWGN) is -174 dBm/Hz. Regarding the 5G slicing classes, we define four slices: HDTV, UEb, MIoT, and ULL. MEC servers are regarded as clients within the context of FL to train their DRL agents individually. An input layer, an output layer, and three fully interconnected layers make up the DQN. The DQN consist of hidden layers with 256, 128, and 120 neurons, respectively. In each layer, the activation function ReLu is used. A discount factor of 0.95 is set for the discount rate, and a learning rate of 0.01 is set for the discount factor.

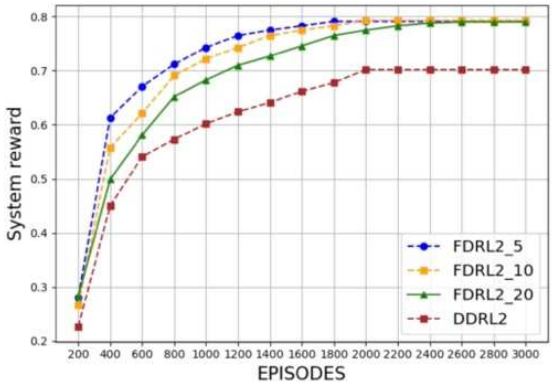
Algorithm 1: MEC-enabled FDRL Network Slicing

Initialize all MEC servers with the same model, i.e., $\theta^{(k)} = \theta, k \in \mathcal{K}$.;
For $(t = 1,2, \dots, T)$:
local model training;
For (each MEC servers $k \in \mathcal{K}$ in parallel):
    Make a random choice a^t with probability ϵ;
    If an action is not found, select action a^t satisfying $a_t = {}_{a_t \in A}\ \text{argmax} Q(s_t, a_t)$;
    Update the resource fraction $w_t^{j,k}$ and observe reward R_t^k and new state s^{t+1};
    Store experience $\langle s^t, a^t, r^t, s^{t+1} \rangle$;
    From the replay memory $\mathcal{D}$, select a mini-batch of experiences for training.
    Update $\theta_t^{(k)}$ based on to (9);
        Update $\theta_t^{(k)}$ as θ
End for;
Update global model;
If $(\text{mod}(t, \tau) = 0)$
    Collect the most updated model from the MEC servers;
    Obtain θ by performing model aggregation according to (10);
    Broadcast θ to all MEC servers;
End If;
End for;

We choose a replay buffer size of 1000 samples and a mini-batch size of 64 samples. The federated learning consists of 3000 rounds of training. In each epoch, we increased the number of users by ten users so that we could analyze the scalability of the algorithm. Each time the averaged parameters of the model are updated, a final average is calculated over the training period of the corresponding learning model. Our simulations were run in a Python 3.7 environment, and TensorFlow 2.0 was used to implement DRL, and FL [12]. We run all simulations on a workstation with an Intel Core i7-9700k CPU and an NVIDIA GeForce RTX 3090/3080 graphics card running on a Windows operating system running on a workstation. An overview of the parameter settings that were used in this study can be found in Table I. There are four benchmark schemes that are used in the simulation:



(a) Performance analysis for different algorithms



(b) Performance analysis for aggregation periods.

Figure 2. Algorithm comparison and performance analysis with various aggregation periods

- Disturbed DRL 4 agents (DDRL-4): In this method, there are four agents. Each agent optimizes its resource allocation policy for slices on the base station without FL.
- Disturbed DRL 2 agents (DDRL-2): In this method, only two agents are deployed in the network, with each agent controlling two base stations. Each agent is responsible for optimizing the resource allocation policy of the base stations assigned to it without FL.
- Centralized DRL (CDRL): This algorithm consists of a centralized agent with a global view of the network and performs resource allocation decisions for all base stations.
- Federated DRL 4 Agents (FDRL-4): In this method, there are four agents, i.e., each base station has its agent. The agents collaborate through FedAvg to optimize resource allocation policy for all base stations.
- Federated DRL 2 Agents (FDRL-4): In this method, only two agents are deployed in the network, with

each agent controlling two base stations. In our proposed method,

TABLE I. SIMULATION PARAMETERS

Parameters	Values
Number of base stations	4
Number of slices	4
System Bandwidth	5MHz
Transmitting power	30dBm
BS coverage	150meters
Distance between BS	120meters
Channel gain	To be generated via path loss model
Noise power density	-174dBm/Hz
User distribution	Uniform
No. of users/slice	UEb: [12,60], Hdtv:[1,12] MIoT:[100,240], ULL:[68, 124]
Packet arriving rate/user	[UEb: 100, Hdtv:100, MIoT:100, ULL:100](packet/second);
Minimum rate demand	[UEb:100,Hdtv:500, MIoT:12, ULL:10]
Maximum delay demand	[UEb:100, Hdtv:120, MIoT:105, ULL:10] (ms)
The packet size	[UEb:400, Hdtv:4000, MIoT:500, ULL:120](bits)
Packet arriving time	ULL: uniform, Others: exponential distribution
Aggregation period	1episode, 5episodes, 10episodes, 20episodes
Learning rate	0.01
Discount number γ	0.95

MEC servers group base stations with similar channel conditions and provide each group with a personalized resource allocation policy for all slices in the network. Fig. 2a compares the system reward produced by different algorithms. The system reward is defined as the sum of local rewards achieved, which is the numerical feedback each agent receives after making a resource allocation decision. It can be observed that the Federated DRL 2 agents (FDRL-2) and FDRL-4 agents achieve the highest system reward. To further improve the performance of the proposed Federated DRL 2 Agents (FDRL-2) and FDRL-4, we propose that they utilize extra knowledge by leveraging other MEC agents to facilitate the training process and therefore facilitate the training process. At the same time, we find that the proposed Federated DRL 2 Agents (FDRL-2) performs better than FDRL-4 between episodes 5 and 10 dues to its faster convergence speed. Having two agents instead of 4 agents can reduce the frequency of edge aggregation, allowing the learning model to converge with lower communication costs. The centralized approach converges relatively quickly due to the perfect completeness of the information. However, as the number of users in the network increases, the performance decreases due to the large data set and communication overhead. Each base station agent in the Disturbed DRL, with four agents and two agents, trains its model based on its observations and actions without communicating local data with the other agents. However, exchanging the parameters of local models among the base stations' multiple agents might increase the local model's training performance and therefore achieve higher system utility. The convergence of Federated DRL 2 agents (FDRL-2) indicates that base station agent collaboration in the training process may effectively overcome the intrinsic non-stationarity in a dynamic environment of a wireless network, contributing to system performance enhancement.

Furthermore, we study the effect of different aggregation periods on the FDRL-2 algorithm on system reward in Fig. 2. Fig. 2 shows the convergence of system reward during training tasks in four different configurations, i.e., an aggregation period of 1 episode (FDRL2-1), five episodes (FDRL2-5), ten episodes (FDRL2-10), and 20 episodes (FDRL2-20). We compare the configuration using DDRL with two agents as the benchmark. As we can see, the performance with one episode and five episodes is practically identical. The system reward performance of the DDRL agent is lower than that of FDRL in all configurations. The results indicate the advantage of a shorter aggregation period. With shorter aggregation periods, the learning model can capture the dynamic changes in the network and maximize the system reward. Finally, Fig. 3 shows the scatter plot of throughput and delay for each slice. For the UEb, MIoT, and HDTV slices, each point in the scatter plot represents each user's throughput per epoch. The point in the ULL slice represents the delay of the users in each epoch. From Fig. 3, the throughput of CDRL users ranges between 110 and 116 kbps for UEb and between 28 and 29 kbps for MIoT. The FDRL-2 algorithm achieves a wide range of throughputs in the UEb and MIoT slices. In the HDTV slice, the agents for CDRL and FDRL achieve a nearly constant throughput of 500-520 kbps per episode. In the ULL slice, we find that users in the CDRL algorithms achieve relatively higher delays, i.e., between 25 and 40 ms, while users in FDRL-2 achieve relatively lower delays in the ULL slice. The wide range of user throughputs in UEb and MIoT is because FDRL-2 can allocate resources dynamically due to each agent's higher model training accuracy.

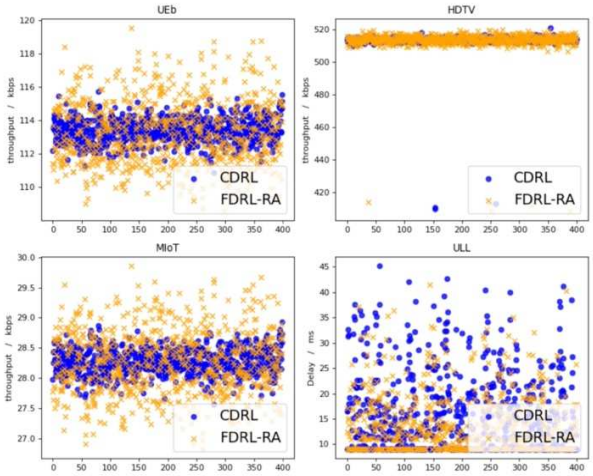


Figure 3. Performance analysis for federated and centralized slicing after convergence.

V. CONCLUSION

This research aims to develop a framework for distributed network slicing that integrates FL and MEC. FL and MEC

enable a distributed resource allocation solution for network slices, simultaneously optimizing the compliance of slice QoS requirements and the consumption of available resources. We turn the problem of network slicing at each base station into a learning task and design a federated DRL approach that enables MEC servers to learn together to determine the best strategies for each network slice. This allows us to solve the problem of dynamic network slicing. During the simulation, we investigated how the quantity of MEC servers and the frequency of FL model aggregation affect the level of QoS satisfaction and resource utilization across all slices. The findings of the experiments indicate that the proposed MEC-enabled FDRL can reduce service latency and increase throughput for users while maintaining the confidentiality of each slice.

ACKNOWLEDGMENT

This study was funded in part by grants from the Natural Science Foundation of China (Nos. 61806040 and 61771098); the Department of Science and Technology in Sichuan Province (No. 2020YFQ0025); the Natural Science Foundation of Guangdong Province (No. 2021A1515011866); and the Intelligent Terminal Key Laboratory of Sichuan Province (No. 2020YFQ0025).

REFERENCES

[1] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5g: Survey and challenges," IEEE Communications Magazine, vol. 55, no. 5, pp. 94–100, 2017.

[2] M.R.Raza,C.Natalino,P.O ̈hlen,L.Wosinska,andP.Monti, "Reinforcement learning for slicing in a 5g flexible ran," Journal of Lightwave Technology, vol. 37, no. 20, pp. 5161– 5169, 2019.

[3] X. Zhang, W. Lu, B. Li, and Z. Zhu, "Drl-based network orchestration to realize cooperative, distributed and tenantdriven virtual network slicing," in 2019 Asia Communications and Photonics Conference (ACP), 2019, pp. 1–3.

[4] G.Sun,Z.T.Gebrekidan,G.O.Boateng,D.Ayepah-Mensah, and W. Jiang, "Dynamic reservation and deep reinforcement learning based autonomous resource slicing for virtualized radio access networks," IEEE Access, vol. 7, pp. 45758– 45 772, 2019.

[5] S. Wang, X. Chai, X. Song, and X. Liang, "Deep q-learning enabled wireless resource allocation for 5g network based vehicle-to-vehicle communications," in 2021 IEEE 6th International Conference on Signal and Image Processing (ICSIP), 2021, pp. 903–907.

[6] Z. Wang, R. Gu, G. Zhang, T. Zhao, Y. Wang, Y. Wang, and Y. Ji, "Demonstration of network slicing in mobile edge computing service migration," in 2018 Asia Communications and Photonics Conference (ACP), 2018, pp. 1–3.

[7] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in IEEE INFOCOM 2019 IEEE Conference on Computer Communications, 2019, pp. 1387–1395.

[8] Y.-J. Liu, G. Feng, Y. Sun, S. Qin, and Y.-C. Liang, "Device association for ran slicing based on hybrid federated deep reinforcement learning," IEEE Transactions on Vehicular Technology, vol. 69, no. 12, pp. 15 731–15 745, 2020.

[9] B. Dai and W. Yu, "Energy efficiency of downlink transmission strategies for cloud radio access networks," IEEE J. Sel. Areas Commun., vol. 34, no. 4, pp. 1037–1050, 2016.

[10] K. Xiong, S. Samuel Rene Adolphe, G. O. Boateng, G. Liu, and G. Sun, "Dynamic resource provisioning and resource customization for mixed traffics in virtualized radio access network," IEEE Access, vol. 7, pp. 115 440–115 453, 2019.

[11] G. Sun, G. O. Boateng, D. Ayepah-Mensah, G. Liu, and J. Wei, "Autonomous resource slicing for virtualized vehicular networks with d2d communications based on deep reinforcement learning," IEEE Systems Journal, vol. 14, no. 4, pp. 4694–4705, 2020.

[12] M. Abadi, A. Agarwal, P. Barham, and E. Brevdo, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/