Deep Reinforcement Learning for Energy-Efficient Power Control in Heterogeneous Networks

Jianhao Peng†, Jiabao Zheng†, Lin Zhang†, and Ming Xiao†*
† University of Electronic Science and Technology of China (UESTC)
* School of Electrical Engineering, Royal Institute of Technology (KTH)

ICC 2022 - IEEE International Conference on Communications | 978-1-5386-8347-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICC45855.2022.9839235

***Abstract*—In a typical heterogeneous network (HetNet), in which a macro base station (BS) and multiple small BSs coexist on the same spectrum band, energy-efficiency (EE) performance is an important design metric and is highly related to the transmit power of BSs. Conventional methods optimize BSs' transmit power to enhance the EE by assuming that the global channel state information (CSI) is available. However, it is challenging or expensive to collect the instantaneous global CSI in the HetNet. In this paper, we utilize deep reinforcement learning (DRL) technique to design an intelligent power control algorithm, with which each BS can independently determine the transmit power based on only local information. Simulation results demonstrate that the proposed algorithm outperforms conventional methods in terms of both EE performance and time complexity.**

I. Introduction

With the rapid development of mobile communications, wireless data traffic increases explosively and it becomes challenging for conventional cellular networks to accommodate the ever-growing wireless data traffic. To enhance the network throughput, *heterogeneous network* (HetNet) is proposed by deploying small *base stations* (BSs) as complementary to the conventional macro BS [1]. However, more BSs will result in higher energy consumption. Therefore, to cope with the contradiction between the throughput improvement and increased energy consumption, *energy efficiency* (EE), which indicates the average throughput of unit energy consumption, becomes a main design metric in the HetNets [2].

It is clear that the EE maximization problem of a single communication link between a BS and a user can be efficiently solved by the traditional *fractional programming* (FP) theory [3]. However, multiple BS-user links may reuse the same wireless channel in a HetNet, and cause significant co-channel interference to each other. This makes the EE maximization problem in the HetNet much more complicated.

To solve the EE maximization problem in HetNets, extensive studies have been conducted in these years. In particular, it is observed that the transmit power of BSs has great influences on the EE performance. Then, [4] proposed a *sequential fractional programming* (SFP) algorithm for the joint power control for BSs and achieved near-optimal EE performance. [5] proposed a power control algorithm based on the branch-and-bound method, which can maximize the common EE. However, both algorithms need to first collect the instantaneous global *channel state information* (CSI) of the whole network and then use the global CSI to optimize the joint transmit power design. In other words, both algorithms need a long time to design proper transmit power and thus lead to a short time for data transmissions, which compromises the system performance in practical scenarios.

Recently, machine learning technology in the computer science field has been suggested to solve the EE maximization problem. For instance, [6] considered a cellular network, and generated data sets with the SFP algorithm and used the data sets to train an *artificial neural network* (ANN) for the power control. It is shown that the well-trained ANN can achieve a *global energy efficiency* (GEE) performance similar to the SFP algorithm. To avoid generating data sets in advance, [7] and [8] adopted the *deep reinforcement learning* (DRL) technique and showed the great potential of DRL for the energy-efficient power control. In particular, [7] used the deep Q-learning for the transmit power design in a HetNet with multiple wireless channels, and [8] used the deep Q-learning, double deep Q-learning, dueling deep Q-learning for the transmit power design in a HetNet with multiple device-to-device links.

However, most of the related literatures directly apply the existing DRL algorithms which are originally designated for the problems (e.g., natural language processing and pattern recognition) in the computer science field for the energy-efficient power control in HetNets, and ignore unique features of wireless communications. This may lead to sub-optimal EE performance.

In this paper, we consider a typical HetNet in which a macro BS and multiple small BSs reuse a common spectrum band. Specifically, we notice that each BS in the edge can configure the transmit power in a real-time manner, and the core network in the cloud has redundant historical data which contains useful knowledge for globally energy-efficient power control. Then, it is straightforward to enable the cloud to use the available historical data for training energy-efficient power control models, with which edge BSs can directly determine proper transmit power to enhance the GEE performance. For this purpose, we develop a cloud-edge collaborative framework and propose an intelligent power control algorithm for BSs by extending the existing policy-based DRL algorithm, i.e., *deep deterministic policy gradient* (DDPG). With the proposed algorithm, each BS can independently determine the transmit power based on only local information. Simulation results show that the proposed algorithm outperforms the SFP algorithm in terms of both GEE performance and time complexity.

The corresponding author is Lin Zhang, email: linzhang1913@gmail.com.

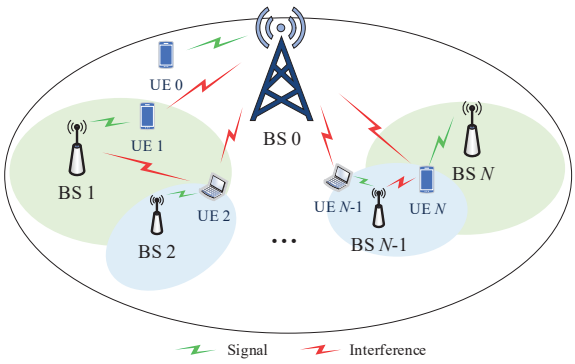


Figure 1. A typical downlink HetNet, in which a macro BS and multiple small BSs reuse the same spectrum band and may cause interference to each other.

II. System Model and Problem Description

As shown in Fig. 1, we consider a downlink HetNet, in which the macro BS is responsible to provide radio services for the whole macro cell and N small BSs are deployed to enhance the radio access for relatively small regions. Both the macro BS and small BSs reuse the same spectrum band and may inevitably cause interference to each other. We index a BS and its served *user equipment* (UE) as BS n and UE n, $n \in \mathbb{N} = \{0, 1, 2, \cdots, N\}$, respectively. In the following, we provide the system model and problem description.

A. System model

The wireless channel between a BS and a UE is composed of large-scale attenuation (path-loss and shadowing) and small-scale block Rayleigh fading. In particular, the large-scale attenuation is highly related to the locations of the BS and UE, while the small-scale block Rayleigh fading is a random variable, which typically remains constant in a single time slot and varies among different time slots. By denoting $\phi_{n,k}$ as the large-scale attenuation and denoting $h_{n,k}$ as the small-scale block Rayleigh fading from BS n to UE k, the corresponding channel gain can be expressed as $g_{n,k} = \phi_{n,k}|h_{n,k}|^2$.

If we denote $p_n(t)$ as the transmit power of BS n in time slot t, the measured *signal to interference and noise ratio* (SINR) at UE n is

$$\gamma_n(t) = \frac{p_n(t)g_{n,n}(t)}{\sum_{k\in\mathbb{N},k\neq n} p_k(t)g_{k,n}(t) + \sigma^2}, \tag{1}$$

where σ^2 is the noise power at UEs. Then, the downlink achievable rate from BS n to UE n in time slot t is

$$r_n(t) = B\log\left(1+\gamma_n(t)\right), \tag{2}$$

where B is the spectrum bandwidth. Accordingly, the GEE of the HetNet in time slot t can be expressed as [9]

$$\mathrm{GEE(t)} = \frac{B\sum_{n=1}^{N}\log_2\left(1+\gamma_n(t)\right)}{p_c + \sum_{n=1}^{N}\psi_n p_n(t)}, \tag{3}$$

where ψ_n is the reciprocal of the power amplifier efficiency of BS n and p_c is the total circuit power.

B. Problem description

From (3), the GEE is highly related to the transmit power of each BS. We aim to optimize the transmit power of each BS and maximize the GEE in each time slot, i.e.,

$$\begin{aligned} \max_{p_n(t),\forall n\in\mathbb{N}} \quad & \mathrm{GEE(t)} \\ \text{s.t.} \quad & 0 \le p_n(t) \le p_{n,\max}, \forall n \in \mathbb{N}, \end{aligned} \tag{4}$$

where $p_{n,\max}$ is the maximum transmit power constraint of BS n. It should be noted that different BSs may have different maximum transmit power constraints in a typical HetNet.

To solve problem (4), conventional methods usually assume that instantaneous global CSI can be utilized. In practical situations, it is challenging or expensive to obtain the instantaneous global CSI. Alternatively, we will develop a DRL-based energy-efficient power control algorithm.

III. Proposed DRL-based Energy-efficient Power Control Algorithm

In this section, we will develop an energy-efficient power control algorithm by enabling the collaboration between the cloud and edge BSs. In the following, we first develop a cloud-edge collaborative framework and then elaborate the detailed algorithm.

A. Cloud-edge collaborative framework

We notice that the core network in the cloud has redundant historical data of the whole HetNet, and edge BSs can configure the transmit power in a real-time manner. To fully utilize the advantages in both cloud and edge, we enable the core network to collaborate with edge BSs and design a cloud-edge collaborative framework as shown in Fig. 2. With the framework, the cloud can utilize the historical data of the whole HetNet to train energy-efficient power control models for edge BSs, which then use the well-trained models to configure proper transmit power based on only local information to enhance the GEE. In other words, each edge BS does not need to exchange instantaneous information with other edge BSs and configure its transmit power independently.

In the edge, each BS acts as an agent and is equipped with a DNN, namely, edge DNN, which is responsible to determine a proper transmit power for BS n based on local information. Thus, the input and output of edge DNN n are the local information and the corresponding transmit power, respectively.

To achieve on-line training in the cloud, each edge BS uploads an edge experience (i.e., local measurements) to the cloud in each time slot through a backhaul link. By automatically integrating edge experiences, the cloud can construct global experiences and store them in a global experience replay buffer, which has a limited storage capacity and works in a first-in-first-out manner.

Note that, different edge BSs may have unique features, and thus different edge DNNs may have different optimal parameters. To train optimal parameters for different edge DNNs, a dedicated DNN, namely, actor DNN n ($n \in \mathbb{N}$) is

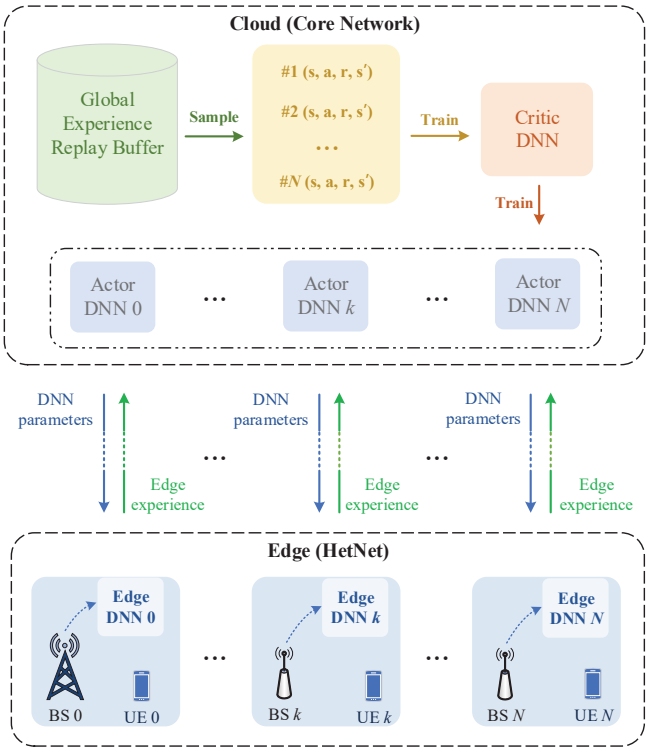


Figure 2. Cloud-edge collaborative framework.

established in the cloud to associate with edge DNN n and has the same structure (e.g., number of input ports, number of layers, number of output ports) as edge DNN n. Meanwhile, a critic DNN is established in the cloud to evaluate the goodness of transmit power configurations of the whole HetNet in terms of the GEE and guide the training of the $N+1$ actor DNNs separately. In this way, the well-trained parameters of actor DNN n can be transmitted to edge BS n and directly used to replace the parameters of edge DNN n. Since the evaluation of the critic DNN contains information of the whole HetNet, using the evaluation of the critic DNN to guide the training of each actor DNN enables the parameters of actor DNNs to converge to the global optimum. Thus, the input of the critic DNN includes the historical radio information and the corresponding transmit power configurations of the whole HetNet, and the output is the long-term GEE.

B. Designs in the edge

According to the proposed cloud-edge collaborative framework, each edge BS needs to determine a proper transmit power based on only local information. Thus, the action of edge DNN n in time slot t is $a_n = p_n$ and the state of edge DNN n in time slot t is designed as (5) at the top of the next page, which includes the channel gain between edge BS n and UE n, the transmit power of edge BS n, the received interference, the received SINR, the achievable rate from edge BS n to UE n in the previous time slot, i.e., $g_{n,n}(t-1)$, $p_n(t-1)$, $\sum_{k\in\mathbb{N},k\neq n} p_k(t-1)g_{k,n}(t-1)$, $\gamma_n(t-1)$, $r_n(t-1)$, as well as the channel gain between edge BS n and UE n in the current time slot, i.e., $g_{n,n}(t)$, the received interference in the beginning of the current time slot before configuring new transmit power, i.e., $\sum_{k\in\mathbb{N},k\neq n} p_k(t-1)g_{k,n}(t)$, and the reciprocal of the power amplifier efficiency of edge BS n, i.e., ψ_n.

Accordingly, the edge experience at BS n in time slot can be constructed as a set including the state-action pair in the current time slot, i.e.,

$$e_n(t) = \{s_n(t), a_n(t)\}. \tag{6}$$

The edge DNN structure is designed as a fully connected NN as shown in Fig. 3, in which the input layer has eight ports corresponding to eight elements in the designed state and the output layer has one port corresponding to the designed action.

C. Designs in the cloud

Since there usually exists a transmission latency from edge BSs to the cloud, we design the global action in the cloud at time slot t as $\boldsymbol{a}(t) = \{a_0(t-T_d), \cdots, a_N(t-T_d)\}$, and design the global state in the cloud at time slot t as $\boldsymbol{s}(t) = \{s_0(t-T_d), \cdots, s_N(t-T_d), s_g(t-T_d)\}$, where $s_g(t-T_d)$ is the channel gain matrix of the HetNet at time slot $t-T_d$.

Note that, it is possible for the cloud to construct the channel gain matrix $s_g(t-T_d)$. In particular, $s_g(t-T_d)$ consists of channel gain $g_{n,n}(t-T_d)$ and interference channel gain $g_{n,k}(t-T_d)$ $(n \in \mathbb{N}, k \in \mathbb{N}, n \neq k)$. Since the channel gain $g_{n,n}(t-T_d)$ is available in $s_n(t-T_d)$, we focus on the interference channel gain $g_{n,k}(t-T_d)$. At the beginning of time slot $t-T_d$, the new transmit power has not been configured although channel has changed, all BSs send orthogonal pilot sequences with the transmit power in the previous time slot to corresponding UEs for channel and interference estimation. Then, UE n can measure the received pilot signal strength $p_k(t-1-T_d)g_{k,n}(t-T_d)$ from BS k and then UE n can transmit it to BS n, which will forward $p_k(t-1-T_d)g_{k,n}(t-T_d)$ to the cloud as the auxiliary information of the edge experience. Then, the cloud can combine $p_k(t-1-T_d)g_{k,n}(t-T_d)$ and $p_k(t-1-T_d)$ to obtain $g_{k,n}(t-T_d)$. In this way, the interference channel gain $g_{n,k}(t-T_d)$ can be obtained, and used to construct channel gain matrix $s_g(t-T_d)$.

Then, we design the global reward in the cloud in time slot t as a function of GEE, i.e.,

$$R(t) = f(GEE(t-T_d)), \tag{7}$$

where $f(\cdot)$ is an increasing function and is used to scale up the value of GEE, such that even a small improvement of GEE can be encouraged. Based on the above designs, the global experience in the cloud in time slot t can be constructed as a set including the global state-action-reward pair in time slot $t-1-T_d$ and the global state in time slot $t-T_d$, i.e.,

$$\begin{aligned} E(t) = \{&\boldsymbol{s}(t-1-T_d), \boldsymbol{a}(t-1-T_d), \\ &R(t-1-T_d), \boldsymbol{s}(t-T_d)\}. \end{aligned} \tag{8}$$

As aforementioned, $N+1$ actor DNNs and one critic network are established in the core network. The actor DNN

$$s_n(t) = \{g_{n,n}(t-1), p_n(t-1), \sum_{k\in\mathbb{N},k\neq n} p_k(t-1)g_{k,n}(t-1), \gamma_n(t-1), r_n(t-1), g_{n,n}(t), \sum_{k\in\mathbb{N},k\neq n} p_k(t-1)g_{k,n}(t), \psi_n\}. \quad (5)$$

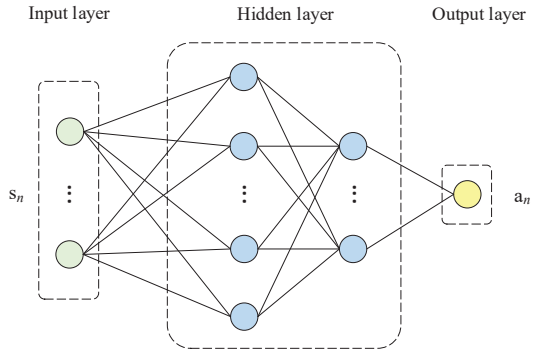


Figure 3. The structure of each edge/actor DNN.

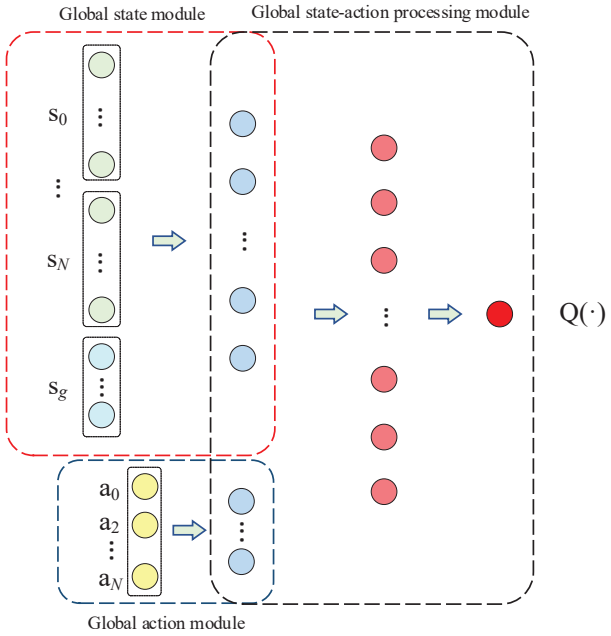


Figure 4. The structure of the critic DNN.

n has the same structure as edge DNN n, while the structure of the critic DNN is designed in Fig. 4. In particular, the input of the critic DNN is composed of three fully-connected NN modules, i.e., global state module, global action module, and the global state-action processing module, which are illustrated by dotted boxes with different colors. The global state module inputs the global state $\boldsymbol{s}$, the global action module inputs the global action $\boldsymbol{a}$ and the global state-action processing module outputs the evaluation (i.e., long-term GEE or Q value) of the global state-action pair. Note that the last layers of the global state and action modules are concatenated as the first layer of the global state-action processing module.

D. DNN training algorithm

There exist $N+1$ actor DNNs and one critic DNN to be trained in the cloud, and $N+1$ edge DNNs to be trained in the edge. In particular, we will extend the existing DDPG algorithm in [10] to train the DNNs in the cloud concurrently and train edge DNNs by using the parameters of actor DNNs to replace the corresponding edge DNNs.

We denote respectively the critic DNN, actor DNNs, and edge DNNs as $Q\left(\boldsymbol{s},\boldsymbol{a};\theta^{(\mathrm{c})}\right)$, $\mu_n^{(\mathrm{a})}\left(s_n;\theta_n^{(\mathrm{a})}\right)$, and $\mu_n^{(\mathrm{e})}\left(s_n;\theta_n^{(\mathrm{e})}\right)$ ($n\in\mathbb{N}$), where $\theta^{(\mathrm{c})}$, $\theta_n^{(\mathrm{a})}$, and $\theta_n^{(\mathrm{e})}$, are the parameter vectors of critic DNN, actor DNN n and edge DNN n, respectively. To stabilize the training of the critic DNN and actor DNNs, we create a target critic DNN for the critic DNN, and denote it as $Q^{-}\left(\boldsymbol{s},\boldsymbol{a};\theta^{(\text{c-})}\right)$. Also, we create a target actor DNN n for each actor DNN n, and denote them as $\mu_n^{(\text{a-})}\left(s_n;\theta_n^{(\text{a-})}\right)$. Note that, the parameter vectors of the critic DNN, actor DNNs, and edge DNNs will be first initialized randomly, and then the parameter vectors of the critic DNN and actor DNNs will be used for the initializations of the corresponding target critic DNN and target actor DNNs.

Suppose that, a batch of D global experiences in the cloud will be used for each training. To trigger the training process in the cloud, the cloud needs to accumulate D global experiences. Thus, all the edge BSs firstly select random transmit power for downlink transmissions at the beginning of each time slot t and conduct local measurements. Once the parameter vectors of edge DNNs are updated, edge BSs use edge DNNs to generate transmit power based on local information, i.e., $p_n(t)=\mu_n^{(\mathrm{e})}(s_n;\theta_n^{(\mathrm{e})})+\mathcal{N}(t)$, where $\mu_n^{(\mathrm{e})}(s_n;\theta_n^{(\mathrm{e})})$ is the output of edge DNN n, and $\mathcal{N}(t)$ is the zero-mean action noise to explore better transmit power in the training stage. In this way, each edge BS can continuously obtain new edge experiences based on local measurements and upload them to the cloud.

We design the action noise variance ζ to decay at a fixed rate λ as the time slot increases. Then, the action noise variance at time slot t can be expressed as

$$\zeta = \zeta_{end} + (\zeta_{ini} - \zeta_{end})e^{-\lambda t}, \quad (9)$$

where ζ_{ini} and ζ_{end} are the initial and ending values, respectively.

In each training, D samples are randomly picked from the global experience replay buffer to update the parameter vectors of the critic DNN and actor DNNs.

1) Training of critic DNN: For the i-th sampled global experience in the form of (8), we re-denote it as $E_i = \{\boldsymbol{s_i},\boldsymbol{a_i},R_i,\boldsymbol{s_i'}\}$. Then, the target Q-value, i.e., long-term GEE, y_i^{Tar} in terms of i-th sample can be written in a temporal-difference form, i.e.,

$$y_i^{\mathrm{Tar}} = R_i + \eta Q^{-}(\boldsymbol{s}_i',\boldsymbol{a}_i';\theta^{(\text{c-})}), \quad (10)$$

where $\eta \in [0,1]$ is the discount factor. By using *Mean Squared Error* (MSE) method to evaluate the difference, i.e., loss function, between the predicted long-term GEE and the target Q-value of the sampled D global experiences as

$$\mathbb{L}(\theta^{(\mathrm{c})})=\frac{1}{D}\sum_{i}\left[y_i^{\mathrm{Tar}}-Q(\boldsymbol{s}_i,\boldsymbol{a}_i;\theta^{(\mathrm{c})})\right]^2, \tag{11}$$

we can adopt the gradient descent method to update the parameter vector $\theta^{(c)}$ of the critic DNN and minimize the loss function in (11).

2) Training of actor DNNs: Since the output transmit power of each edge DNN aims to maximize the GEE of the whole HetNet, the training of each actor DNN is designed to update its parameter vector in the direction of maximizing the expected Q value of the critic network, which can be expressed in (12) at the top of the next page.

Accordingly, the update rule of the parameter vector of the actor DNN is $\theta_n^{(\mathrm{a})} = \theta_n^{(\mathrm{a})} - \nabla_{\theta_n^{(\mathrm{a})}} J(\theta_1^{(\mathrm{a})},\cdots,\theta_N^{(\mathrm{a})})$, where $\nabla_{\theta_n^{(\mathrm{a})}} J(\theta_1^{(\mathrm{a})},\cdots,\theta_N^{(\mathrm{a})})$ is the partial derivation of the expected Q value in terms of $\theta_n^{(a)}$ and can be expressed in (13) at the top of the next page.

3) Training of target critic DNN and target actor DNNs: Similar to [10], we adopt the soft update method to update the target critic DNN and the target actor DNNs as follows,

$$\theta^{(\text{c-})} \leftarrow \tau^{(\mathrm{c})}\theta^{(\mathrm{c})} + (1-\tau^{(\mathrm{c})})\theta^{(\text{c-})}, \tag{14}$$

$$\theta_n^{(\text{a-})} \leftarrow \tau^{(\mathrm{a})}\theta_n^{(\mathrm{a})} + (1-\tau^{(\mathrm{a})})\theta_n^{(\text{a-})}. \tag{15}$$

Finally, the parameter vectors of edge DNNs are replaced by those of the associated actor DNNs. To reduce the communication overheads, we design to update the parameter vectors of edge DNNs every T_u time slots until convergence.

IV. Simulation results

In this section, we provide simulation results to evaluate the performance of the proposed algorithm. Also, we consider the SFP algorithm in [4], random power algorithm, and full power algorithm for comparisons.

A. Simulation settings

To begin with, we provide the hyperparameters of the DNNs. Each edge/actor DNN has four fully-connected layers, including an input layer with eight neurons corresponding to eight elements in the state design (5), and two hidden layers each with 100 neurons, and one output layer with one neuron corresponding to the action design of each edge BS. The learning rate is set to 0.0001, the initial value of the noise variance $\zeta_{ini} = 1$, the ending value is $\zeta_{end} = 0.5$, and the decay rate λ is 0.00125.

For the critic DNN, the global state module has three fully-connected layers including an input layer with $8(N+1)+(N+1)^2$ neurons corresponding to the elements in the global state design, and two hidden layers each with 200 neurons. The global action module has two layers including an input layer with $N+1$ neurons corresponding to the elements in the global action design and a hidden layer with 200 neurons. The global state-action processing module has three layers including an input layer with 400 neurons, and one hidden layer with 200 neurons, and an output layer with a neuron corresponding to the Q value. The learning rate is set to 0.001, the batch size D is 128, and the discount factor η is 0.5.

In our simulation, we consider a two-layer HetNet scenario, in which five BSs are deployed. The first layer contains only BS 0, located at coordinates $(0,0)$, with a maximum transmit power constraint of 30 dBm, covering a disc area with a minimum radius of 10 meters and a maximum radius of 1000 meters. BS 1 to BS 4 are distributed in the second layer, located at coordinates $(500,0)$, $(0,500)$, $(-500,0)$, $(0,-500)$, each with a maximum transmit power constraint of 23 dBm, covering a disc area with a minimum radius of 10 meters and a maximum radius of 200 meters, respectively. The UEs are randomly distributed within the coverages of corresponding BSs.

Besides, we set $\psi_n = 10$ for all $n \in \mathbb{N}$ and $p_c = 30$ dbm, bandwidth $B = 10$ MHz. The path-loss is modeled by $120.9 + 37.6\log 10(d)$ in dB, where d represents the distance (in kilometers), the log-normal shadowing standard deviation is 8 dB, the noise power σ^2 is -114 dBm. Besides, we set the transmission latency between the cloud and the edge to $T_d = 50$ time slots, and set the period of updating the parameter vectors of edge DNNs to $T_u = 100$ time slots.

B. Performance comparisons

The simulation results are shown in Fig. 5 and Fig. 6, in which each curve is the average of 10 independent trials.

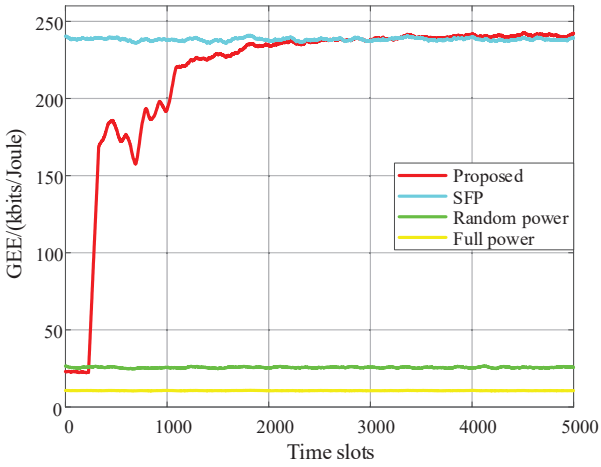
Fig. 5 provides the GEE performance of the four algorithms. Fig. 5-(a) shows the average GEE performance during the training stage, in which the parameter vectors of all the DNNs are continuously updated. It can be observed that the performance of the proposed algorithm starts to increase after about 300 time slots, and exceeds the performance of random power and full power algorithms. Besides, the proposed algorithm converges quickly and can approach the performance of the SFP algorithm after 2500 time slots. Fig. 5-(b) shows the performance comparison in the testing stage, in which the parameter vectors of all the DNNs are fixed after the convergence and each edge BS configures independently its transmit power with the output of the corresponding edge DNN. From the figure, the proposed algorithm outperforms the SFP algorithm as well as the random algorithm and full power algorithm.

Fig. 6 shows the average time complexity of the proposed algorithm and the SFP algorithm for calculating a proper transmit power. It can be seen that the proposed algorithm needs around 1.03 ms on average while the SFP algorithm needs around 68.34 ms on average to optimize a transmit power.

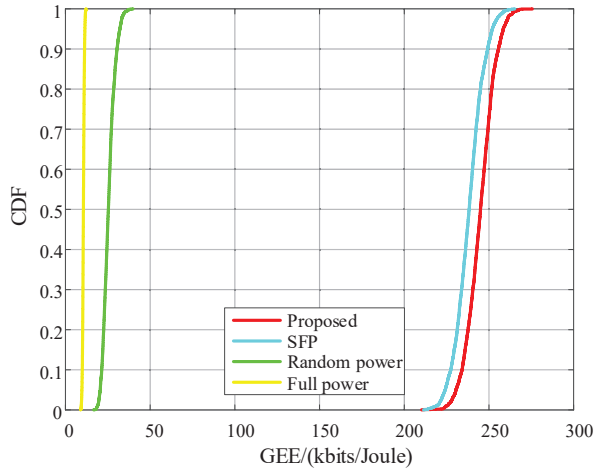
From Fig. 5 and Fig. 6, it can be concluded that the proposed algorithm outperforms the SFP algorithm in terms of both GEE performance and time complexity.

$$J(\theta_1^{(\mathrm{a})},\cdots,\theta_N^{(\mathrm{a})}) = \mathbb{E}_{\boldsymbol{s}}\left[Q\left(\boldsymbol{s},\mu_1^{(a)}(s_1;\theta_1^{(\mathrm{a})}),\cdots,\mu_N^{(a)}(s_N;\theta_N^{(\mathrm{a})});\theta^{(\mathrm{c})}\right)\right]. \tag{12}$$

$$\nabla_{\theta_n^{(\mathrm{a})}} J(\theta_1^{(\mathrm{a})},\cdots,\theta_N^{(\mathrm{a})}) \approx \frac{1}{D}\sum_i \nabla_{a_n} Q\left(\boldsymbol{s},\boldsymbol{a};\theta^{(\mathrm{c})}\right)|_{a_n=\mu_n^{(\mathrm{a})}(s_n;\theta_n^{(\mathrm{a})})} \nabla_{\theta_n^{(\mathrm{a})}} \mu_n^{(\mathrm{a})}(s_n;\theta_n^{(\mathrm{a})}). \tag{13}$$



(a) GEE performance in the training stage



(b) GEE performance in the testing stage

Figure 5. GEE performance comparison in the training stage and testing stage. Each value is a moving average of the previous 100 time slots.

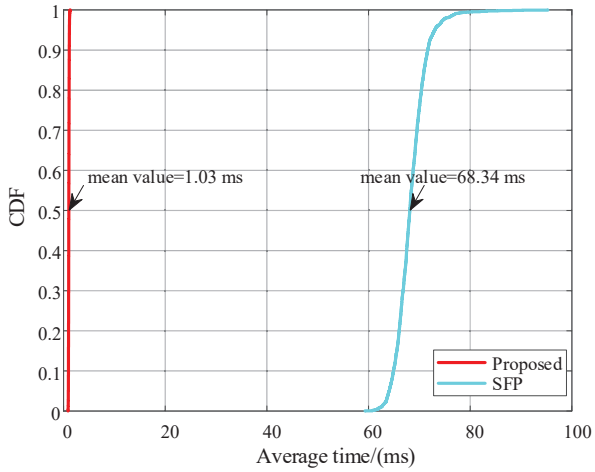


Figure 6. Time complexity comparison with the SFP algorithm.

V. Conclusions

In this paper, we studied a typical HetNet, in which a macro BS and multiple small BSs reuse the same spectrum band. By enabling the cloud-edge collaboration, we proposed a DRL-based energy-efficient power control algorithm. With the proposed algorithm, each BS can independently determine the transmit power based on only local information. Simulation results show the advantages of the proposed algorithm compared with conventional methods.

VI. Acknowledgement

This work was supported in part by National Key R&D Program of China (No. 2020YFB1805001), National Natural Science Foundation of China under Grants 62031008, 61801101 and 61801102.

References

[1] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5G wireless networks: A comprehensive survey," *IEEE Commun. Surveys & Tutorials*, vol. 18, no. 3, pp. 1617-1655, Third Quarter, 2016.

[2] D. Feng, C. Jiang, G. Lim, L. J. Cimini, G. Feng, and G. Y. Li, "A survey of energy-efficient wireless communications," *IEEE Commun. Surveys & Tutorials*, vol. 15, no. 1, pp. 167-178, First Quarter, 2013.

[3] K. Shen and W. Yu, "Fractional programming for communication systems—part I: Power control and beamforming," *IEEE Trans. on Signal Process.*, vol. 66, no. 10, pp. 2616-2630, May 2018.

[4] A. Zappone, E. Björnson, L. Sanguinetti, and E. Jorswieck, "Globally optimal energy-efficient power control and receiver design in wireless networks," *IEEE Trans. on Signal Process.*, vol. 65, no. 11, pp. 2844-2859, Jun. 2017.

[5] B. Matthiesen, A. Zappone, K. -L. Besser, E. A. Jorswieck, and M. Debbah, "A globally optimal energy-efficient power control framework and its efficient implementation in wireless interference networks," *IEEE Trans. on Signal Process.*, vol. 68, pp. 3887-3902, Jun. 2020.

[6] A. Zappone, M. Debbah, and Z. Altman, "Online energy-efficient power control in wireless networks by deep neural networks," *Proc. IEEE 19th Int. Workshop Signal Process. Adv. Wireless Commun.*, pp. 1-5, Jun. 2018.

[7] D. Shi, F. Tian, and S. Wu, "Energy efficiency optimization in heterogeneous networks based on deep reinforcement learning," *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, pp. 1-6, Jun. 2020.

[8] K. K. Nguyen, T. Q. Duong, N. A. Vien, N. -A. Le-Khac, and M. N. Nguyen, "Non-cooperative energy efficient power allocation game in D2D communication: A multi-agent deep reinforcement learning approach," *IEEE Access*, vol. 7, pp. 100480-100490, 2019.

[9] A. Zappone and E. Jorswieck, "Energy efficiency in wireless networks via fractional programming theory," *Found. Trends Commun. Inf. Theory*, vol. 11, no. 3-4, pp. 185-396, 2015.

[10] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," ICML, Jun. 2016.