Dynamic Spectrum Access for Femtocell Networks: A Graph Neural Network Based Learning Approach

He Jiang*, Haibo He*, and Lingjia Liu†
*Department of Electrical, Computer, and Biomedical Engineering, University of Rhode Island
Kingston, RI 02881, USA
Email: {hjiang, he}@ele.uri.edu
†Bradley Department of Electrical and Computer Engineering
Virginia Tech, Blacksburg, VA, 24060, USA
Email: ljliu@vt.edu

Abstract—**This paper concerns the dynamic spectrum access problem for femtocell networks, where traffic loads are different among cells. We model the interference relationship of the femtocell networks with conflict graphs, and a graphical game is employed as the channel access coordination mechanism. A graph neural network based architecture is proposed, which directly maps traffic loads to the channel access scheme for each femtocell. With our method, each femtocell first estimates the qualities of all available channels based on the information from its neighbors, and then the channels of the highest quality are accessed. A multiagent reinforcement learning framework is designed to train the proposed architecture to make accurate estimations of channel quality.**

I. Introduction

In recent years, the number of wireless devices have a great increase which makes it is necessary to upgrade our wireless communication networks to meet the unprecedented traffic demands from these wireless devices. Under this circumstance, femtocell network is proposed as a promising cost-effective solution to provide sufficient traffic capacity [1]–[3]. A significant property of femtocell networks is that the transmitting power of APs is low, such that it is possible to boost the spectrum efficiency via spectrum spatial reuse [4]. Moreover, the traffic loads of femtocells can be varied both temporally and spatially [5]. Consequently, dynamic spectrum access (DSA) [6] for femtocell network attracts the interests of numerous researchers in the community. Substantial DSA methods have been proposed to improve the spectrum efficiency from different perspectives [7]–[9]. Most of these approaches generate the solutions in an iterative searching style. However, as spectrum demands of femtocells can vary dynamically over time, the iterative searching approaches need to be reimplemented for each encountered states. Accordingly, it is desirable to have a generalized spectrum access policy that directly reveals the solution for each specific traffic state. Machine learning techniques have shown great potentials in various decision-making problem of many other disciplines [10]–[12] as well as in solving complex problems for large-scale communication networks [13]–[16]. This article aims to develop an autonomous machine learning based approach to discover a spectrum access policy that directly maps the traffic loads of femtocell network to an efficient channel access scheme.

In this paper, we investigate the DSA for femtocell network, where femtocells experience different channel demands. Each femtocell is served by one AP and the corresponding AP selects a certain amount of channels based on its traffic load. The interference relationship of the femtocell network is modeled by a distance-based conflict graph. A channel access graphical game is employed to coordinate the interference among the neighboring femtocells. In the game, each AP aims to select the channels that can maximize its expected throughput. We formulate a reinforcement learning framework for the DSA of femtocell networks. With this approach, a desired spectrum access policy can be automatically discovered. The learned policy can instruct each AP to find the channels which will lead to low interference. Moreover, we design a graph neural network based deep learning architecture to implement the proposed reinforcement learning. In this way, the learned policy is embedded in the deep architecture and possesses excellent generalization capability, which can be directly applied to the each encountered traffic state.

II. System model and problem formulation

A. System Model

We consider a femtocell network containing N femtocells with each cell served by one AP. And there are M wireless channels of the same transmission rate, which compose the set $\mathcal{C} = \{C_1, \ \ldots, \ C_M\}$. The n-the AP selects D_n channels from $\mathcal{C}$ to transmit data for its users. D_n is jointly determined by the user number and their traffic demands [9]. Following the previous works [9], [17], the interference relationship of the APs is described by a distance based model, where if the distance between two APs is less than a threshold d_0, they can interfere with each other. Fig.1 illustrates an example of the considered femtocell network. According to this interference model, we can obtain a conflict graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{1, \ 2, \ \ldots, \ N\}$ is the vertex set, and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is the edge set. In the graph, each vertex represents an AP, and an edge between two vertices indicates that the corresponding APs can interfere with each other. The collection of all

Authorized licensed use limited to: IEEE Editors-in-Chief. Downloaded on June 07,2020 at 16:07:27 UTC from IEEE Xplore. Restrictions apply.

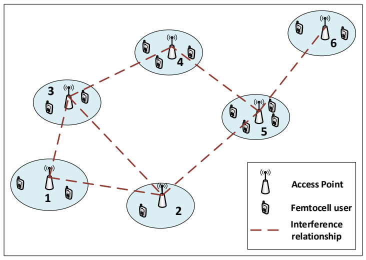


Fig. 1. An example of the femtocell network. Each femtocell serves several users. The number in each femtocell is its index. Two APs can interfere with each other if their distance is less than the threshold.

neighbors of n-th vertex are denoted by $\mathcal{J}_n$. When multiple neighboring APs access one channel simultaneously, CSMA is applied to coordinate transmission collisions [17], [18]. In such a femtocell network, the throughput of each cell depends not only on its own channel selection but also on the actions of all its neighbors. Therefore, it is essential to design a channel access policy to improve transmission efficiency. Game theory has been widely utilized to obtain efficient channel access policies for communication networks [9], [15], [19], [20]. In our work, we use a graphical game to coordinate the channel access of the APs.

B. Graphical Channel Access Game

Denote the set of the channels accessed by AP n by $a_n = \{C_{n_1}, \ldots, C_{n_{D_n}}\}$. With CSMA, an AP can succefully use one channel only if all other neighboring APs on the same channel keep silent. Following the definition of [17], the MAC layer interference on C_{n_i} is $I_{n_i} = \sum_{k\in\mathcal{J}_n}\sum_{j=1}^{D_k}\delta(C_{n_i}, C_{k_j})$, where $\delta(C_{n_i}, C_{k_j}) = 1$ if $C_{n_i} = C_{k_j}$; else $\delta(C_{n_i}, C_{k_j}) = 0$. For AP n, the probability of successfully accessing C_{n_i} is $1/(1 + I_{n_i})$; and the expected achievable transmission rate on C_{n_i} can be estimated by $E[r_{n_i}] = S_{C_i}/(1 + I_{n_i})$. The quantity $S_{C_{n_i}}$ is the transmission rate of C_{n_i}, which is assume to be identical for all channels. Accordingly, the expected throughput of AP n can be calculated as $R_n = \sum_{i=1}^{D_n} E[r_{n_i}]$.

The graphical channel access game is denoted by $\mathcal{F} = (\mathcal{G}, \mathcal{A}, \mathcal{U})$, where $\mathcal{G}$ is the conflict graph defined in the previous subsection; $\mathcal{A} = \mathcal{A}_1 \otimes \mathcal{A}_2 \otimes \ldots \otimes \mathcal{A}_N$; $\otimes$ represents Cartesian product; $\mathcal{U} = \{u_n\}_{n=1}^{N}$. For vertex n, $\mathcal{A}_n$ is the set of all feasible actions, and the quantity u_n is the utility function, which is defined as $u_n \equiv R_n$. Under this definition, an AP can achieve the maximum expected throughput by maximizing its own utility function. Consequently, the optimal policy can be written as $a_n^* = \arg\max_{a_n}\{u_n(a_n, a_{\mathcal{J}_n})\}$, where $a_{\mathcal{J}_n}$ is the action profile of all its neighbors. This problem and many of its variants have been studied in [9], [19], [21], [22]. In the following section, we will propose a graph neural network based multiagent Q-learning algorithm that learns a generalized policy that directly gives out the actions based on the information of the local traffic loads.

III. Graph Neural Network Based Learning Solution

A. Dynamic Channel Access via Reinforcement Learning

From I_{ni} and $E[r_{ni}]$, it can be observed that the expected throughput of each AP depends on the MAC layer interference of its selected channels. Therefore, a low interference level naturally indicates a high channel quality. For AP n, define the Q value of the channels as

$$Q_n = [q_{n,1}, q_{n,2}, \ldots, q_{n,M}]^{\mathrm{T}}, \quad (1)$$

where $q_{n,m}$ represents the Q value of C_m, and $q_{n,m} = -\sum_{k\in\mathcal{J}_n}\sum_{j=1}^{D_k}\delta(C_m, C_{k_j})$. Based on this definition, a reasonable policy π_n should select D_n channels of the highest quality. However, estimating the accurate value of Q_n can be difficult since this quantity is jointly decided by the actions of all adjacent APs. Thus, we design a multiagent Q learning algorithm to improve the estimation accuracy and to get a near optimal action for each AP, which is shown in Table I. In our

TABLE I
MULTIAGENT Q LEARNING FOR THE CHANNEL ACCESS GAME

Initialization:
Set $t = 0$ and randomly initialize Q_n^0, for $\forall n \in \{1, 2, \ldots, N\}$.

for $t \leq T$ **do**:
1. For $\forall n$, generate a_n with ϵ-greedy policy according to Q_n^t.
2. Implement the actions and calculate the corresponding $\{I_{n_1}, \ldots, I_{n_M}\}$, for $\forall n$;
3. For $\forall n$ and $\forall m$, update $q_{n,m}$ with
$$q_{n,m}^{t+1} = \alpha r_{n,m} + (1-\alpha) q_{n,m}^t \quad (2)$$
4. $t = t + 1$

algorithm, all APs will select channels according to their own Q value with an ϵ-greedy policy. After the channel selection being implemented, the APs will exchange the information of their action among the neighbors and get the channel interference levels based on the received information. Then, Q value is updated by (2) for all channels. In (2), α is the learning rate; $r_{n,m}$ is $r_{n,m} = -\sum_{k\in\mathcal{J}_n}\sum_{j=1}^{D_k}\delta(C_m, C_{k_j})$, which counts how many neighboring APs access C_m.

This Q learning algorithm can be easily implemented via a Q table. However, as the system state space grows exponentially with the number of AP, the table size can be extremely large. Moreover, the generalization capability is quite limited since the table will not contain the Q value of a state that is not encountered in the training process. To overcome these disadvantages, we developed a graph neural network based implementation scheme in the next subsection.

B. Graph Neural Network Based Implementation

When implementing a Q learning algorithm with a neural network, the neural network functions as a mapping from state representations to the Q values of feasible actions. Traditionally, the state representation is extracted from the raw input through a deep neural network. However, in the graphical channel access game studied in this article, the data is graph-structured with each node carrying the local channel demand, such that a traditional convolutional networks or fully connected networks are no longer applicable. Fortunately, graph neural networks have been proved to be powerful tools for processing graph-structured data. Consequently, we design the following graph neural network based architecture to implement the proposed multiagent Q learning algorithm,

Authorized licensed use limited to: IEEE Editors-in-Chief. Downloaded on June 07,2020 at 16:07:27 UTC from IEEE Xplore. Restrictions apply.

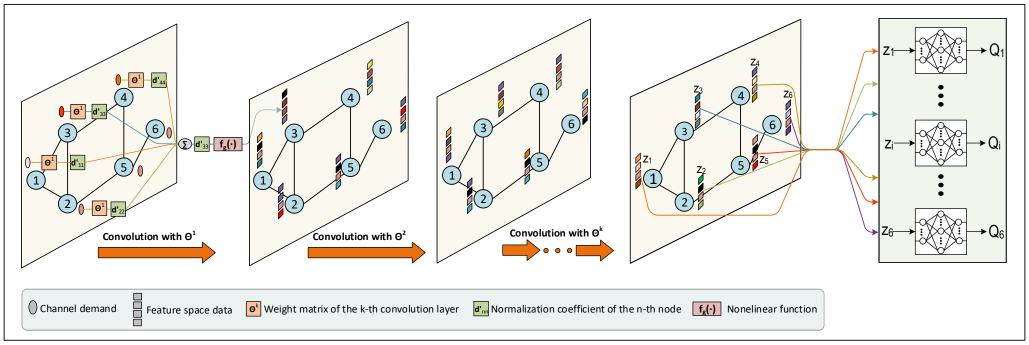


Fig. 2. The proposed graph neural network based architecture for the estimation of channel quality.

which is illustrated in Fig. 2. This architecture takes the graph data as input, where each vertex value is defined by the corresponding channel demand. A graph convolutional neural network (GCN) [23] is employed to extract the feature for each vertex. At each layer of GCN, the node feature is updated according to the features of all neighbors. Denoting $Z = [z_1^{\mathrm{T}},\ z_2^{\mathrm{T}},\ \dots,\ z_N^{\mathrm{T}}]^{\mathrm{T}}$, where z_n is the feature of the n-th vertex, the feature aggregation procedure at the k-th iteration can be written as

$$Z^k = f_g(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}Z^{k-1}\Theta^k), \tag{3}$$

where f_g is a nonlinear function; Θ^k is a trainable matrix; $\tilde{A} = I + A$ with A and I being the adjacency matrix and identity matrix respectively; $\tilde{D}$ is the diagonal node degree matrix of $\tilde{A}$. It should be noted that the initial feature of a vertex is set to the corresponding channel demands. To make this process intelligible, we make an illustration on vertex three in Fig. 2. First, the channel demands of vertex three and all its neighbors are mapped to the feature space though Θ^1. Then, the obtained data are normalized via the corresponding coefficients and added together. After that, the summation will be normalized by d'_{33} where $d'_{ii} = 1/\sqrt{\tilde{D}_{ii}}$, and input to the nonlinear function f_g. Finally, the output data is used to update the feature of vertex three. Based on the obtained feature of each vertex, the corresponding Q value will be estimated through a local fully connected neural network which is designed as $Q_n = W_{n,2}f_q(W_{n,1}z_n)$, where $W_{n,1}$ is the weight matrix connecting the input layer and the hidden layer; $W_{n,2}$ is the weight matrix connecting the hidden layer and the output layer.

Since the Q value is directly related to the weights of both GCN and the fully connected neural networks, to generate an accurate estimation, we train this proposed architecture through the multiagent Q learning algorithm of Table I. First, we randomly initialize weights, and for a specific channel demand state $\{D_n\}_{n=1}^N$, Q_n is calculated through our designed network. Then, the actions of all vertices are generated based on the ϵ-greedy policy. With these actions, the training error is calculated by

$$e = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{M}\sum_{m=1}^{M}(q_{n,m} - y_{n,m})^2, \tag{4}$$

where $y_{n,m} = \alpha r_{n,m} + (1-\alpha)q_{n,m}$. Finally, to minimize this training error, all the weights are updated by the gradient descent algorithm. The whole training process is summarized in Table II.

TABLE II
THE TRAINING PROCESS OF THE PROPOSED ARCHITECTURE

Initialization:
Set $t = 0$ and initialize $\{\Theta^k\}_{k=1}^K$, $\{W_{i,1}\}_{i=1}^N$, $\{W_{i,2}\}_{i=1}^N$.

for $t \le T$ **do**:
1. Calculate Q_n^t for $\forall n$;
2. With ϵ-greedy policy, generate a_n based on Q_n^t, for $\forall n$;
3. Implement the actions and calculate the rewards based ;
4. For $\forall n$, calculate the training error e and update the parameters with gradient descent algorithm.
5. $t = t + 1$

IV. SIMULATIONS AND ANALYSIS

The femtocell networks we considered in the simulations are generated based on the model used in [18], i.e. a certain number of APs randomly deployed in a fixed square region. Our simulations involve six femtocell networks with different femtocell numbers. The topologies of these generated femtocell network are shown in Fig. 3. For all the six

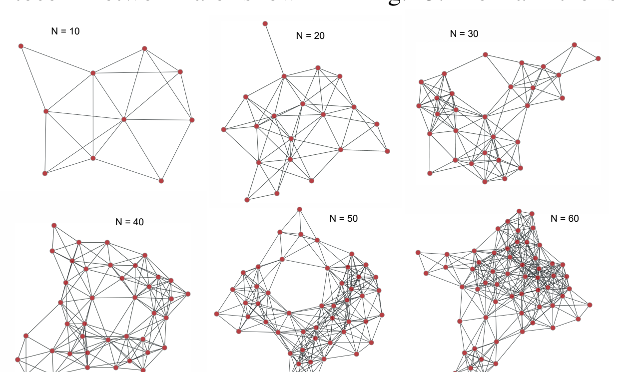


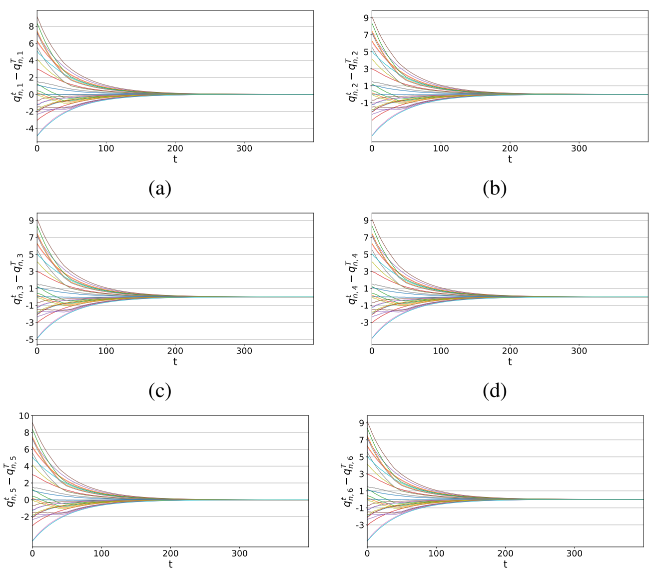
Fig. 3. The communication topologies of the six femtocell networks. The red circles in each subfigure denote the APs and the edges represent the interference relationship.

networks, we assume that there are total six channels available, and the channel demands of the APs are randomly sampled from the set $\{0,\ 1,\ 2,\ 3,\ 4\}$.

To begin with, we check the convergence capability of the proposed multiagent Q learning algorithm with the femtocell

Authorized licensed use limited to: IEEE Editors-in-Chief. Downloaded on June 07,2020 at 16:07:27 UTC from IEEE Xplore. Restrictions apply.

network with 30 APs. The convergence capability is verified from two aspects: 1) for a specific traffic state, whether the multiagent Q learning algorithm can generate converged channel quality estimations; 2) with multiple training cases, whether the weights of proposed graph neural network based architecture can be trained to converged value. To verify the first point, we randomly generate a channel demand and assign a random channel Q value for each of the 30 APs. The multiagent Q learning algorithm of Table I is applied to update the Q value for all APs. The evolution procedure of the Q

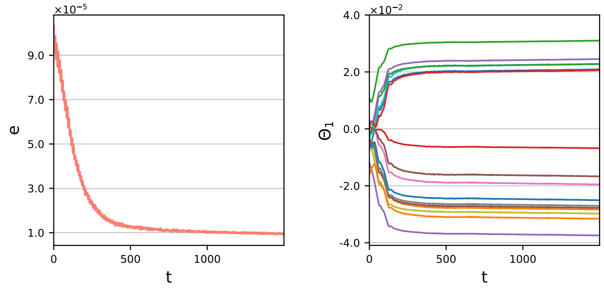


(a) (b) (c) (d) (e) (f)

Fig. 4. The evolution of the Q values of the six available channels. The sub-figures (a)-(f) show the results of C_1-C_6 correspondingly. In a sub-figure, x-axis denotes the learning step, and the colored lines shows Q value evolution of all APs with one line representing an AP.

values is shown in Fig. 4, where six sub-figures show the Q values for the six channels. We can observe that through the multiagent Q learning, each AP can obtain a converged quality estimation for all channels. For the second point, we randomly generate 1000 traffic states of the same femtocell network for the training purpose and apply these training cases to train the proposed architecture with the multiagent Q learning algorithm of Table II. The applied GCN contains four layers with 16 convolutional channels in each layer. For each AP, the fully connected network that maps the vertex feature to the Q value is set to a three-layer neural network with 16 hidden neurons. Both $f_g(\cdot)$ and $f_q(\cdot)$ are set to the leaky Rectifier function. To verify that this architecture can be trained to converged weights, we plot the training error (e) and the value of Θ^1 in Fig.5, which converge during training.

Then, we evaluate the performance of our method on all six femtocells. For each network, we randomly generate 1000 traffic states for training purpose and another 2000 traffic states for testing. After training, these deep neural networks are applied to solve the corresponding testing cases. Naturally, a good performance is indicated by a low interference level for each AP. Thus, we define a quantity that measures the



(a) (b)

Fig. 5. (a) The training error of the proposed network for the femtocell network with 30 APs. (b) the weights evolution process of the first convolutional layer of the applied GCN.

average interference level which is

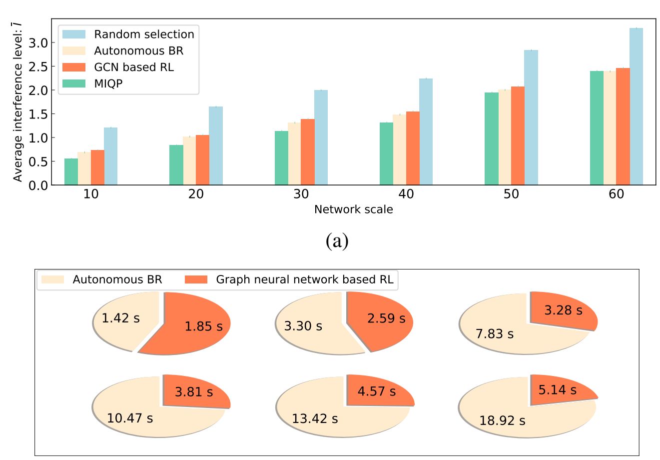
$$\overline{I} = \frac{1}{L}\sum_{j=1}^{L}\sum_{n=1}^{N}\frac{1}{D_{n,j}}\sum_{C_{n_i}\in a_{n,j}} I_{n_i,j}. \tag{5}$$

In (5), the number L represents the total testing volume which is 2000; j is the testing cases index. The optimal value of $\overline{I}$ can be obtained by solving a mixed-integer quadratic programming (MIQP) problem for every testing case j:

$$\text{Minimize} \quad \sum_{n=1}^{N}\frac{1}{D_{n,j}}\sum_{m=1}^{M} x_{n,m}[\sum_{k\in\mathcal{J}_n} x_{k,m}]$$

$$\text{Subject to} \sum_{m=1}^{M} x_{n,m} = D_n, \ x_{n,m}\in\{0,1\}, \forall n, \forall m.$$

In this formulation, $x_{n,m}$ is the binary decision variable of AP n on channel m. To make a better evaluation, we also compare our method with the autonomous best response algorithm proposed in [9] as well as the the random selection policy. The simulations results concerning the average interference level of all these four methods are shown in Fig. 6 (a). We can see that the average interference levels under both



(a)

(b)

Fig. 6. Performance comparison under different femtocell networks. (a) Average interference level of the solution generated by the corresponding method. (b) The computation time (in seconds) of the autonomous best response algorithm and the proposed graph neural network based reinforcement learning approach.

the autonomous best response algorithm and our proposed

Authorized licensed use limited to: IEEE Editors-in-Chief. Downloaded on June 07,2020 at 16:07:27 UTC from IEEE Xplore. Restrictions apply.

methods are significantly lower than that achieved by the random selection policy. Solving MIQP will lead us to the best results in terms of $\overline{I}$. However, the computation time can be tremendous for the large femtocell networks, which is the most critical disadvantage of this approach. In our experiments, we solve the MIQP with CPLEX [24] and it will take several days to get the converged results when $N = 60$. Under this situation, we limit the maximum optimizing time of CPLEX to two hours, and the corresponding results are illustrated with the green bars in Fig. 6 (a). It can be observed that the MIQP can achieve a certain advantage over our method in terms of $\overline{I}$. But, the advantage appears to be not significant enough considering the huge computation time. With the autonomous best response algorithm, we can obtain a marginally lower average interference level comparing with our proposed method. However, this slight improvement is based on the cost of longer computation time. We depict the computation time of these two methods for solving the 2000 testing cases in Fig. 6 (b). We can see that when the femtocell network is small ($N = 10$), the computation time of our method is a little longer than that of the autonomous best response algorithm; however, for larger femtocell networks, our method cost obviously shorter computation time. This is because the autonomous best response algorithm needs more iterations to converge as the AP number in a femtocell network increase. These results evidently demonstrate the advantage of our method on the DSA of large femtocell networks.

V. Conclusion

In this paper, we investigate the DSA problem for femtocell networks. Traffic loads in each femtocell are assumed to be different, which is characterized by different channel demands. A graphical game is employed to coordinate the channel access among the APs. We develop a graph neural network based multiagent reinforcement learning approach for this problem. With our method, each AP aggregates information from its neighbors through a graph neural network and based on this information, it estimates the channel quality via a local fully connected neural network. Simulations on femtocell networks of different scales demonstrate that comparing with the existing autonomous best response algorithm, our method can generate the solutions of nearly the same quality while achieving advantages on the computation time, especially for the femtocell network with a quantity of APs.

VI. Acknowledgement

This work was supported by the National Science Foundation under grant ECCS 1731672.

References

[1] V. Chandrasekhar, J. G. Andrews, and A. Gatherer, “Femtocell networks: a survey,” *IEEE Communications magazine*, vol. 46, no. 9, 2008.

[2] J. G. Andrews, H. Claussen, M. Dohler, S. Rangan, and M. C. Reed, “Femtocells: Past, present, and future,” *IEEE Journal on Selected Areas in communications*, vol. 30, no. 3, pp. 497–508, 2012.

[3] V. Chandrasekhar and J. G. Andrews, “Uplink capacity and interference avoidance for two-tier femtocell networks,” *arXiv preprint cs/0702132*, 2007.

[4] M. Dohler, R. W. Heath, A. Lozano, C. B. Papadias, and R. A. Valenzuela, “Is the phy layer dead?” *IEEE Communications Magazine*, vol. 49, no. 4, 2011.

[5] P. Rost, C. J. Bernardos, A. De Domenico, M. Di Girolamo, M. Lalam, A. Maeder, D. Sabella, and D. Wübben, “Cloud technologies for flexible 5g radio access networks,” *IEEE Communications Magazine*, vol. 52, no. 5, pp. 68–76, 2014.

[6] Q. Zhao and A. Swami, “A survey of dynamic spectrum access: Signal processing and networking perspectives,” in *Acoustics, speech and signal processing, 2007. ICASSP 2007. IEEE international conference on*, vol. 4. IEEE, 2007, pp. IV–1349.

[7] X. Chen and J. Huang, “Distributed spectrum access with spatial reuse,” *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 3, pp. 593–603, 2013.

[8] P. Chandhar and S. S. Das, “Area spectral efficiency of co-channel deployed ofdma femtocell networks,” *IEEE Transactions on Wireless Communications*, vol. 13, no. 7, pp. 3524–3538, 2014.

[9] Y. Xu, C. Wang, J. Chen, J. Wang, Y. Xu, Q. Wu, and A. Anpalagan, “Load-aware dynamic spectrum access for small-cell networks: a graphical game approach,” *IEEE Transactions on Vehicular Technology*, vol. 65, no. 10, pp. 8794–8800, 2016.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, p. 529, 2015.

[11] E. Khalil, H. Dai, Y. Zhang, B. Dilkina, and L. Song, “Learning combinatorial optimization algorithms over graphs,” in *Advances in Neural Information Processing Systems*, 2017, pp. 6348–6358.

[12] H. He and H. Jiang, “Deep learning based energy efficiency optimization for distributed cooperative spectrum sensing,” *IEEE Wireless Communications*, 2019.

[13] H. Li, “Multi-agent q-learning for competitive spectrum access in cognitive radio systems,” in *2010 Fifth IEEE Workshop on Networking Technologies for Software Defined Radio Networks (SDR)*. IEEE, 2010, pp. 1–6.

[14] H. Jiang, H. He, L. Liu, and Y. Yi, “Q-learning for non-cooperative channel access game of cognitive radio networks,” in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–7.

[15] C. Fan, B. Li, C. Zhao, W. Guo, and Y.-C. Liang, “Learning-based spectrum sharing and spatial reuse in mm-wave ultradense networks,” *IEEE Transactions on Vehicular Technology*, vol. 67, no. 6, pp. 4954–4968, 2018.

[16] O. Naparstek and K. Cohen, “Deep multi-user reinforcement learning for distributed dynamic spectrum access,” *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 310–323, 2019.

[17] Y. Xu, Q. Wu, L. Shen, J. Wang, and A. Anpalagan, “Opportunistic spectrum access with spatial reuse: Graphical game and uncoupled learning solutions,” *IEEE Transactions on Wireless Communications*, vol. 12, no. 10, pp. 4814–4826, 2013.

[18] Y. Xu, J. Wang, Q. Wu, J. Zheng, L. Shen, and A. Anpalagan, “Dynamic spectrum access in time-varying environment: Distributed learning beyond expectation optimization,” *IEEE Transactions on Communications*, vol. 65, no. 12, pp. 5305–5318, 2017.

[19] H. Li, “Multi-agent q-learning of channel selection in multi-user cognitive radio systems: A two by two case,” in *2009 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 2009, pp. 1893–1898.

[20] B. Wang, Y. Wu, and K. R. Liu, “Game theory for cognitive radio networks: An overview,” *Computer networks*, vol. 54, no. 14, pp. 2537–2561, 2010.

[21] M. Azarafrooz and R. Chandramouli, “Distributed learning in secondary spectrum sharing graphical game,” in *2011 IEEE Global Telecommunications Conference-GLOBECOM 2011*. IEEE, 2011, pp. 1–5.

[22] Y. Xu, J. Wang, Q. Wu, A. Anpalagan, and Y.-D. Yao, “Opportunistic spectrum access in cognitive radio networks: Global optimization using local interaction games,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 6, no. 2, pp. 180–194, 2012.

[23] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *arXiv preprint arXiv:1609.02907*, 2016.

[24] I. I. Cplex, “V12. 1: User's manual for cplex,” *International Business Machines Corporation*, vol. 46, no. 53, p. 157, 2009.

Authorized licensed use limited to: IEEE Editors-in-Chief. Downloaded on June 07,2020 at 16:07:27 UTC from IEEE Xplore. Restrictions apply.