Deep Reinforcement Learning and Graph Neural Networks for Efficient Resource Allocation in 5G Networks

Martín Randall*†, Pablo Belzarena*, Federico Larroca*, Pedro Casas†
*Instituto de Ingeniería Eléctrica, Facultad de Ingeniería, Universidad de la República, Uruguay
†AIT, Austrian Institute of Technology, Austria
{mrandall, belza, flarroca}@fing.edu.uy, pedro.casas@ait.ac.at

***Abstract*—The increased sophistication of mobile networks such as 5G and beyond, and the plethora of devices and novel use cases to be supported by these networks, make of the already complex problem of resource allocation in wireless networks a paramount challenge. We address the specific problem of user association, a largely explored yet open resource allocation problem in wireless systems. We introduce GROWS, a deep reinforcement learning (DRL) driven approach to efficiently assign mobile users to base stations, which combines a well-known extension of Deep Q Networks (DQNs) with Graph Neural Networks (GNNs) to better model the function of expected rewards. We show how GROWS can learn a user association policy which improves over currently applied assignation heuristics, as well as compared against more traditional Q-learning approaches, improving utility by more than 10%, while reducing user rejections up to 20%.**

***Index Terms*—User Association; Mobile Networks; Reinforcement Learning; Graph Neural Networks**

I. Introduction

The problem of resource allocation, as in finding a resource distribution that satisfies some chosen metric can be thought of as *how to do more with less?*. Specially in engineering, optimization of resources to achieve a given purpose is a prominent area of studies. In wireless systems, next generation mobile networks such as 5G and beyond, are meant to accommodate to the rapid growth of connected devices (e.g., IoT), coupled with a huge surge of data consumption (e.g., high definition video streaming, etc). An important enhancement to next generation mobile networks is the possibility of implementing smart, adaptable resource allocation approaches. This is promoted from the 5G and beyond standards, and accelerated by the astonishing results that machine learning has achieved as of late – albeit mainly in other fields, and much work has been devoted to bringing the artificial intelligence and the wireless communication worlds together. All of this leads to a renewed interest on research for smart adaptable user association policies.

Among novel deep learning techniques, graph neural networks (GNNs) have gained much attention, due to their ability to exploit complex data structures and to generalize to unseen scenarios, and they seem particularly fit to dynamic applications such as user association. They have been deemed particularly suited to networking problems, due to the wide use of graphs in the domain and their generalization capability. Furthermore, they are amenable to a distributed implementation, a quality of paramount importance in several situations.

In this paper we focus on **the problem of User Association (UA): to which connectivity provider (e.g., base station) should a user get connected to, to maximize a global system utility function**, typically throughput related. The optimal policy is usually intractable, since this kind of resource allocation problems is NP-hard, and the number of possible states grows too large to do a search over all possible decisions. This makes of user association an open problem, and although simple heuristics yield good enough results for simple scenarios (e.g., low congestion), there is much to gain from combining state of the art artificial intelligence with simple and robust system models. The UA problem may be stated as a sequential decision-making problem; therefore, we consider it from a Deep Reinforcement Learning (DRL) perspective, where the value function is approximated by a GNN. Our work constitutes, to the extent of our knowledge, the first application of DRL and GNNs to user association on mobile networks. The main contributions of this paper are summarized as follows:

(1) User Association Modeling: GROWS tackles the modelling of the UA problem using graph representations and a reinforcement learning framework. We propose a decentralized decision making framework in which agents are able to serve users through a general enough system representation that allows the algorithm to function in different use cases.

(2) Validation Results in Simulated 5G Networks: we provide a series of results validating GROWS in simulated 5G-network scenarios. Benchmarking results show that GROWS reduces user rejections up to 20%, while achieving an increased utility of more than 10%, as compared to classical RL-based algorithms and currently applied UA heuristics.

(3) GROWS Code and Reproducibility: for the sake of reproducibility and as an additional contribution of this paper, we openly release the current implementation of GROWS to the community. The code is available at ***https://gitlab.fing.edu.uy/mrandall/grows***.

The remainder of the paper is organized as follows. Sec. II overviews related work on user association and the application of GNNs to resource allocation problems. The GROWS algorithm is presented in Sec. III, including the system model, the reinforcement learning formulation, and the GNN model. Validation and benchmarking results are presented in Sec. IV. The paper ends with concluding remarks and future lines of work in Sec. V.

II. Related Work

A. Graph Neural Networks

Let us first briefly present GNNs. In a nutshell, it consists of a cascade of layers, each of which applies a graph filter followed by an activation function. Consider that each node in the graph has an associated vector $\mathbf{x}_i \in \mathbb{R}^d$ (for $i = 1, \ldots, N$), which may regarded as the input features. Making the analogy to discrete-time convolution, a first-order convolutional layer for a GNN may be obtained as follows [1]:

$$\mathbf{x}'_i = \sigma\left(\mathbf{\Theta}^T \sum_{j \in \mathcal{N}_i \cup \{i\}} S_{j,i}\mathbf{x}_j\right), \tag{1}$$

where $\mathbf{x}'_i \in \mathbb{R}^{d'}$ is the output of the layer, $\sigma(\cdot)$ is a point-wise non-linearity (e.g., the ReLU function), $\mathbf{\Theta} \in \mathbb{R}^{d \times d'}$ is the learnable parameter of this layer, $\mathcal{N}_i$ is the set of neighbors of node i, and $S_{i,j}$ is the i, j entry of matrix $\mathbf{S} \in \mathbb{R}^{N \times N}$, the so-called Graph Shift Operator (GSO). This is a matrix representation of the graph, which should respect its sparsity (i.e. $S_{i,j} \neq 0$ whenever there is an edge between nodes i and j). The adjacency matrix of the graph, its Laplacian or their normalized versions are all valid GSOs.

Note that in (1) each node needs to linearly combine the vectors of its neighbors only. As we concatenate K such layers, the final vector representation of node i (i.e. the output of the GNN) will depend on its neighbors up to K hops away. This observation implies that a GNN may be implemented in a fully-distributed way, as long as an edge in the graph means that the corresponding pair of nodes can communicate.

We may be more general and build higher-order filters. Let us stack all nodes' vectors $\mathbf{x}_i$ into matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$, which is called a graph signal. The matrix product $\mathbf{SX} = \mathbf{Y}$ results in another graph signal, corresponding to the first-order convolution we used in (1) (albeit without parameter $\mathbf{\Theta}$, which we will include shortly). By writing $\mathbf{S}^K\mathbf{X} = \mathbf{S}(\mathbf{S}^{K-1}\mathbf{X})$ we may see that this way we aggregate the information K hops away. Again, although it requires K rounds of information exchange, this operation may be performed without intervention of a central entity.

Finally, a general graph convolution is defined simply as a weighted sum of these K signals (i.e. $\sum_k \mathbf{S}^k\mathbf{X}h_k$, where scalars h_k are the taps of the filter). In this context, parameter $\mathbf{\Theta}$ in (1) is interpreted as a filter bank. That is to say, by considering a $d \times d'$ matrix $\mathbf{H}_k$ instead of the scalar taps, a single-layer GNN (or graph perceptron) is obtained by applying the pointwise non-linear function $\sigma(\cdot)$ to this convolution [2] [3]:

$$\mathbf{X}' = \sigma\left(\sum_{k=0}^{K-1} \mathbf{S}^k\mathbf{X}\mathbf{H}_k\right), \tag{2}$$

whereas a deep GNN is constructed by concatenating several perceptrons.

B. GNN and Reinforcement Learning for Resource Allocation

Although GNN represents a recent paradigm [4], [5], it has already proved its usefulness to address diverse resource allocation problems [6]–[8]. For complete reviews of Graph Neural Networks methods and applications we refer the reader to [9]–[11]. Examples of GNN applied to solving NP-hard or resource allocation problems include combinatorial optimization [12], measurements [13], and network virtualization [14]. The authors of [15] use a combination of fine-tuned GNN and RL to organize channel capacities on optical networks.

Closer to wireless systems, [6], [7], [16]–[19] propose the use of GNNs to find the optimal policy for a power allocation wireless system problem. An interesting work addressing radio resource management can be found in [20]. In [18], authors use GNNs to build a digital twin for network slicing. GROWS differs in several aspects from previous work: instead of optimizing an energy constrained problem, we focus on a throughput related system's utility; we establish a graph independent of the interference, which is closer to the expected 5G scenarios and smart policies avoiding interference; last, we build a general enough algorithm capable of adaptable decision making to different use cases.

C. User Association in Mobile Networks

There is a number of interesting surveys tackling the user association problem for 5G and beyond mobile networks [21], [22]. User association for millimeter wave communications has been addressed already a decade ago [23], [24], using classical optimization to find close to optimal heuristics (although making several assumptions in order to have 0-gap dual problems). In [21], authors analyze user association with focus on four major changing aspects due to 5G implementation: massive MIMO (mMIMO), heterogeneous networks (HetNet), millimeter waves (mmWave) and energy harvesting. Most work focus on one of these main changes, and choose one or more metrics for comparison with existing solutions [23], [25]–[28]. Most papers tackle the macro cell/micro cell relationship (e.g., backhaul saturation and/or load balancing) [11], [25], [27], [28] or handover schemes [29], which are are closely related to our target, but are not our main focus.

An interesting proposal arises from [30], in which authors propose a DRL solution to jointly optimize user association and resource allocation, but their algorithm needs to centralize and distribute information to every user through message passing. Authors only compare their solutions to the most common baseline (e.g., maximum received signal power, MRSP) in a few scenarios, achieving a lower utility than MRSP. The closest work to our proposal is an approach to user association from a multi-agent reinforcement learning viewpoint [31]. Authors propose an algorithm centralized in training but distributed on execution, which takes into account message passing between neighbors and delay on this information exchange. Yet, they only optimize two decisions for each base station: which user to serve at each time-step, and with what power, using an arbitrary user association policy as the

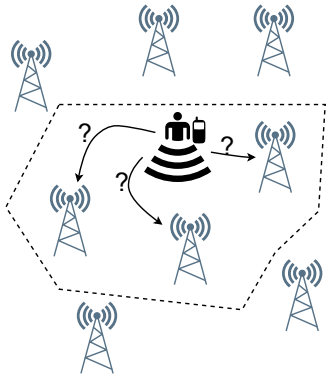

Figure 1. We consider the system formed by the k base stations with better SNR to the arrived user (in the figure $k = 3$). The system model allows this problem to be tackled through local decisions inside a larger system, limiting the complexity of the algorithms and improving scalability.

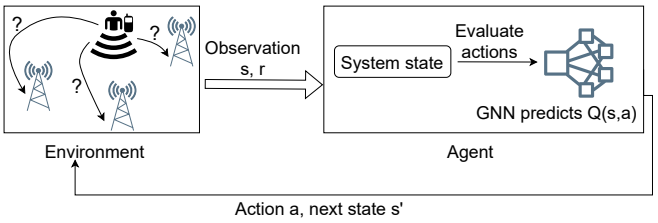

Figure 2. System model. We consider at most one arrival at each time-step. The combination of past decisions and the present arrival constitutes the state of the system. The choice of which base station associates with the currently arrived user is the action. After executing an action, a new state is observed and a reward is obtained.

maximum reference signal received power – close to a max-SNIR. To achieve scalability, they limit the user observation to a constant number – whilst we integrate the users state in our observation, and the base stations are only allowed to serve one of the top-three users.

III. The GROWS Algorithm

A. System Model

Let us consider a set of N base stations. Each base station has a limited set of frequency resources, following the 5G taxonomy we will refer to them as resource blocks (RB). We assume that base stations have an internal assignation policy they follow for their associated users, for instance to distribute equally among connected users the available resources.

As many decision problems, time will be considered slotted. At each time interval t a user may or may not arrive, following some probability distribution. As shown in Fig. 1, the user association problem can be split to consider only local decisions, enabling scaling by limiting the graph complexity and the action/state size. In the event of an arrival, one of the k base stations with strongest SNIR with the user and available resources has to associate itself with the newcomer user.

As a reinforcement learning formulation is desired [32], we need to define precisely the tuple formed by (s, a, T, r, γ). We refer to Fig. 2 for a clearer visualization.

The **system's state** s is defined as the aggregation of the base station's states and the user's state. For each base station, the state is composed of a representation of the present system's state and the new user's characteristics. Defining the base station's state are the number of users associated to it and the mean utility achieved so far. Defining the new user characteristics are: the RSSI with the corresponding base stations and a certain demand to be satisfied.

The **action** a will be to select one of the possible base stations. Please note that not every time step involves actions: they may or may not occur. In order to have a well defined Markov process, we include the decision-making in the state, by setting the demand to 0 for the time steps on which no user arrives. In this case, only one course of action is available: when no users arrive there is no action to be taken, only updating the system's state.

Transitions T will then occur as depicted in figure 2: the base station's descriptors have to be updated to include: the action effect (+1 on the number of users associated, new mean utility), and the time effect (-1 on the number of users associated if a user's demand has been satisfied). Each time step will update the "new user characteristics" in the event of a new arrival. Note that transitions are deterministic over the base station's features given the action a and the state s, but stochastic for the new user features.

Finally, the **reward** r is defined as the instantaneous utility of executing an action for a given state. In our use cases, we will use as utility/reward the log-sum of the throughput over the users, $r = \sum_N \sum_i \log(1 + th(u_{i,n}))$, where $u_{i,n}$ represents the user i associated with base station n. This promotes fairness in the resource distribution, and is widely used in literature [33].

In RL formulations, the **discount factor** γ is defined in order to estimate the expected discounted cumulative reward, which is the value algorithms will try to maximize:

$$R_t = \Sigma_t^{\infty} \gamma^t r(t)$$

The expected discounted cumulative reward is optimized by updating a policy (π) through one of Bellman's equations. In our case we consider the action-value function for policy π, defined as [32]:

$$q^{\pi}(s, a) = \mathbb{E}_{\pi}\left[\Sigma_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a\right]$$

and use the update rule given by the optimality equation for the state-action value function:

$$q^{\pi}(s, a) = r + \gamma q^{\pi}(s', \pi(s'))$$

For the graph representation, we consider at each decision time a graph formed by the base stations considering the new arrived user. Edges between nodes depict connections between base stations. All base stations are able to interact with each other in the considered 5G scenario, forming a fully connected graph. As depicted in Fig. 3, the state of each node

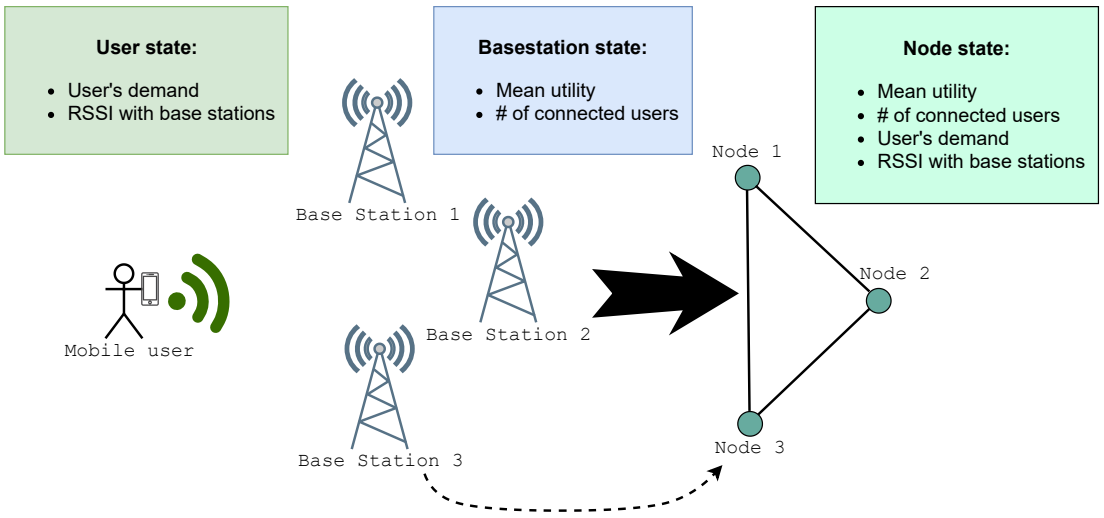

Figure 3. Graph representation of the system. Nodes represent base stations, and edges represent connections between these base stations.

is given by the state of the base station it depicts and the user's characteristics with respect to that base station.

B. Algorithm Design

GROWS algorithm is based on the classic Double DQN reinforcement learning algorithm [34], and integration with the GNN is inspired on previous work [12], [15]. The goal of the GNN is to learn how to best approximate the q-function, an estimation of the value-action function for the RL problem. The actor-critic role in reinforcement learning is to stabilize convergence of the algorithm. It is important to notice that training and execution are done separately: once the GNN is trained, prediction of the q-function according to the state and possible actions can be done instantly.

For the GNN model, we use the *LocalGNN* implementation proposed in [35], corresponding to an implementation of the popular Graph Convolutional Network (GCN), introduced in II-A. An important characteristic of the GCN is that information aggregation is done locally for each node – and extended to the K neighboring hops, which means that the GCN output for each node can run locally, enabling scaling for the proposed algorithm.

A final merge of the reward prediction for choosing which base station serves the user has to be made. This can be either decentralized (by exchanges between the base stations of their expected rewards), or centralized (if a single entity receives the states updates and calculates the maximum expected reward for the possible actions).

IV. GROWS Evaluation and Results

To evaluate GROWS behavior and performance in a mobile network use case, we use synthetic scenarios simulating 5G networks. As explained before, the goal is to learn a UA policy which maximizes the log-sum of the users' throughput, potentially improving the overall system utility as compared to certain baseline policy. As stated, the optimal policy is intractable: the problem is NP-hard and the number of states grows too fast to run a search over all possible decisions. For this reason, UA in current mobile networks is generally done through a simple heuristic, selecting the BS with the highest signal strength – we refer to this policy as an *argmax* policy, and we would consider it as the ***baseline***. Even if simple, this strategy achieves good performance, and is usually considered as baseline in the field [11], [23], [24], [27], [28], [30], [36], [37]. We compare GROWS against this simple policy, representing the currently followed strategy. As we consider a small scenario, it is possible to address the UA problem through a pure RL approach; therefore, we additionally compare GROWS against a ***Q-learning*** approach. In this sense, Q-learning tends to find the optimal policy if states and actions are visited enough, allowing to compare GROWS to an algorithm close to the optimal. In a more complex setting, the Q-learning would be unfeasible due to the large number of states, but a DRL approach as the one followed by GROWS is still viable.

We simulate a 5G network where base stations are interconnected through a back-haul. The scenario consists of three BSs, one of them having a slightly better RSSI. At each step t, users arrive with a probability $p = [0.5, 0.7, 1]$. Each user has a discrete set of possible demands to be satisfied, and a discrete set of possible RSSI values with the three BSs, randomly generated. Episodes are composed of $T = 40$ time steps, and we use an epsilon-greedy exploration/exploitation policy, exploring on the first $40,000$ episodes, and exploiting on the last $10,000$ episodes, following an exponential decay. Hyperparameters for the LocalGNN model are calibrated through grid search, resulting in the following values: a learning rate of $1e^{-4}$, a batch size of 32, and a simple architecture composed of 2 convolutional layers with filters of size 4 and 2 respectively, followed by a readout layer. The point-wise non linear function used is the hyperbolic tangent, and we include information from a 1-hop neighborhood. To avoid

	mean utility per episode									mean user rejections per episode								
	$p = 0.5$			$p = 0.7$			$p = 1$			$p = 0.5$			$p = 0.7$			$p = 1$		
$\overline{\mathcal{D}}$	**Q**	**B**	**GROWS**	**Q**	**B**	**GROWS**	**Q**	**B**	**GROWS**	**Q**	**B**	**GROWS**	**Q**	**B**	**GROWS**	**Q**	**B**	**GROWS**
6	2.67	2.64	**2.70**	**2.94**	2.86	2.92	**3.39**	3.15	3.24	0.01	0.01	**0.01**	0.25	0.26	**0.34**	2.86	3.12	**2.65**
8	2.81	2.73	**2.87**	3.07	2.93	**3.08**	**3.50**	3.2	3.42	**0.08**	0.09	0.09	**0.99**	1.07	1.10	5.59	5.63	**5.35**
10	**2.86**	2.76	2.80	3.15	2.96	**3.16**	**3.56**	3.21	3.54	0.24	0.26	**0.21**	1.93	2.00	**1.77**	7.04	7.32	**7.03**
12	2.92	2.77	**2.95**	3.19	2.96	**3.23**	**3.57**	3.21	3.55	**0.45**	0.52	0.48	2.75	2.86	**2.66**	8.34	8.38	**8.17**
14	2.92	2.78	**2.99**	**3.20**	2.95	**3.20**	**3.57**	3.21	3.52	0.69	0.83	**0.63**	3.44	3.62	**3.38**	9.15	**9.08**	9.13

Table I
(LEFT) MEAN UTILITY AND (RIGHT) MEAN USER REJECTIONS FOR DIFFERENT EXPERIMENTS, VARYING AVERAGE DEMAND ($\overline{\mathcal{D}}$) AND ARRIVAL RATE (p).

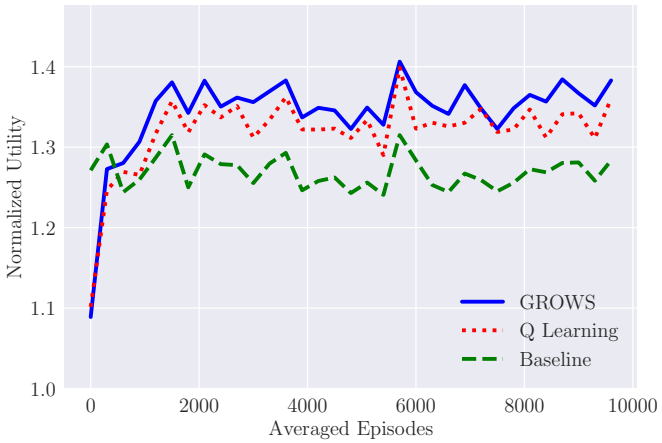

Figure 4. Already in a small mobile network topology (three BSs) with a simple traffic demand, GROWS outperforms current UA policies. The more complex the scenario, the highest the benefits we expect from GROWS as compared to the baseline *argmax* approach.

vanishing/exploding gradient issues, GNN model input are normalised. The policy network is updated every 20 steps, and the target network is updated every 200 steps, using a replay buffer of $1e^5$ samples. As discount factor, we take $\gamma = 0.5$. We take a learning rate of 0.5 for the Q-learning algorithm. Finally, initialization for the action-value function is set to 0 and policies are set to random, for both RL algorithms.

Fig. 4 reports the obtained results in terms of utility for one of the experiments ($p = 0.5$ and average demand $\overline{\mathcal{D}} = 14$), where the cumulative reward is averaged over 300 episodes. For a better visualization and interpretation of results, utility is normalized to a random UA policy, where users are assigned to BSs in a random manner; this means that a value of 1 on the normalized utility is equivalent to a random UA policy. Results are encouraging: this version of GROWS learns a proper policy, outperforming the *argmax* heuristic by more than 10%. During the first episodes, exploration is still dominant for both GROWS and Q-learning, and the greedy exploration/exploitation policy is strongly noticeable. After 1,000 episodes, GROWS learns a better UA policy with the same exploration as Q-learning, suggesting it was able to better approximate the Q-value function.

An important and desirable characteristic is the ability to handle more users/traffic. To analyze how GROWS behaves in the event of user/traffic variations, we simulate different network-load scenarios, varying the average demand ($\overline{\mathcal{D}}$) and the arrival rate (p) – p represents the probability of a user arriving on a time-slot. We assess performance in terms of the mean utility and the mean number of user rejections realized for the different experiments, comparing the three different approaches: ***baseline*** (**B**), ***Q-learning*** (**Q**), and **GROWS**. Results are summarized in Tab. I. For the small topology and experimental settings, the Q-learning policy is able to explore the system states enough to be close to optimal, but the GROWS algorithm still achieves better results for many scenarios, meaning the GCN was able to learn a good approximation of the Q value function. When demand is low, there are very few rejections, and all algorithms achieve similar results. However, as mean demand increases, GROWS is able to increase the gained utility over the baseline, proving its ability to handle traffic over more stressed situations. Regarding both mean utility and user rejection, either GROWS or the Q-learning fare better in almost all scenarios. In some cases, GROWS is able to reject a 20% less of users.

V. CONCLUSIONS

We have presented our work in the field of user association, an open problem with renewed interest to fulfill next generation mobile network requirements. We proposed a DRL formulation of the UA problem, using a GNN to estimate the Q-value function, giving birth to the GROWS algorithm: a decentralized and scalable solution for UA on wireless systems. We presented benchmarking results for several synthetic scenarios simulating mobile networks, evidencing the advantages of GROWS as compared to the state of the art in the practice of UA, realizing a higher system utility – up by 10%, and a lower user rejection – down by 20%. Generalization to unseen scenarios and traffic variations, as well as the integration of user mobility are yet to be analyzed. To fully exploit GROWS advantages, more complex and realistic scenarios would be explored as part of future work, including experiments involving different wireless systems (e.g., WIFI, Flying Ad-Hoc Networks) and different situations (e.g., flashcrowds, mobility).

ACKNOWLEDGEMENTS

This work is partially funded by the Agencia Nacional de Investigación e Innovación (ANII) project *Artificial Intelligence for 5G Networks* (FMV_1_2019_1_155700), as well as supported by the Austrian FFG ICT-of-the-Future DynAISEC project (*Adaptive AI/ML for Dynamic Cybersecurity Systems*). Martín Randall's PhD is supported by a scholarship granted by the Agencia Nacional de Investigación e Innovación (ANII).

REFERENCES

[1] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *5th International Conference on Learning Representations (ICLR-17)*, 2017.

[2] Fernando Gama, Antonio G. Marques, Geert Leus, and Alejandro Ribeiro. Convolutional neural network architectures for signals supported on graphs. *IEEE Transactions on Signal Processing*, 67(4):1034–1049, 2019.

[3] Elvin Isufi, Fernando Gama, and Alejandro Ribeiro. Edgenets:edge varying graph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021.

[4] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[5] José Suárez-Varela, Paul Almasan, Miquel Ferriol-Galmés, Krzysztof Rusek, Fabien Geyer, Xiangle Cheng, Xiang Shi, Shihan Xiao, Franco Scarselli, Albert Cabellos-Aparicio, et al. Graph neural networks for communication networks: Context, use cases and opportunities. *arXiv preprint arXiv:2112.14792*, 2021.

[6] Yifei Shen, Yuanming Shi, Jun Zhang, and Khaled B Letaief. A graph neural network approach for scalable wireless power control. In *2019 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE, 2019.

[7] Yifei Shen, Yuanming Shi, Jun Zhang, and Khaled B Letaief. Graph neural networks for scalable radio resource management: Architecture design and theoretical analysis. *IEEE Journal on Selected Areas in Communications*, 39(1):101–115, 2020.

[8] Ziyan He, Liang Wang, Hao Ye, Geoffrey Ye Li, and Biing-Hwang Fred Juang. Resource allocation based on graph neural networks in vehicular communications. In *GLOBECOM 2020-2020 IEEE Global Communications Conference*, pages 1–5. IEEE, 2020.

[9] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.

[10] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[11] Ziwei Zhang, Peng Cui, and Wenwu Zhu. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[12] Elias Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. *Advances in neural information processing systems*, 30, 2017.

[13] Miles Cranmer, Peter Melchior, and Brian Nord. Unsupervised resource allocation with graph neural networks. In *NeurIPS 2020 Workshop on Pre-registration in Machine Learning*, pages 272–284. PMLR, 2021.

[14] Penghao Sun, Julong Lan, Junfei Li, Zehua Guo, and Yuxiang Hu. Combining deep reinforcement learning with graph neural networks for optimal vnf placement. *IEEE Communications Letters*, 25(1):176–180, 2020.

[15] Paul Almasan, José Suárez-Varela, Arnau Badia-Sampera, Krzysztof Rusek, Pere Barlet-Ros, and Albert Cabellos-Aparicio. Deep reinforcement learning meets graph neural networks: Exploring a routing optimization use case. *arXiv preprint arXiv:1910.07421*, 2019.

[16] Mark Eisen, Clark Zhang, Luiz FO Chamon, Daniel D Lee, and Alejandro Ribeiro. Learning optimal resource allocations in wireless systems. *IEEE Transactions on Signal Processing*, 67(10):2775–2790, 2019.

[17] Mark Eisen and Alejandro Ribeiro. Optimal wireless resource allocation with random edge graph neural networks. *ieee transactions on signal processing*, 68:2977–2991, 2020.

[18] Haozhe Wang, Yulei Wu, Geyong Min, and Wang Miao. A graph neural network-based digital twin for network slicing management. *IEEE Transactions on Industrial Informatics*, 18(2):1367–1376, 2020.

[19] Jia Guo and Chenyang Yang. Learning power allocation for multi-cell-multi-user systems with heterogeneous graph neural networks. *IEEE Transactions on Wireless Communications*, 21(2):884–897, 2021.

[20] Arindam Chowdhury, Gunjan Verma, Chirag Rao, Ananthram Swami, and Santiago Segarra. Unfolding wmmse using graph neural networks for efficient power allocation. *IEEE Transactions on Wireless Communications*, 20(9):6004–6017, 2021.

[21] Hawar Ramazanali, Agapi Mesodiakaki, Alexey Vinel, and Christos Verikoukis. Survey of user association in 5g hetnets. In *2016 8th IEEE Latin-American conference on communications (LATINCOM)*, pages 1–6. IEEE, 2016.

[22] Xiaohu Ge, Hui Cheng, Mohsen Guizani, and Tao Han. 5g wireless backhaul networks: challenges and research advances. *IEEE network*, 28(6):6–11, 2014.

[23] Hongseok Kim, Gustavo De Veciana, Xiangying Yang, and Muthaiah Venkatachalam. Distributed α-optimal user association and cell load balancing in wireless networks. *IEEE/ACM Transactions on Networking*, 20(1):177–190, 2011.

[24] Qiaoyang Ye, Beiyu Rong, Yudong Chen, Mazin Al-Shalash, Constantine Caramanis, and Jeffrey G Andrews. User association for load balancing in heterogeneous cellular networks. *IEEE Transactions on Wireless Communications*, 12(6):2706–2716, 2013.

[25] Redhwan Q Shaddad, A Alsarori Neda'a, Mushira O Alzylai, and Tareq M Shami. Biased user association in 5g heterogeneous networks. In *2021 International Conference of Technology, Science and Administration (ICTSA)*, pages 1–4. IEEE, 2021.

[26] Ali Calhan and Murtaza Cicioğlu. Handover scheme for 5g small cell networks with non-orthogonal multiple access. *Computer Networks*, 183:107601, 2020.

[27] Xin Ge, Xiuhua Li, Hu Jin, Julian Cheng, and Victor CM Leung. Joint user association and user scheduling for load balancing in heterogeneous networks. *IEEE Transactions on Wireless Communications*, 17(5):3211–3225, 2018.

[28] Ning Wang, Ekram Hossain, and Vijay K Bhargava. Joint downlink cell association and bandwidth allocation for wireless backhauling in two-tier hetnets with large-scale antenna arrays. *IEEE Transactions on Wireless Communications*, 15(5):3251–3268, 2016.

[29] Ali Çalhan and Murtaza Cicioğlu. Handover scheme for 5g small cell networks with non-orthogonal multiple access. *Computer Networks*, 183:107601, 2020.

[30] Nan Zhao, Ying-Chang Liang, Dusit Niyato, Yiyang Pei, Minghu Wu, and Yunhao Jiang. Deep reinforcement learning for user association and resource allocation in heterogeneous cellular networks. *IEEE Transactions on Wireless Communications*, 18(11):5141–5152, 2019.

[31] Navid Naderializadeh, Jaroslaw J Sydir, Meryem Simsek, and Hosein Nikopour. Resource management in wireless networks via multi-agent deep reinforcement learning. *IEEE Transactions on Wireless Communications*, 20(6):3507–3523, 2021.

[32] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[33] Tian Bu Li Erran Li Ramachandran Ramjee, T Bu, and LE Li. Generalized proportional fair scheduling in third generation wireless data networks. In *IEEE INFOCOM*, pages 1–12, 2006.

[34] Hado Hasselt. Double q-learning. *Advances in neural information processing systems*, 23, 2010.

[35] Fernando Gama, Antonio G. Marques, Geert Leus, and Alejandro Ribeiro. Convolutional neural network architectures for signals supported on graphs. *IEEE Transactions on Signal Processing*, 67(4):1034–1049, 2019.

[36] Hisham Elshaer, Mandar N Kulkarni, Federico Boccardi, Jeffrey G Andrews, and Mischa Dohler. Downlink and uplink cell association with traditional macrocells and millimeter wave small cells. *IEEE Transactions on Wireless Communications*, 15(9):6244–6258, 2016.

[37] Jingjing Cui, Yuanwei Liu, and Arumugam Nallanathan. Multi-agent reinforcement learning-based resource allocation for uav networks. *IEEE Transactions on Wireless Communications*, 19(2):729–743, 2020.