Distributed Channel Allocation for Mobile 6G Subnetworks via Multi-Agent Deep Q-Learning

Adeogun, Ramoni Ojekunle; Berardinelli, Gilberto

Published in:
IEEE WCNC

Publication date:
2023

Document Version
Accepted author manuscript, peer reviewed version

Distributed Channel Allocation for Mobile 6G Subnetworks via Multi-Agent Deep Q-Learning

Ramoni Adeogun, Gilberto Berardinelli
Department of Electronic Systems, Aalborg University, Denmark
E-mail:{ra, gb}@es.aau.dk

***Abstract*—Sixth generation (6G) in-X subnetworks are recently proposed as short-range low-power radio cells for supporting localized extreme wireless connectivity inside entities such as industrial robots, vehicles, and the human body. The deployment of in-X subnetworks in these entities may lead to fast changes in the interference level and hence, varying risks of communication failure. In this paper, we investigate fully distributed resource allocation for interference mitigation in dense deployments of 6G in-X subnetworks. Resource allocation is cast as a multi-agent reinforcement learning problem and agents are trained in a simulated environment to perform channel selection with the goal of maximizing the per-subnetwork rate subject to a target rate constraint for each device. To overcome the slow convergence and performance degradation issues associated with fully distributed learning, we adopt a centralized training procedure involving local training of a deep Q-network (DQN) at a central location with measurements obtained at all subnetworks. The policy is implemented using Double Deep Q-Network (DDQN) due to its ability to enhance training stability and convergence. Performance evaluation results in an in-factory environment indicated that the proposed method can achieve up to 19% rate increase relative to random allocation and is only marginally worse than complex centralized benchmarks.**

***Index Terms*—Machine learning, reinforcement learning, interference management, beyond 5G networks, resource allocation**

I. Introduction

The proliferation of more demanding applications clearly indicates that wireless networks beyond 5G must be designed to cope with more stringent performance requirements in denser environments than current systems. Recent publications on sixth generation (6G) [1]–[3] networks have identified short-range wireless communication for replacing wired connectivity in applications such as industrial control at the sensor-actuator level, augmented- or virtual reality, and intra-vehicle control. Replacing wired connectivity with wireless offers the inherent benefits of higher scalability, lower equipment weight, enhanced flexibility, and lower maintenance cost among others. Clearly, some of these examples are life-critical use cases requiring performance guarantees at all times. Such use cases can also lead to dense scenarios (e.g., in-body subnetworks in a crowded environment) leading to potentially high and dynamic interference footprint. In order to achieve the above requirements, mechanisms for mitigating the adverse effects of interference are important.

Radio resource allocation has been an important component of wireless research for several years as a key framework for interference mitigation. The goal of resource allocation is to optimize specified performance metric(s) (subject to practical constraints on resource availability) by adjusting the utilization of the limited radio resources such as transmit power, frequency channel, and time. Resource allocation typically involves non-convex objective function and is known to be NP-hard with no universal optimal solution [4]. To overcome this limitation, algorithms for resource allocation have been traditionally based on hard-coded heuristics [5] or using optimization techniques such as game theory [6], genetic algorithm [7] and geometric programming [8]. Over the last few years, the focus appears to have shifted towards machine learning-based algorithms [4] resulting in a large number of published works applying supervised [9], unsupervised [10] and reinforcement learning techniques [11] for resource allocation in different types of wireless systems.

While several solutions have been proposed for resource allocation in different wireless systems over the years, works targeting the peculiar nature of short-range low-power 6G in-X subnetworks are still rather limited. In our previous works, we have proposed distributed rule-based heuristics [5], [12] and a supervised learning method [13] in which a deep neural network (DNN) is trained with data generated using centralized graph coloring for channel allocation in scenarios with dense deployment of 6G in-X subnetworks. In a recent work [14], a Q-learning method for joint power and channel allocation using quantized state information is proposed. While the results in this paper highlight the potential of Q-learning for resource allocation, the method suffers from non-scalability to large problem dimensions as well as the effect of state quantization on the performance of Q-learning algorithms. The authors of [15] presented a complex architecture referred to as GA-Net which combines graph attention (GAT) networks, graph neural networks (GNN), and multi-agent reinforcement learning (MARL) for channel allocation in 6G subnetworks. The introduction of multi-head attention for feature extraction allows for only centralized training which requires the transmission of sensing measurements from all subnetworks to a central location translating to high communication overhead and potential security threats. The lack of possibility for distributed training limits the usability of GA-Net in practical applications where connection to a central network may be impossible. Moreover, relying solely on centralized training is not feasible for in-X subnetworks applications (such as in-vehicle or in-body) where privacy constraints may hinder the transmission of raw sensing data to a central server for training. In such cases, methods that are amenable to both distributed and centralized training are desired.

In this paper, we propose a simple, scalable, and robust multi-agent double deep Q-network (MADDQN) method for channel allocation using sensing measurements of the aggregate interference power collected at each subnetwork. The proposed method can be applied for distributed channel allocation with or without the exchange of measurements between subnetworks and is amenable to centralized, distributed, or federated training. We perform extensive simulations to evaluate the performance of the proposed using parameters defined for the in-factory environment. The performance and complexity analysis results show that the MADDQN method can achieve significant performance improvement relative to random allocation and has low computation complexity. The proposed method is also scalable and generalizes well to scenarios with parameters different from those used for training.

The remaining part of this paper is organized as follows. The system model, the distributed channel allocation problem, and a short overview of DQN are presented in Section II. In III, we present the proposed method. Performance evaluation and complexity analysis results are presented in Section IV. Finally, we draw conclusions in Section V.

II. Problem Formulation

A. System Model

We consider a network with N mobile subnetworks each serving M devices. Each subnetwork has a single access point (AP) that coordinates transmission for its associated devices. We index the subnetworks (and hence, APs) with $n \in \mathcal{N} = \{1, 2, \cdots, N\}$ and the devices in each subnetwork with $m \in \mathcal{M} = \{1, 2, \cdots, M\}$. We assume that a total bandwidth, B, which is partitioned into K equal-sized channels is available in the system and that each subnetwork operates on a single channel at each time slot. We index the channels with $k \in \{1, 2, \cdots, K\}$. Denoting the transmit power as p_{tx}, the power received on the link between the nth AP from the mth device in the zth subnetwork is defined as:

$$g^k_{n,z,m}[t] = p_{\mathrm{tx}} |h^k_{n,z,m}[t]|^2 \Gamma^k_{n,z,m} \psi_{n,z,m}, \tag{1}$$

where $h^k_{n,z,m}[t]$, $\Gamma^k_{n,z,m}$ and $\psi_{n,z,m}$ are the Rayleigh distributed complex small scale gain, path-loss, and log-normal shadowing, respectively. By considering Jakes model, the small scale gain, $h^k_{n,z,m}[t]$, is defined as

$$h^k_{n,z,m}[t] = \rho h^k_{n,z,m}[t-1] + \sqrt{1-\rho^2}\epsilon^k_{n,z,m}, \tag{2}$$

where $\epsilon^k_{n,z,m}$ is an iid complex Gaussian variable and ρ is the lag-1 temporal autocorrelation coefficient. The temporal autocorrelation coefficient is modeled as $\rho = J_0(2\pi f_d T_s)$, where $J_0(\cdot)$, f_d and T_s are the zeroth order Bessel function of the first kind, the maximum Doppler frequency, and slot-duration, respectively.

Denoting the corresponding distance as $d_{n,z,m}$, the path-loss component, $\Gamma^k_{n,z,m}$ is expressed as $\Gamma^k_{n,z,m} = c^2 d^{-\beta}_{n,z,m}/16\pi^2 f_k^2$, where $c \approx 3 \times 10^8$ ms^{-1} is the speed of light, f_k and α are the center frequency of channel k and the path-loss exponent, respectively. We compute the log-normal shadowing using [16]

$$\psi_{n,z,m} = \ln\left\{ \frac{1 - e^{\left(-\frac{d_{n,z,m}}{d_c}\right)}}{\sqrt{2}\sqrt{1 + e^{\left(-\frac{d_{n,z,m}}{d_c}\right)}}} \left(\mathbf{S}_n + \mathbf{S}_{z,m}\right) \right\}, \tag{3}$$

where $\mathbf{S}_x$ is the value of a two-dimensional Gaussian random process with exponential covariance at the location of the device or AP, and d_c denotes the de-correlation distance.

At slot, t, the signal-to-noise-plus-interference ratio (SINR) on the link between the AP in subnetwork n and its mth device can be expressed as

$$\gamma^k_{nm}[t] = \frac{g^k_{n,n,m}[t]}{\sum_{n' \in \mathcal{I}_{nn'}} g^k_{n,n',m'}[t] + \sigma^2} \tag{4}$$

where $\mathcal{I}_{nn'}$ denotes the set of all other subnetworks that are operating on the same channel as the nth subnetwork and $\sigma^2 = 10^{(-174 + n_{\mathrm{f}} + 10\log_{10}(\mathrm{BW}))/10}$ is the noise power with n_{f} and BW denoting the noise figure and channel bandwidth, respectively. Assuming single antenna at both the APs and devices and considering the Shannon approximation, the achieved rate at slot t can then be written as

$$\zeta_{nm}[t] \approx \log_2(1 + \gamma_{nm}[t]). \tag{5}$$

B. Distributed Resource Allocation Problem

We consider a resource allocation problem involving fully distributed selection of frequency channels. We consider in-X subnetworks supporting applications that require high data rates with or without minimum rate constraints. The resource optimization problem can then be defined as a constrained multi-objective task involving the maximization of N objective functions, one for each subnetwork. To support the requirement, we take the objective function as the per subnetwork sum-rate subject to a minimum rate per device constraint. The problem can be formally expressed as:

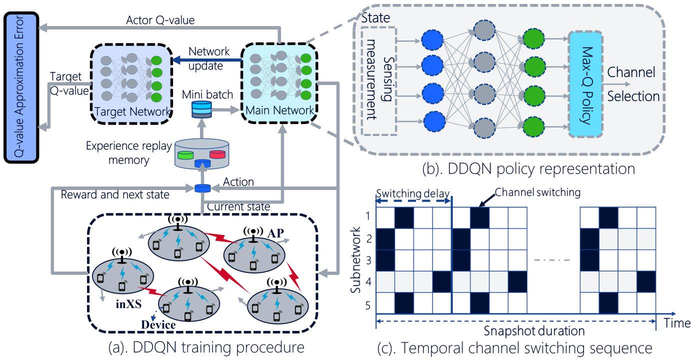
$$\mathrm{P} : \left\{ \max_{\{\mathbf{c}^t\}} \sum_{m=1}^{M} \zeta_{nm}(\mathbf{c}^t) \right\}_{n=1}^{N} \quad \text{st:} \quad \zeta_{nm} \geq \zeta_{\text{target}} \quad \forall n, m \tag{6}$$

where $\mathbf{c}^t = [c^t_1 \cdots c^t_N]$; $c^t_n \in \{1, 2, \cdots, K\} \, \forall n$ denotes the vector of indices of the channel selected by all subnetworks at time, t and ζ_{target} is the target minimum rate which is assumed equal for all subnetworks. The problem in (6) involves joint optimization of N conflicting non-convex objective functions and is known to be difficult to solve. A multi-agent reinforcement learning method for solving the problem is proposed in this paper.

C. Deep Q-Learning Fundamentals

In deep Q-learning, a deep neural network often called Deep Q-Network (DQN) is used to approximate the Q-function. The DQN circumvents the limitations associated with its table-based counterpart and has been shown to provide better performance. The DQN can be expressed as

$$\hat{Q}(s, a) = f(s, a, \boldsymbol{\theta}), \tag{7}$$



(a). DDQN training procedure

(b). DDQN policy representation

(c). Temporal channel switching sequence

Fig. 1: Illustration of the MADDQN-based channel allocation.

where f is a function determined by the DQN architecture and $\boldsymbol{\theta}$ is a vector of the DQN parameters. The Q-value estimation is now reduced to optimization of $\boldsymbol{\theta}$. This optimization is typically performed using standard gradient descent algorithms with the Huber loss defined as [17]

$$\mathcal{L}(\theta) = \begin{cases} (\Gamma(\theta))^2 & \text{if } |\Gamma(\theta)| \leq \delta \\ \delta|\Gamma(\theta)| - \frac{1}{2}\delta^2 & \text{otherwise} \end{cases} \tag{8}$$

where $\Gamma = r(s_t, a) + \gamma \max_{a'} Q^{'}(s_{t+1}, a^{'}; \theta) - Q(s_t, a; \theta)$ is the difference between expected and predicted Q-values and δ is the discriminating parameter of the loss function.

III. Multi-agent DDQN for Channel Allocation

We cast the resource selection described above in a MARL framework in which each subnetwork has an agent at the AP whose goal is to learn a policy for selecting a frequency channel such that its communication requirements are met via interaction with the wireless environment as shown in Fig. 1a. As with other RL techniques, MARL requires the definition of the environment, state (or feature) space, action space, and reward signal as well appropriate model for the policy. As described in section II-A, a wireless environment with N mobile subnetworks each serving M devices is considered. The other components are described below.

A. State space

We consider two cases viz: fully independent resource selection and resource selection with limited cooperation. In the former, no communication is possible among subnetworks. Each subnetwork, therefore, makes resource selection decisions based solely on its local sensing information. The latter allows communication of only sensing measurement between a subnetwork and others in its neighbour set, denoted as $\mathcal{D}_n$ for the nth subnetwork. The feature set of subnetwork n is represented as

$$\mathcal{S}_n = \{I_{z,1}, I_{z,2}, \cdots, I_{z,K}\} \quad \forall z \in \{n, \mathcal{D}_n\} \tag{9}$$

where $I_{z,k}$ is the measured aggregate interference power on channel k at the zth subnetwork. Note that the dimension of the neighbour set, $|\mathcal{D}_n|$ can be varied between 0 and $N-1$ to control the number of neighbours from which each subnetwork receives state information. If $|\mathcal{D}_n| = 0$, we have the fully independent learning case. With $|\mathcal{D}_n| < N-1$, the strongest interfering subnetworks are included $|\mathcal{D}_n|$.

B. Action space

The action space is the set of all possible actions that the agent can choose from at each time. While the method presented here can be applied to the selection of any wireless resource, we consider the allocation of frequency channels.

The action space for each subnetwork is therefore the set of all available frequency channels defined as

$$\mathcal{A} = \{c_1, c_2, \cdots, c_K\}, \tag{10}$$

where c_k denotes the k channel. At each time, the nth subnetwork's action is denoted $a_n^t; a_n^t \in \mathcal{A}$.

C. Reward signal

As stated in section II-B, the goal of each agent is to maximize the achieved rate while also ensuring that a target rate, r_{target} is achieved. To guide the agent towards achieving this goal, we define the reward function considering the optimization problem defined in (6). The reward for the nth subnetwork at time, t is defined as

$$r_n = \begin{cases} \zeta_n & \text{if } \zeta_{nm} \geq \zeta_{\text{target}}, \forall n, m \\ \zeta_n - \lambda\Delta\zeta_n & \text{otherwise} \end{cases}, \tag{11}$$

where $\zeta_n = \sum_{m=1}^{M} \zeta_{nm}$ is the sum rate achieved by all devices in subnetwork n, $\Delta\zeta_n = \sum_{m=1}^{M}(\zeta_{\text{target}} - \zeta_{nm})$ and λ is a control parameter which is set to ensure a balance between maximizing the achieved rate and guaranteeing that the minimum rate is at least equal to ζ_{target}.

D. Policy Representation

Motivated by the work in [18] where it was shown that a DQN-variant referred to as Double DQN (DDQN) offered up to 2-fold performance improvement and better training stability than classic DQN, we adapt the DDQN with experience replay [19] in a multi-agent version for channel selection. The considered DDQN architecture is shown in Fig. 1. The DDQN comprises two networks viz:

- Main Network: The main network acts as the action-value function approximator which maps the features to actions. This mapping for the nth subnetwork is denoted as $Q(\mathbf{s}_t, a_k; \theta_t) : \mathbf{s}_t \rightarrow \{q(a|\mathbf{s}_t, \theta_t)|a \in \mathcal{A}\}$, where $q(a|\mathbf{s}_t, \theta_t)$ denotes the expected cumulative rewards for taking action a at state, $\mathbf{s}_t$.
- Target Network: In DDQN, the target network is used for estimating expected returns from choosing an action at a given state as shown in Fig. 1. Estimates of the expected reward are then used to compute the Q-value approximation error while performing optimization of the main network. We denote the target network as $\tilde{Q}(\mathbf{s}_t, a_k; \tilde{\theta}_t)$. The target network has the same structure as the main network but its weights are only updated after a specified number of steps, T_{update}, i.e., $\tilde{\theta}_t := \theta_t$ every T_{update} steps.

Algorithm 1 Training of MADDQN-based channel allocation

```
**Input**: Learning rate, $\alpha$, discount factor, $\gamma$, number of episodes, $T$, number of episode steps, $N_e$, batch size, $N_\mathrm{b}$, target network update interval, $T_\mathrm{up}$, switching delay, $\tau_\mathrm{delay}$
Compute initial states, $\{\mathbf{s}_n^1\}_{n=1}^N$
Initialize replay memory, $\{\mathcal{D}_n\}_{n=1}^N$, main network parameters, $\{\boldsymbol{\theta}_n\}_{n=1}^N$, target network parameters $\tilde{\boldsymbol{\theta}}_n = \boldsymbol{\theta}_n$
for $t = 1$ to $T$ do
  Generate random switching index, $\{\tau_n\}_{n=1}^N$
  for $i = 1$ to $N_e$ do
    for $n = 1$ to $N$ do
      if $i$ modulo $\tau_\mathrm{delay} == \tau_n$ then
        subnetwork $n$ obtain feature vector, $\mathbf{s}_n^t$
        subnetwork $n$ select $a_n^t$ using $\epsilon$-greedy strategy
      end if
    end for
    The joint resource selection of all subnetworks yield
    transitions into next states, $\{\mathbf{s}_n^{t+1}\}_{n=1}^N$ and
    immediate rewards, $\{r_n(\mathbf{s}^t, \mathbf{a})\}_{n=1}^N$
    if $i$ modulo $\tau_\mathrm{delay} == \tau_n$ then
      Store experience samples $(\mathbf{s}_n^t, a_n^t, r_n^t, \mathbf{s}_n^{t+1})$ in replay
      memory $\mathcal{D}_n; \forall n \in \{1, \cdots, N\}$
    end if
    Decay exploration probability as in (12).
    if $t$ modulo $N_\mathrm{b} == 0$ then
      for $n = 1$ to $N$ do
        Randomly choose a mini-batch, $(\mathbf{s}_n^\tau, a_n^\tau, r_n^\tau, \mathbf{s}_n^{\tau+1})$
        Perform gradient descent to minimize (8)
      end for
    end if
    if $t$ modulo $T_\mathrm{up} == 0$ then
      Update target networks: $\tilde{\boldsymbol{\theta}}_n = \boldsymbol{\theta}_n; \forall n \in \{1, \cdots, N\}$
    end if
  end for
end for
**Output**: Trained DQNs, $\{Q_n\}_{n=1}^N$
```

E. Action Selection

During the training, resource selection decision is made by each agent via the ϵ-greedy strategy [17], where ϵ is the exploration probability, i.e., the probability that the agent takes random action. During the training, ϵ is decayed according to

$$\epsilon = \max\left(\epsilon_\mathrm{min}, (\epsilon_\mathrm{max} - \epsilon_\mathrm{min})/\epsilon_\mathrm{step}\right), \tag{12}$$

where ϵ_min and ϵ_max denote the minimum and maximum exploration probability, respectively, and ϵ_step is the number of exploration steps. The multi-agent training procedure is described in Algorithm 1.

IV. Performance Evaluation

A. Simulation settings

We consider a network with $N = 25$ subnetworks each with a single controller serving as the AP for a sensor-actuator pair. Subnetworks are uniformly distributed in a 40 m × 40 m rectangular area leading to a deployment density of 15625 subnetworks/km^2. Each subnetwork moves according to a restricted random direction mobility with a velocity, $v = 2$ m/s translating to a Doppler frequency, $f_\mathrm{d} = 40$ Hz. We assume that transmissions occur over a bandwidth, $B = 10$ MHz. Except where stated otherwise, we set the number of frequency channels, $K = 4$, and the transmit power, $P_\mathrm{tx} = 0$ dBm. Without loss of generality, we consider a single device per subnetwork, i.e., $M = 1$. Other simulation parameters are listed in Tab. I.

TABLE I: Default simulation parameters.

Parameter	Value	Parameter	Value
Deployment area [m^2]	40 × 40	Number of subnetworks, N	25
subnetwork radius [m]	3.0	Velocity, v [m/s]	2.0
Number of frequency channels, $\lvert\mathcal{A}\rvert$	4	Pathloss exponent, γ	2.7
Shadowing standard deviation, σ_s [dB]	5	Carrier frequency [GHz]	6
Transmit power [dBm]	0	Noise figure [dB]	10
Channel bandwidth [MHz]	10	Network structure	$\lvert S\rvert - 24 - 24 - \lvert\mathcal{A}\rvert$
Optimizer	Adam	Learning rate	0.001
Batch size	500	Number of training episodes	2000
Initial/final Epsilon	1/0.01	Discount factor, γ	0.99

B. DDQN Design and Training Procedure

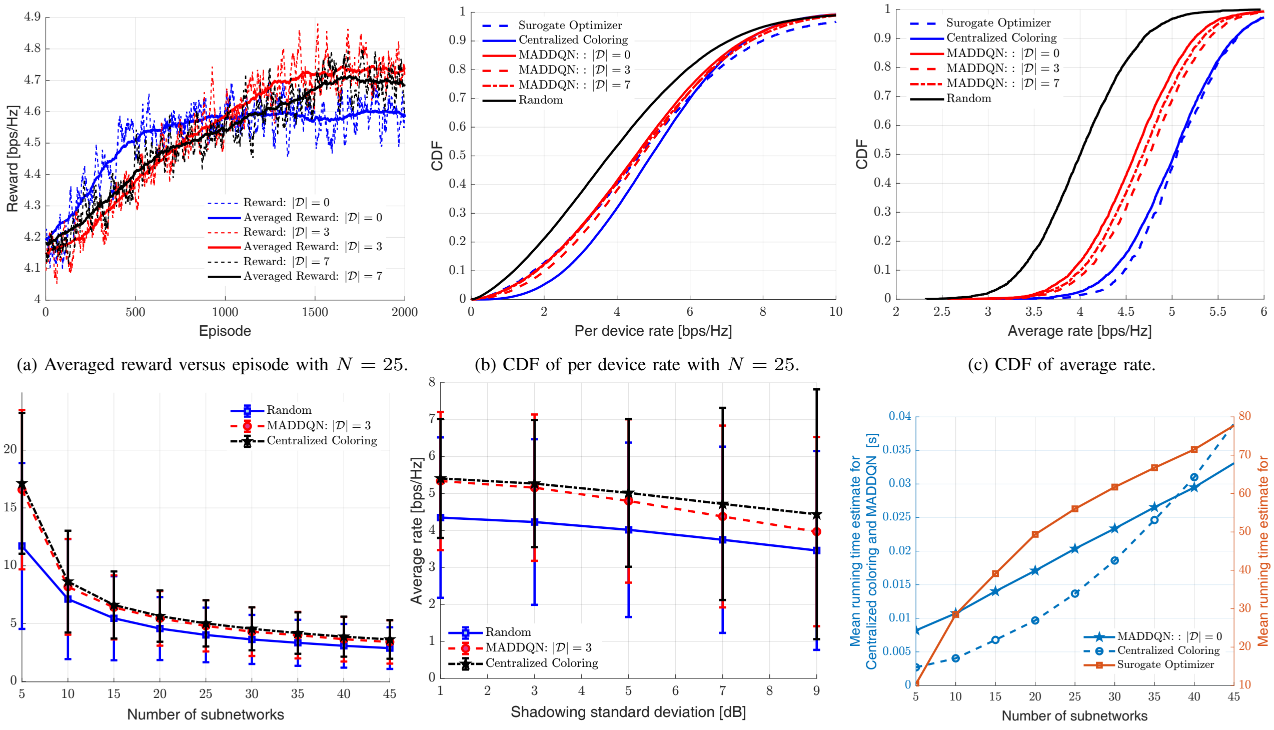
The main and the target DDQN policy are implemented as fully connected neural network (FCNN) architectures with two hidden layers each with 24 neurons in MATLAB[1].

We studied both *distributed training and execution* approaches in which N agents are trained simultaneously and the *centralized training with distributed execution* which involves training a single agent and copying its weights to other agents either during the training or at convergence. The goal is to understand the potential of both training mechanisms for the channel selection problem. Our initial results showed that distributed training results in excessively long training time. With $N = 25$ subnetworks, it took approximately 12× longer (time to convergence of about 44 hours) on a quad-core laptop with 8 GB RAM to train the agents in a distributed version compared to centralized training which took about 3.8 hours on the same machine. The distributed training procedure is summarized in Algorithm 1. The centralized approach follows the same procedure except that only a single agent is applied at all subnetworks during training.

To achieve stability and improve convergence, *experience replay technique* is used to store previous experiences in a replay buffer. Samples for updating the DDQN weights are then drawn randomly from the buffer thereby eliminating correlations between successive samples. The agents are trained using the reward function in (11) with $\zeta_\mathrm{target} = 0$ bps/Hz.

Similar to the works in [5], [12], we utilized random switching delays to minimize the impact of *ping-pong* effects resulting from simultaneous switching by multiple subnetworks to the same channel. The delay is generated for all subnetworks at the beginning of each snapshot as a random integer factor of the transmission interval with a maximum value of 10. A subnetwork is then allowed to perform channel switching at time instants determined by its assigned delay value.

[1] The implementation are available via the GitHub repository available via https://github.com/MADDQN-based-subnetwork-channel-allocation.git



(a) Averaged reward versus episode with $N = 25$.

(b) CDF of per device rate with $N = 25$.

(c) CDF of average rate.

(d) Sensitivity to the number of subnetworks.

(e) Sensitivity to shadowing standard deviation

(f) Running time estimates

Fig. 2: Plots of the learning curves (a), performance (b-C) and sensitivity evaluation (d-e) results, and running time estimates (f).

C. Simulation Results

1) Training: Fig. 2a shows the averaged reward over successive episodes with no target rate constraint, i.e., $\zeta_{\text{target}} = 0$ bps/Hz and size of neighbor set for each subnetwork, $|D| = [0, 3, 7]$. The averaging is performed over all steps within each episode and all subnetworks. The figure shows that convergence is achieved at approximately 1000 episodes with fully independent, i.e., $|\mathcal{D}| = 0$ and 1600 episodes with $|\mathcal{D}| = 3$ and $|\mathcal{D}| = 7$. This indicates that an agent requires longer training to learn the feature-to-action mapping function using sensing measurements from multiple subnetworks than using only local measurements. At convergence, averaged reward of about 4.60 bps/Hz, 4.75 bps/Hz and 4.70 bps/Hz is achieved with $|\mathcal{D}| = 0$, $|\mathcal{D}| = 3$ and $|\mathcal{D}| = 7$, respectively, indicating marginal improvement of 3.3% with $|\mathcal{D}| = 3$ and 2.2% with $|\mathcal{D}| = 7$ compared to the fully independent case, i.e. $|\mathcal{D}| = 0$.

2) Execution: The trained DDQN agents are deployed for distributed channel allocation and performance compared with three benchmark algorithms viz:

1) Random: assign frequency channels randomly to all subnetworks at the start of a snapshot.
2) Mixed Integer Surrogate Optimizer: the surrogate optimization method [20] is applied in a centralized version to the mixed integer problem involving maximization of the network sum rate. This method is implemented using the *surrogateopt* function in MATLAB with default parameters except for the number of iterations which is set to 400.
3) Centralized coloring: Greedy graph coloring is applied to the interference graph, G created from the matrix of mutual interference power between subnetworks with a $K - 1$ strongest interfering neighbours edge constraint. To guarantee colorability G, the successive graph sparsification involving removal of the weakest edges until no more than K colors are required [12] is used in the simulations.

Fig. 2b shows the empirical Cumulative Distribution Function (CDF) of the per-device rate for the different methods. The proposed MADDQN scheme performs better than the random channel allocation, similar to centralized coloring, and only marginally worse compared to the iterative surrogate optimization technique.

The averaged rate (or equivalently sum rate) performance of the different channel allocation methods is shown in Fig. 2c where we plot the CDF of the rate averaged over all subnetworks. Compared to random allocation, the proposed MADDQN method offers between $\sim 15\%$ (with $|\mathcal{D}| = 0$) and $\sim 19\%$ (with $|\mathcal{D}| = 3$) improvement at the median of the average rate distribution and is only about $\sim 6\%$ below the median average rate achieved by the centralized benchmark schemes, i.e., centralized coloring and surrogate optimizer. We remark here that the proposed method offers the advantage of much lower signaling overhead since only a very limited exchange of information is required.

3) Sensitivity Evaluation: We study the robustness of the proposed method to changes in the wireless environment than those used during the training. Due to its high computation complexity, the iterative surrogate optimizer is not included in the sensitivity evaluation. The MADDQN model trained with $N = 25$ subnetworks and shadowing standard deviation of $\sigma_s = 5$ dB is evaluated with values of N between 5 and 45 in the same $40 \text{ m} \times 40 \text{ m}$ and σ_s between 1 dB and 9 dB. We plot the mean and standard deviation of the average rate as a function of the number of subnetworks in Fig. 2d and shadowing standard deviation in Fig. 2e. In both cases, the MADDQN method shows a similar trend as well as relative performance to the centralized coloring and random allocation benchmarks indicating that all schemes are equally affected by the changes in the number of subnetworks and shadowing standard deviation. It is therefore reasonable to conclude that the proposed scheme is robust to changes in the considered wireless parameters.

4) Complexity Analysis: We compare the computational complexity of the proposed MADDQN method with the benchmark algorithms by estimating the total time required to perform channel allocation for all subnetworks at each transmission instant. In Fig. 2f, we plot the averaged total running time per step as a function of the number of subnetworks. The figure shows that the proposed MADDQN and our implementation of greedy coloring can provide up to a factor of 2000 reduction in time complexity relative to the iterative surrogate optimizer. While the running time for centralized coloring is marginally lower than that of MADDQN for values of N between 5 and 35, the linear growth achieved by the latter makes it more attractive for deployments with higher number of subnetworks, i.e., $N \geq 40$.

Note that the distributed MADDQN method has minimal signaling overhead compared to the centralized benchmarks. Assuming a constant time cost for exchanging sensing measurement between any pair of subnetworks or from a subnetwork to the central resource manager, the signaling complexity for MADDQN and centralized benchmarks (i.e., centralized coloring and surrogate optimizer) is upper bounded by $\mathcal{O}(N|\mathcal{D}_n|)$ and $\mathcal{O}(N^2)$, respectively. As observed from the training curves in Fig. 2a and the mean rate performance in Fig. 2c, no performance improvement is achieved with values of $|\mathcal{D}_n| > K - 1$. In practical interference-limited scenarios, the number of available channels, K is much less than the number of subnetworks. i.e., $N << K$ and hence the signalling cost complexity for MADDQN reduces to $\mathcal{O}(N)$.

V. Conclusion

A simple multi-agent DDQN (MADDQN) approach is proposed for fully distributed dynamic channel allocation in dense deployments of 6G in-X subnetworks. The access point in each subnetwork act as the DDQN agent which dynamically makes channel selection decisions based on aggregate interference power per channel measurements obtained via sensing. The presented performance results indicated that DDQN agents for channel allocation can be trained with reasonably fast convergence. The MADDQN approach yields a median average rate that is up to 19% higher than baseline random allocation and only about 6% lower than the computational intensive surrogate optimizer as well as the centralized graph coloring with high signaling overhead. Our results further indicated that the proposed method is robust to changes in the deployment density as well as propagation parameters.

References

[1] V. Ziegler, H. Viswanathan, H. Flinck, M. Hoffmann, V. Räisänen, and K. Hätönen, “6G architecture to connect the worlds,” *IEEE Access*, vol. 8, pp. 173 508–173 520, 2020.

[2] H. Viswanathan and P. E. Mogensen, “Communications in the 6G Era,” *IEEE Access*, vol. 8, pp. 57 063–57 074, 2020.

[3] G. Berardinelli, P. Baracca, R. Adeogun, S. Khosravirad, F. Schaich, K. Upadhya, D. Li, T. B. Tao, H. Viswanathan, and P. E. Mogensen, “Extreme Communication in 6G: Vision and Challenges for ‘in-X’ Subnetworks,” *IEEE OJCOM*, 2021.

[4] F. Hussain, S. A. Hassan, R. Hussain, and E. Hossain, “Machine Learning for Resource Management in Cellular and IoT Networks: Potentials, Current Solutions, and Open Challenges,” *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1251–1275, 2020.

[5] R. Adeogun, G. Berardinelli, I. Rodriguez, and P. E. Mogensen, “Distributed Dynamic Channel Allocation in 6G in-X Subnetworks for Industrial Automation,” in *IEEE Globecom Workshops*, 2020.

[6] R. O. Adeogun, “A novel game theoretic method for efficient downlink resource allocation in dual band 5G heterogeneous network,” *Wireless Personal Communications*, vol. 101, no. 1, pp. 119–141, Jul 2018.

[7] U. Mehboob, J. Qadir, S. Ali, and A. Vasilakos, “Genetic algorithms in wireless networking: techniques, applications, and issues,” *Soft Computing*, vol. 20, no. 6, pp. 2467–2501, 2016.

[8] K. T. Phan, T. Le-Ngoc, S. A. Vorobyov, and C. Tellambura, “Power allocation in wireless relay networks: A geometric programming-based approach,” in *IEEE GLOBECOM 2008 - 2008 IEEE Global Telecommunications Conference*, 2008, pp. 1–5.

[9] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, “Learning to Optimize: Training Deep Neural Networks for Interference Management,” *IEEE Transactions on Signal Processing*, vol. 66, no. 20, pp. 5438–5453, 2018.

[10] C. Sun and C. Yang, “Learning to Optimize with Unsupervised Learning: Training Deep Neural Networks for URLLC,” in *IEEE PIMRC*, 2019, pp. 1–7.

[11] J. Burgueno, R. Adeogun, R. L. Bruun, C. S. M. García, I. de-la Bandera, and R. Barco, “Distributed Deep Reinforcement Learning Resource Allocation Scheme For Industry 4.0 Device-To-Device Scenarios,” in *IEEE VTC-Fall)*. IEEE, 2021, pp. 1–7.

[12] R. Adeogun, G. Berardinelli, and P. E. Mogensen, “Enhanced interference management for 6G in-X subnetworks,” *IEEE Access*, vol. 10, pp. 45 784–45 798, 2022.

[13] R. O. Adeogun, G. Berardinelli, and P. E. Mogensen, “Learning to Dynamically Allocate Radio Resources in Mobile 6G in-X Subnetworks,” in *IEEE PIMRC*, 2021.

[14] R. Adeogun and G. Berardinelli, “Multi-agent dynamic resource allocation in 6G in-X subnetworks with limited sensing information,” *Sensors*, vol. 22, no. 13, p. 5062, 2022.

[15] X. Du, T. Wang, Q. Feng, C. Ye, T. Tao, L. Wang, Y. Shi, and M. Chen, “Multi-agent reinforcement learning for dynamic resource management in 6G in-X subnetworks,” *IEEE Transactions on Wireless Communications*, pp. 1–1, 2022.

[16] S. Lu, J. May, and R. J. Haines, “Effects of correlated shadowing modeling on performance evaluation of wireless sensor networks,” in *IEEE Vehicular Technology Conference*, 2015, pp. 1–5.

[17] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. The MIT Press, 2018.

[18] H. van Hasselt, A. Guez, and D. Silver, “Deep reinforcement learning with double q-learning,” *CoRR*, vol. abs/1509.06461, 2015. [Online]. Available: http://arxiv.org/abs/1509.06461

[19] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing Atari with deep reinforcement learning,” 2013.

[20] H.-M. Gutmann, “A radial basis function method for global optimization,” *Journal of global optimization*, vol. 19, no. 3, pp. 201–227, 2001.