Deep Reinforcement Learning for Joint User Association and Resource Allocation in Factory Automation

Mohammad Farzanullah*, Hung V. Vu† and Tho Le-Ngoc*
* Department of Electrical & Computer Engineering, McGill University, Montréal, QC, Canada
† Huawei Technologies Canada, Ottawa, ON, Canada

2022 IEEE Wireless Communications and Networking Conference (WCNC) | 978-1-6654-4266-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/WCNC51071.2022.9771770

***Abstract*—We consider the problem of joint user association, channel assignment, and power allocation for mobile robot application in factory automation system that require ultra-reliable and low latency communications (URLLC). The aim is to deliver control commands from the controller to mobile robots with stringent requirements of latency and reliability. To achieve URLLC, we develop a two-phase communication scheme. The robots work close to each other in a factory environment and can form clusters for reliable device-to-device (D2D) communications. Within the latency requirements, the combined payload of a cluster is transmitted to the leader in Phase I. In Phase II, the leader broadcasts the payload to its members. Under this strategy, we use multi-agent reinforcement learning (MARL) for resource allocation. The cluster leaders in Phase I act as the agents and interact with the environment to optimally select the Access Point (AP) for connection along with the sub-band and power level. The objective is to maximize the successful payload delivery probability to all the robots. Illustrative simulation results indicate that the proposed scheme can offer average successful payload delivery probability close to that of centralized exhaustive search algorithm.**

***Index Terms*—Factory Automation, Reinforcement Learning, Resource Allocation**

I. Introduction

The purpose for the development of 5G was not restricted to the enhancement of mobile broadband. Apart from mobile broadband, it aimed to provide connectivity for various sectors such as manufacturing, industrial and automotive industries [1]. To achieve a diverse set of requirements, 5G New Radio (NR) supports three main use cases, which are Enhanced Mobile Broadband (eMBB), massive machine-type communications (mMTC), and ultra-reliable and low latency communications (URLLC) [2]. The URLLC has stringent requirements with regard to latency, reliability, and availability. URLLC is relevant to many emerging applications such as vehicle-to-vehicle (V2V) communication in the automotive industry, wireless control and automation in the industrial environment, and tactile internet [3]. The existing 4G architecture cannot provide URLLC services since it offers reliability of 95% with a latency of 15 ms [3]. The general requirement for URLLC for a single transmission is to transmit a packet of 32 bytes with reliability of $1 - 1 \times 10^{-5}$ [3].

Industry 4.0 aims to revolutionize and enhance industrial production. It integrates the manufacturing process with the Industrial Internet of Things (IIoT) to increase the flexibility and efficiency of future smart factories [4]. The existing control systems use wired connections since 4G, and previous generations of wireless communications were unable to provide the required reliability and latency. Industry 4.0 aims to replace wired connections with wireless connectivity. Factory automation will be the main area of interest and it includes automated control, monitoring, and optimization of processes and workflows within a factory. This will enable industrial mass production with high quality and low costs. The communication in the factory of the future will require low latency coupled with high reliability and availability. There will be a large number of mobile robots/assets in the factory that will require URLLC [5]. Within the factory automation system, one of the use-cases is about mobile robots. An example of a mobile robot will be the automated guided vehicle (AGV), which will be used to transport goods and materials efficiently within a factory setting. A mobile robot can be programmed to follow a certain path in order to complete different activities or tasks. The mobile robots require radio-controlled guidance to avoid collisions and manage traffic. Mobile robots have stringent requirements with regard to latency and service availability [5]. These requirements necessitate an intelligent resource allocation (RA) design.

Due to the stringent latency requirements in a factory setting with mobility, the optimization techniques that use a centralized controller will not be practical. This is due to the fast channel variations because of mobility, and the increase in latency to send channel state information (CSI) to the central controller. Further, there is high CSI overhead and the algorithm does not scale well to a large number of users. Recently, the distributed approach has been considered extensively [6], [7].

Traditional optimization algorithms have been used for RA in factory automation systems [8]–[10]. Recently, Reinforcement Learning (RL) has been used for solving the RA optimization problems in multiple domains, e.g., in Cellular Vehicle-to-Everything (C-V2X) [7], and cellular communications [6], [11]. In the context of factory automation, [12] considered a multi-connectivity problem, together with dynamic channel allocation for interference management in a so-called campus network where the wireless communications

978-1-6654-4266-4/22/$31.00 ©2022 IEEE
Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 12,2023 at 14:15:44 UTC from IEEE Xplore. Restrictions apply.

in the factory are impaired by the external interference. Reinforcement learning is adopted to resolve the joint user association and channel assignment problem. In their system model, there is a central network manager that regulates the channel allocation between the Access-Points (APs) and the actuators.

Inspired from [8], [13], [14], we develop a two-phase transmission scheme in a mobile robot scenario in an industrial setting. Mobile robots are monitored and controlled by a central controller/guidance system. In the smart factory environment, all mobile robots and the guidance control system are connected through the 5G network. We assume that each cluster includes one leader and multiple members. The leader receives control messages from APs and relays these messages to all members via point-to-multipoint (i.e., star) device-to-device (D2D) connections. A typical factory setting could consist of hundreds of devices. In such a scenario of massive connectivity, it would be difficult to achieve URLLC with conventional broadcasting techniques. The devices will face high interference, especially those located far from the AP. This would make it difficult to transmit to each user reliably. The robots in a factory/warehouse setting are generally close to each other. Due to their proximity, they can form clusters for reliable D2D communications. In Phase I, within the latency requirement, the APs combine the messages of all robots within each group together and transmit them to the corresponding group leader. In Phase II, the leaders broadcast the messages to the other robot members in their cluster on separate time-frequency slots.

This paper uses multi-agent reinforcement learning (MARL) in a distributed approach for Phase I of the transmission. Each of the cluster leaders acts as an agent and performs user association, channel assignment, and power allocation based on the interaction with the environment. The agents use only their local CSI and learn to optimally allocate resources after trial-and-error interaction with the factory environment. The optimization RL problem is based on deep Q-learning, which was first used by DeepMind for video game control [15]. Our optimization objective is the reliability, which is formally defined as the average probability of successful payload delivery within a required latency. Simulation results indicate that the proposed MARL algorithm can offer the successful payload delivery probability close to that of the exhaustive search algorithm [16]. Exhaustive search is a brute-force approach which examines all possible candidates to select the best action.

II. System Model and Problem Formulation

As illustrated in Fig. 1, we consider downlink communication in a single-floor factory setting. The factory consists of K Access Points (APs). The APs deliver control messages to all the mobile robots. It is assumed that the robots are in close proximity and form N clusters for reliable communications. Each cluster n contains one leader robot and O member robots. We denote the set of members in cluster n as $\mathcal{V}_n$, $n = 1, \ldots, N$, i.e., there are N clusters with O members

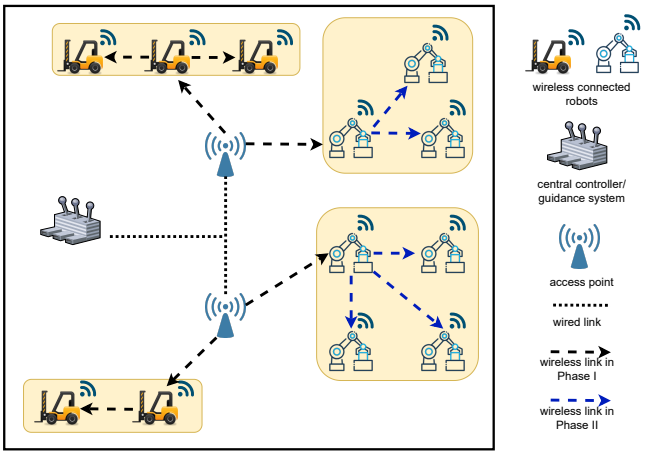

Fig. 1. Illustrative factory/warehouse automation system with multiple clusters (highlighted) of mobile robots including AGVs and robot arms. In Phase I, each cluster leader autonomously selects the AP along with the sub-band and transmit power level. In Phase II, the leader groupcasts the messages to members via *point-to-multipoint* connections.

in each cluster. In addition, each member can be part of just one cluster, i.e., $\mathcal{V}_n \cap \mathcal{V}_i = \emptyset$ for $n \neq i$. The members are located close to each other and can have a maximum possible distance d from the cluster leader. Each robot requires delivery of payload of B_o bytes within a latency of T seconds. Further, we denote the set of APs as $\mathcal{K}$ and the set of available sub-bands as $\mathcal{M}$.

We consider a two-phase communication approach. The leader receives control messages from APs and relays these messages to all members via point-to-multipoint (i.e., star) D2D connections. During the first phase of time T_1 seconds, the AP associated with a cluster n combines the payload for the robots in the cluster and transmits it to the cluster leader. This payload consists of $B = B_o \times (O + 1)$ bytes. In the second phase of duration T_2 seconds, the cluster leader broadcasts this message to its members. Each robot uses a single antenna to transmit/receive signals. The APs have M sub-bands to communicate on. It is noted that $M < N$, resulting in spectrum sharing during Phase I.

A. Phase I

Phase I consists of the communication from the AP to the cluster leader. The leader needs to decide the AP ($k \in \mathcal{K}$) it would be served by along with the sub-band ($m \in \mathcal{M}$) and transmit power. We assume that the Channel State Information (CSI) is available at the leader. An AP k can serve multiple cluster leaders, however a cluster leader n can be served by a single AP. Each cluster leader will experience interference from other AP-leader links that are communicating in the same sub-band. The signal-to-interference-plus-noise-ratio (SINR) at leader n in sub-band m, being served by AP k is given by:

$$\text{SINR}_{kn}^{(c)} = \frac{P_{kn}^{(c)} L_{kn}^{(c)} g_{kn}^{(c)}[m]}{\sum_{i \in K} \sum_{j \neq n} \rho_{ij}^{(c)}[m] P_{ij}^{(c)} L_{ij}^{(c)} g_{ij}^{(c)}[m] + \sigma^2} \quad (1)$$

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 12,2023 at 14:15:44 UTC from IEEE Xplore. Restrictions apply.

where superscript (c) signifies communication in Phase I. $P_{kn}^{(\mathrm{c})}$ ($P_{ij}^{(\mathrm{c})}$) denotes the power from transmitter k (i) to receiver n (j) in sub-band m, and σ^2 refers to the noise power at the receiver. $g_{kn}^{(\mathrm{c})}$ ($L_{kn}^{(\mathrm{c})}$) refers to the small-scale (large-scale) fading power gain in the AP-leader link kn. Similarly, $g_{ij}^{(\mathrm{c})}$ ($L_{ij}^{(\mathrm{c})}$) refers to the small-scale (large-scale) fading gain from the AP i to cluster leader j. We consider that the small-scale fading gain follows exponential distribution with unit mean. The large-scale fading consists of both path-loss and shadowing. $\rho_{ij}^{(\mathrm{c})}[m]$ indicates the indicator function defined as:

$$\rho_{ij} = \begin{cases} 1, & \text{if link } ij \text{ reuses the sub-band } m, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The achievable throughput of leader n in sub-band m is given by:

$$R_n^{(\mathrm{c})} = W \log_2(1 + \mathrm{SINR}_{kn}^{(\mathrm{c})}) \tag{3}$$

where W is the bandwidth of each spectrum sub-band.

B. Phase II

Phase II consists of the communication from the cluster leader to members. The cluster leader broadcasts the payload received to all of the members. We assume that each member can decode its message from the combined payload. A member o will be affected by the interference from the leaders that transmit using the same sub-band. The SINR at the receiver of member $o \in O$ is given by:

$$\mathrm{SINR}_{no}^{(\mathrm{d})} = \frac{P_{no}^{(\mathrm{d})} L_{no}^{(\mathrm{d})} g_{no}^{(\mathrm{d})}[m]}{\sum_{i \neq n} \rho_{io}^{(\mathrm{d})}[m] P_{io}^{(\mathrm{d})} L_{io}^{(\mathrm{d})} g_{io}^{(\mathrm{d})}[m] + \sigma^2} \tag{4}$$

where where superscript (d) signifies communication in Phase II. $P_{no}^{(\mathrm{d})}$ ($P_{io}^{(\mathrm{d})}$) denotes the power from transmitter n (i) to receiver o in sub-band m.

The throughput of member o of cluster n is given by:

$$R_o^{(\mathrm{d})} = W \log_2(1 + \mathrm{SINR}_{no}^{(\mathrm{d})}) \tag{5}$$

C. Problem Formulation

In this work, we aim to maximize the reliability of transmission to each robot in a factory setting. Each robot leader and member is required to reliably receive the control messages having size B using available sub-bands with the latency T. We formulate the requirement of the successful delivery probability of packets of size B bytes with the total time constraint T in the two phases.

1) Phase I: $\mathbb{P}\left(\Delta_T \sum_{t=t_1}^{T_1} R_n^{(\mathrm{c})}(t) \geq B\right)$. where Δ_T is the time duration of one time slot t. This is an optimization problem where we need to find the user association, channel assignment, and power allocation for all the cluster leaders $n \in N$ to maximize the payload delivery probability.

2) Phase II: $\mathbb{P}\left(\Delta_T \sum_{t=t_2}^{T_2} R_o^{(\mathrm{d})}(t) \geq B\right)$. For Phase II, we consider the scheme to be discussed in Section III, where each cluster leader transmits to its cluster members on different time-frequency resources. The members are close to their corresponding cluster leader and reliable communication can be achieved.

III. Resource Allocation using Deep Reinforcement Learning

A. Reinforcement Learning and Deep Q-Learning

Reinforcement Learning (RL) is an area of Machine Learning (ML) where an agent interacts with the environment and takes actions to maximize the cumulative reward. RL is based on trial-and-error, where the agent needs to learn which actions to take based on the feedback signal of reward. The job of the agent is to maximize the expected cumulative reward in the future in accordance with a long-term objective. The RL can be modeled as a Markov Decision Process (MDP) consisting of a set of states $\mathcal{S}$, a set of actions $\mathcal{A}$ and a reward function R. According to the Markov property, the current state captures all relevant information from history. As a result, the future state is independent of the past states given the present state. At each time step t, the agent observes the environment through the state $s_t \in \mathcal{S}$ and takes an action $a_t \in \mathcal{A}(s)$. The agent receives a reward r_{t+1} and moves to a new state s_{t+1}. In RL, the policy is defined as the distribution $\pi(a|s)$ that maps the states to the possible actions in that state.

In RL, the aim of the agent is to maximize the cumulative reward G_t it receives in the long run, given by:

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{k+t+1}$$

where $\gamma \in [0, 1]$ represents the discount factor which reduces the present value of the future rewards.

The action-value function $Q_\pi(s, a)$ is defined as the expected return by taking an action a in state s by following a policy π.

$$Q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

where the expectation is taken over all possible transitions following the distribution π. The goal of RL is to find the optimal policy π^* that maximizes the Q-function over all the policies.

Q-learning [17] is a model-free RL algorithm that learns the value of an action a in a state s. It has been proven that Q-learning can always converge to the optimum action-value function if the actions in a state are sampled sufficiently [18]. It iteratively updates the action-value function for each state-action pair (s, a) until they converge to the optimal action-value function $Q_*(s, a)$. If the Q-function is estimated accurately, the optimal policy π^* at a given state s would be to select the action a^* that yields the highest action-value. However, for situations with a large number of action and state spaces, saving a separate q-value each state-action pair (s, a) is not a feasible option due to practical reasons.

The Deep Q-Learning [17] uses a Deep Neural Network (DNN) as a function approximator to solve for the optimal policy. In Deep Q-Network (DQN), the parameters of the DNN learn to approximate the Q-function. The DNN takes the state $s \in \mathcal{S}$ as input and outputs the value for each action a. The state-action space is explored with a soft-policy such as ϵ-greedy [19]. where the agent takes random action at a given

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 12,2023 at 14:15:44 UTC from IEEE Xplore. Restrictions apply.

state s_t at time t with a probability of ϵ. Otherwise, greedy action $a^* = \arg\max_{a \in \mathcal{A}} Q^*(s_t, a)$ is selected. The tuple $< s_t, a_t, r_{t+1}, s_{t+1} >$ is stored in the replay memory at each time instance. At each time-step, a mini-batch is sampled from the replay memory to update the DNN parameters θ using the gradient descent algorithm.

B. Multi-Agent Resource Allocation Algorithm

In this section, we formulate the multi-agent RL problem where each cluster leader acts as an agent. The agents need to decide the joint user association, channel assignment, and power allocation. The agents interact with the environment and learn to adjust to allocate resources by trial and error. The agents aim to maximize a common reward which encourages the agents to collaborate to increase the reward.

The multi-agent RL problem consists of two phases that are training and testing. The training phase is centralized where each agent has access to the common reward to train the DQN. Each agent has a distinct DQN. The testing phase is distributed, where each agent uses its trained DQN to select the action.

1) State Space: The state space of the agent consists of the measurements from the last time step. Let $\mathcal{Z}_n(t)$ denote the state of agent n at time step t. The state space consists of the following measurements:

- The direct channels from all the APs $k \in K$ to the cluster leader n. This consists of both the large-scale and small-scale fading powers i.e., $\{L_{kn}^{(\mathrm{c})}, g_{kn}^{(\mathrm{c})}\}_{k \in K}$
- the aggregate interference and noise powers at the agent n i.e., $\{\sum_{i \in K} \sum_{j \neq n} \rho_{ij}^{(\mathrm{c})}[m] P_{ij}^{(\mathrm{c})} L_{ij}^{(\mathrm{c})} g_{ij}^{(\mathrm{c})}[m] + \sigma^2\}$
- The remaining payload and time limitation after the current time step.
- The training iteration number e and the probability of random action selection ϵ of agent n.

2) Action Space: Each agent n has an identical action space $\mathcal{A}$. The action space consists of the combination of AP selected, the occupied sub-band, and the power level of the agent. There are K distinct APs the agent can connect to. Meanwhile, there are M sub-bands and the power level is assumed to be divided into discrete levels in the range $[0, P_d]$ where P_d is the maximum allocated power.

3) Reward: To ensure collaboration among the agents, a common reward function is used for all agents. Our objective is to maximize the successful payload delivery probability within a time constraint of T_1. To achieve this objective, we define the reward as the sum rate of all cluster leaders until the payload is delivered. This reward at each cluster leader n at time-step t is mathematically formulated as:

$$U_n^{(\mathrm{c})}(t) = \begin{cases} R_n^{(\mathrm{c})}(t), & \text{if } B_n(t) \geq 0, \\ U, & \text{otherwise.} \end{cases} \tag{6}$$

U needs to be adjusted empirically and should be greater than the maximum rate achievable by the AP-to-leader link [7] and $B_n(t)$ is the remaining payload for cluster leader n at time-step t. The agents need to learn to select their AP, corresponding sub-band, and power level such that they create minimal interference to the communication of other cluster leaders. The reward function at an instant t is thus given by:

$$r_t = \sum_{n=1}^{N} U_n^{(\mathrm{c})}(t) \tag{7}$$

4) Training and Testing Algorithms: The training and testing algorithm consists of an episodic setting with a time limitation of $T = T_1 + T_2$ to complete the transmission to all of the robots. The training phase consists of the communication in Phase I only since we consider a fixed scheme for Phase II. The episode consists of multiple time-steps and at the start of each episode, the large-scale and small-scale fading gains are randomly generated. The small-scale fading gain is updated at each time step while the large-scale fading gain remains constant during the episode. The variation in small-scale fading changes the state space causing the agents to adjust their actions. The agents terminate their transmission once the payload has been delivered.

Training Phase: We use Deep Q-learning to train the agents. At each time-step t, the agents take an action based on ϵ-greedy policy and the tuple $< s_t, a_t, r_{t+1}, s_{t+1} >$ is stored in the replay memory. At the end of each episode, a mini-batch is sampled from the replay memory and stochastic gradient descent is used to update the parameters. The algorithm is outlined in Algorithm 1.

Algorithm 1 Training Algorithm

```
Initiate the environment
Initiate the parameters θ for DNN of all cluster leaders
for each episode do
    Update the device locations and large-scale fading
    for each time step t do
        Update small-scale fading
        for each cluster leader n do
            Observe the state s_t
        end for
        The leaders take action according to the ε-greedy policy and obtain a common reward r_t
        for each cluster leader n do
            Observe the next state s_{t+1}
            Store < s_t, a_t, r_t, s_{t+1} > in the replay memory
        end for
    end for
    for each cluster leader n do
        Sample a mini-batch X from the replay memory
        Train the deep Q-network for cluster leader n using X
    end for
end for
```

Testing Phase: The testing phase consists of both Phase I and Phase II. During Phase I, the agents observe the environment state $\mathcal{Z}_n(t)$ and use the trained DNN to output the optimal action. After T_1 seconds, a fixed scheme is used in Phase II for the communication between the cluster leader and members. In this scheme, each of the cluster leader communicates to its member in different frequency-time slots, i.e., on a separate sub-band and time-step combination, which eliminates the need for interference management. This strategy is possible because the distance between cluster leaders and

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 12,2023 at 14:15:44 UTC from IEEE Xplore. Restrictions apply.

TABLE I
SIMULATION PARAMETERS

Carrier frequency	3 GHz
Bandwidth of each sub-band	1 MHz
Network area	40×40 m^2
Number of clusters N	[2, 4, 6, 8]
Number of available sub-bands M	$N/2$
Number of members in each cluster O	4
Robot velocity	1 m/s
Transmit power of AP to cluster leader	[30, 25, 20, -100] dBm
Noise figure	5 dB
Noise PSD	-169 dBm/Hz
Latency T	1 ms
Latency of Phase I T_1	0.667 ms
Latency of Phase II T_2	0.333 ms
Robot payload B_o	[20, 100] bytes
Max distance between leader and members d	3 m

members is shorter, so they can reliably transmit using such a fixed scheme.

IV. ILLUSTRATIVE RESULTS

We consider the downlink communication where control messages need to be delivered from the APs to the mobile robots. The mobile robots are able to perform multiple operations within a factory and can have large mobility.

We consider a factory floor setting of 40×40m^2 consisting of 4 APs. Considering the south-east corner of the floor having co-ordinates of $(0, 0)$, the APs are located at co-ordinates of $(10, 10)$, $(10, 30)$, $(30, 10)$ and $(30, 30)$. There are N clusters in the factory, each is randomly located within the floor. The value of N ranges from 2 to 8 in our simulations. Each cluster consists of 1 leader and 4 members, and each member is located at a maximum distance of 3 m from the leader. We assume the robots move with a velocity of 1 m/s and the cluster moves together to perform a specific task. We assume the cluster can move up, down, left or right with an equal probability. The slow-scale and fast-scale fading is updated every 1 ms and 0.1667 ms respectively. For path-loss we consider the A1-Indoor scenario of WINNER II channel model [20]. According to the channel model, supposing a distance of d meters between the transmitter and receiver, the probability of having Line-of-Sight (LOS) component is given by:

$$p_{\text{LOS}} = \begin{cases} 1, & \text{for } d \leq 2.5 \\ 1 - 0.9\left(1 - (1.24 - 0.61\log_{10}(d))^3\right)^{\frac{1}{3}}, & \text{for } d > 2.5 \end{cases} \tag{8}$$

The path-loss in dB is then given by:

$$L_{\text{LOS}} = 18.7\log_{10}(d) + 46.8 + 20\log_{10}(0.6),$$
$$L_{\text{NLOS}} = 36.8\log_{10}(d) + 43.8 + 20\log_{10}(0.6).$$

We assume log-normal shadowing with standard deviation of 3 dB. The small-scale fading is modeled as Rayleigh fading. The power control was divided into 4 levels with -100 dBm signifying no transmission and P_d =30 dBm. The simulation parameters are highlighted in Table I

The DNN for the DQN consists of three hidden layers, with 500, 250, and 120 nodes [7]. The rectified linear unit (ReLU) is used as the activation function and RMSProp is used as the optimizer with a learning rate of 0.001. The training phase consists of 6000 episodes and there are 6 time-steps per episode. For URLLC applications, we consider a subframe of 1 ms which is further divided into 6 sub-slots [21]. The fast-scale fading is updated at the end of each sub-slot, due to which the agents re-adjust their transmission strategy to cater for the change in dynamics of the system. During the training, the value of ϵ is linearly reduced from 1 to 0.02 for the first 4200 episodes after which it is kept a constant at 0.02. The value of γ is set as 0.9. We observed that setting the value of γ as 1 reduced the performance. We set the value of U defined in (6) as 40. During the training, we set the robot payload size B_o as 100 bytes while during the testing phase we vary it in the range of 20-100 bytes. The testing phase consists of 1000 episodes. During Phase II, since $M = N/2$ and there were 2 time steps, the N clusters broadcasted on different time-frequency resources to their members. For comparison, we developed the exhaustive search and random allocation scheme. For both exhaustive search and random allocation, we assume the leader connected to the nearest AP. The exhaustive search would have been impractical without this assumption due to the large action space. The exhaustive search algorithm performs search over all sub-band and power levels at each time-step t while the random allocation algorithm randomly selects the sub-band and power level.

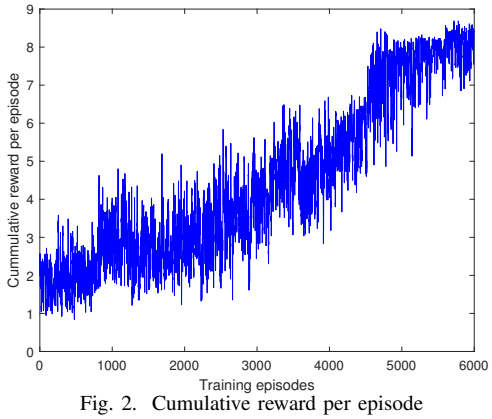

Fig. 2. Cumulative reward per episode

Fig. 2 shows the reward obtained per episode against the number of training episodes. The reward gradually increases during the training phase, becoming more stable at the end.

Fig. 3 shows the average successful payload delivery probability of the robots versus payload size B_o. The robot payload size B_o is varied in the range of 20-100 bytes. It can be seen that the exhaustive search yields a successful delivery probability of 1 up to 80 bytes. Our RL algorithm gets a successful payload delivery probability of 1 for 20-40 bytes. Further, it achieves a successful payload delivery probability of more than 0.99 for 80 bytes and less. The random allocation scheme in comparison achieves a successful payload delivery probability of 0.9 and less for 60 bytes and more. Increasing payload size reduces the performance since a larger payload requires longer duration to complete transmission.

Fig. 4 shows the average payload delivery probability of the robots versus the number of clusters N. The payload size B_o

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 12,2023 at 14:15:44 UTC from IEEE Xplore. Restrictions apply.

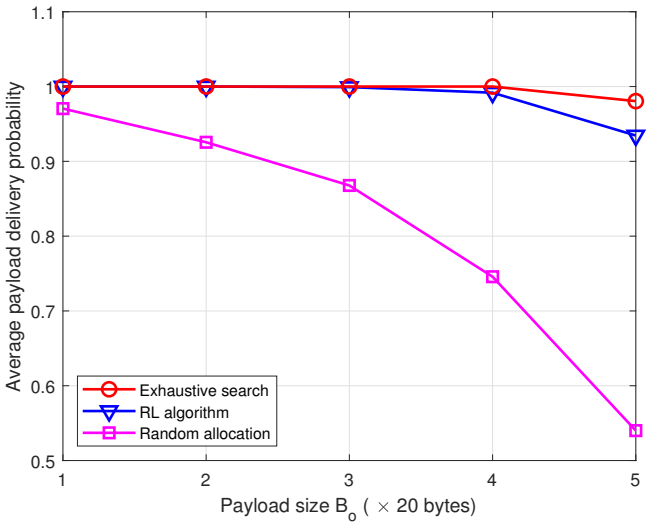

Fig. 3. Average robot payload delivery probability versus payload size

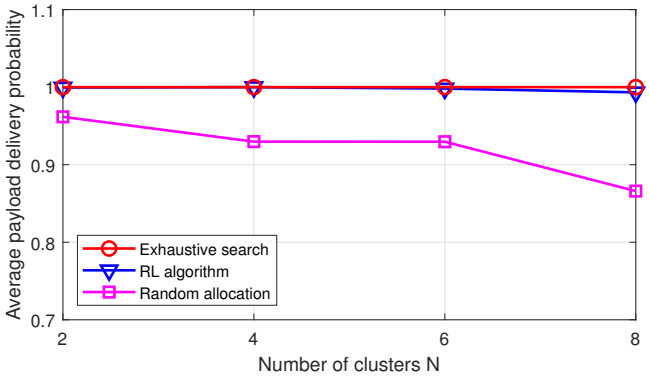

Fig. 4. Average robot payload delivery probability versus the number of clusters

was kept at 40 bytes for these experiments. It can be seen that generally, the successful delivery probability decreases with the increase in number of clusters. This is expected since the interference amongst the cluster leaders would be expected to increase with an increasing number of cluster leaders. The success probability decreases for the RL algorithm as well, however it obtains successful delivery probability in excess of 0.99 for all cases.

V. Conclusion

The paper developed a two-phase communication scheme for mobile robot scenario in the factory automation system. The robots work in close proximity and from multiple clusters for D2D communication. In Phase I, the AP associated with the cluster leader, combines the messages for the cluster and transmits them to the cluster leader. Meanwhile, in Phase II, the cluster leader broadcasts the message to all cluster members. For the communication in Phase I, we used MARL as a distributed algorithm for joint user association, channel assignment, and power control. The cluster leaders acted as the agents by interacting with the environment to find the optimal resource allocation. The proposed algorithm was decentralized, and the cluster leaders needed only the local CSI to make decisions. Its performance is close to that of centralized exhaustive search algorithm.

Acknowledgment

This work was supported in part by Huawei Technologies Canada and in part by the Natural Sciences and Engineering Research Council of Canada.

References

[1] "5G for Connected Industries and Automation (White Paper-Second Edition)," 5G-ACIA, Feb. 2019.
[2] T. Hößler, P. Schulz, E. A. Jorswieck, M. Simsek, and G. P. Fettweis, "Stable Matching for Wireless URLLC in Multi-Cellular, Multi-User Systems," *IEEE Trans. Commun.*, vol. 68, no. 8, pp. 5228–5241, Aug. 2020.
[3] M. Bennis, M. Debbah, and H. V. Poor, "Ultrareliable and low-latency wireless communication: Tail, risk, and scale," *Proceedings of the IEEE*, vol. 106, no. 10, pp. 1834–1853, Oct. 2018.
[4] H. Ren, C. Pan, Y. Deng, M. Elkashlan, and A. Nallanathan, "Joint pilot and payload power allocation for massive-MIMO-enabled URLLC IIoT networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 816–830, May 2020.
[5] "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Study on Communication for Automation in Vertical Domains (release 16)," 3GPP, Tech. Rep., Jul. 2020.
[6] Y. S. Nasir and D. Guo, "Deep Reinforcement Learning for Joint Spectrum and Power Allocation in Cellular Networks," *arXiv preprint arXiv:2012.10682*, 2020.
[7] L. Liang, H. Ye, and G. Y. Li, "Spectrum Sharing in Vehicular Networks Based on Multi-Agent Reinforcement Learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2282–2292, Oct. 2019.
[8] L. Liu and W. Yu, "A D2D-based Protocol for Ultra-Reliable Wireless Communications for Industrial Automation," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5045–5058, Aug. 2018.
[9] H. Ren, C. Pan, Y. Deng, M. Elkashlan, and A. Nallanathan, "Resource allocation for URLLC in 5G mission-critical IoT networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2019, pp. 1–6.
[10] N. Jayaweera, D. Marasinghe, N. Rajatheva, and M. Latva-Aho, "Factory Automation: Resource Allocation of an Elevated LiDAR System with URLLC Requirements," in *IEEE 6G Wireless Summit*, 2020, pp. 1–5.
[11] R. Balakrishnan, K. Sankhe, V. S. Somayazulu, R. Vannithamby, and J. Sydir, "Deep Reinforcement Learning Based Traffic-and Channel-Aware OFDMA Resource Allocation," in *IEEE Global Commun. Conf. (GLOBECOM)*, 2019, pp. 1–6.
[12] B. Khodapanah, T. Hößler, B. Yuncu, A. N. Barreto, M. Simsek, and G. Fettweis, "Coexistence Management for URLLC in Campus Networks via Deep Reinforcement Learning," in *IEEE Wireless Commun. Networking Conf. (WCNC)*, 2020, pp. 1–6.
[13] Y. Han, L. Liu, L. Duan, and R. Zhang, "Towards Reliable UAV Swarm Communication in D2D-Enhanced Cellular Network," *IEEE Trans. Wireless Commun.*, Nov. 2020.
[14] S. R. Khosravirad, H. Viswanathan, and W. Yub, "Exploiting Diversity for Ultra-Reliable and Low-Latency Wireless Control," *IEEE Trans. Wireless Commun.*, Sep. 2020.
[15] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with Deep Reinforcement Learning," 2013, cite arxiv:1312.5602 Comment: NIPS Deep Learning Workshop 2013. [Online]. Available: http://arxiv.org/abs/1312.5602
[16] R. Bird and J. Gibbons, *Algorithm Design with Haskell*. Cambridge University Press, 2020.
[17] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, Cambridge United Kingdom, May 1989.
[18] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.
[19] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
[20] P. Kyosti, "WINNER II channel models," *IST, Tech. Rep. IST-4-027756 WINNER II D1. 1.2 V1. 2*, 2007.
[21] T. Fehrenbach, R. Datta, B. Göktepe, T. Wirth, and C. Hellge, "URLLC services in 5G low latency enhancements for LTE," in *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, 2018, pp. 1–6.

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 12,2023 at 14:15:44 UTC from IEEE Xplore. Restrictions apply.