Scalable and Efficient URLLC Resource Allocation Based on Offline Graph Reinforcement Learning

Ren Jiaqi, Wenpeng Jing, Xiangming Wen, Zhaoming Lu, Shuyue Zhao
Beijing Key Laboratory of Network System Architecture and Convergence
Beijing Laboratory of Advanced Information Networks
Beijing University of Posts and Telecommunications, Beijing, China
{Renjiaqi, jingwenpeng, xiangmw, lzy0372, syzhao}@bupt.edu.cn

***Abstract*—Deep reinforcement learning (RL) has emerged as a transformative technology for addressing intricate radio resource allocation problems in both 5G and 6G networks. Nevertheless, during the training phases of RL-based resource allocation algorithms, online RL methods introduce potential risks as they necessitate continuous interaction with the environment. Motivated by the fact that the historical data of the resource allocation process in the base station (BS) reveals the operation rules of the mobile network dynamics, this paper proposes a novel data-driven framework for resource allocation optimization, termed Offline Graph Reinforcement Learning (OGRL). Different from existing intelligent resource allocation schemes, which acquire policies by interacting either with wireless network simulators or real-world networks, the proposed OGRL framework exclusively leverages historical data from BS to train a high-performance resource allocation policy. Furthermore, by harnessing the capabilities of the graph neural networks (GNNs) at processing intricate data structures, OGRL can adeptly manage dynamic network environments characterized by continuously changing active users. Extensive experimental results substantiate that, following training with historical data, the proposed OGRL attains Quality of Service (QoS) and packet loss rates comparable to those of the best online algorithms. Furthermore, OGRL exhibits excellent scalability and generalization capabilities.**

***Index Terms*—Resource allocation, Offline reinforcement learning, Graph neural network**

I. Introduction

The upcoming wireless communication systems will be able to handle important connections that have strict requirements for both reliability and latency. International Mobile Telecommunications for 2030 and beyond (IMT-2030) expands three scenarios on the basis of the three typical scenarios of IMT-2020 (5G), namely immersive communication, massive communication, and hyper-reliable and low-latency communication [1]. As a further expansion of Ultra Reliable Low Latency Communication (URLLC) , hyper-reliable and low-latency communication can better support services that require high levels of reliability and low latency, such as vehicle-to-everything communication, factory automation [2], vehicle-to-vehicle communication [3], and augmented reality [4]. Among the techniques that can be utilized to satisfy the strict quality of service (QoS) of URLLC , radio resource allocation (i.e., BSs allocate radio resources based on user requirements and the wireless environment) is undoubtedly one of the key techniques. This is because the wireless resources at the BS are limited, and the rationality of resource allocation directly affects the overall performance of the wireless network. However, most of the traditional scheduling algorithms, e.g., proportional fair (PF) [5], and earliest-dead-line-first (EDF) [6], can hardly satisfy the QoS of URLLC services. This means that the radio resource allocation algorithm needs to be redefined to effectively support these new cases.

More recently, reinforcement learning (RL) based algorithms are developed to deal with the resource allocation problem of URLLC services. In [7], the authors propose a deep RL framework to solve the problem of semantic-aware resource allocation. In [8], the authors propose a deep RL based resource allocation framework to mitigate the interference and improve the constellation capacity in integrated satellite and terrestrial networks. In [9], the authors consider the resource allocation algorithm design for downlink multiple-input single-output orthogonal frequency division multiple access URLLC systems. Compared with traditional scheduling strategies, the RL-based scheduling policies are capable of handling resource allocation tasks in more complex 5G network environments. However, all existing resource allocation algorithms in [7-9] are designed based on online RL methods and commonly relay on fully connected neural networks to approximate the objective functions. This approach leads to the following issues:

- ***The Sim-to-Real Gap:*** While wireless simulators are useful tools to accelerate the training of RL-based resource allocation algorithms, they often are provided only with an approximated model of the wireless environment, thus resulting in what is called the sim-to-real gap: a mismatch of simulated and real resource allocation performances caused by the inaccurate representation of the real environment in simulation. This means that resource allocation algorithms that are fully pre-trained in a simulated environment still need to be fine-tuned online [10]. The inappropriate decisions during fine-tuning may try bad scheduling actions that would incur quality of service degradation, congestion, or even system instability in the wireless networks.
- ***Low Scalability and Generalization Performance:*** Most of the existing RL-based resource allocation algorithms proposed for B5G URLLC make use of a neural network that has a fixed input/output structure. If the number of active users (i.e., the users with packets awaiting transmission) in the real environment is much different from that in the training process, the resource allocation algorithms need to be retrained. This means RL-based models that are

only suitable for specific scenarios, with poor scalability and generalization performance in the large-scale resource allocation problem.

Motivated by the issues above, this paper investigates the resource allocation optimization problem, which aims at providing a sufficient QoS guarantee for the URLLC services in the B5G and 6G communication systems. Specifically, the resource allocation problem is formulated, where the optimization objective is to maximize the total number of successfully transmitted and decoded URLLC data packets for users over a long time scale. The resource allocation problem is proved to be a Markov decision process (MDP), and a novel RL-based optimization framework that combines the offline RL and graph neural network (GNN) is proposed to solve the problem. The main contributions of this paper are as follows:

- We propose the OGRL, which is a novel resource allocation optimization framework for the resource allocation of URLLC. Specifically, different from the existing RL-based framework that needs to interact with the practical wireless network, or network simulator to improve the policy, OGRL only utilizes the historical data of the resource allocation process of the practical wireless networks to train the algorithm in an offline manner. Without interacting with the wireless network environment, this framework can effectively ensure the safety of policy learning.
- We introduce GNN into OGRL to deal with the dynamic wireless network environment where the number of active users is constantly changing. By GNN modules' capability of handling variable-size inputs, OGRL can process graph-structured data with varying numbers of active users.
- We design a resource allocation algorithm for URLLC services based on OGRL. Extensive simulations show the superiority and effectiveness of the proposed algorithm, which can perform better than other offline-based RL algorithms (e.g., Conservative Q-Learning (CQL) [11], Batch-Constrained deep Q-learning (BCQ) [12]) in terms of QoS and can achieve the same performance as the state of the art online Soft Actor-Critic (SAC) [13] algorithm.

II. System Model and Problem Formulation

A. System Model

In this work, we focus on the radio resource allocation of the downlink transmission in the OFDMA wireless network for URLLC services. We consider a single BS that serves a set $\mathcal{N}$ of N active users and the wireless network has a set $\mathcal{K}$ of K available RBs. A RB is the minimum resource unit allocated to users in the time frequency domain, and the BS's scheduler determines the number of RBs allocated to the users according to the CSI and QSI in each transmission time interval (TTI) Δ^t. $x_n(t), n \in \mathcal{N}$, is used to indicate whether user n is allocated resources in t-th slot. If the n-th user is not scheduled in the t-th slot, $x_n(t)$ = 0. Otherwise, $x_n(t)$ = 1. The amount of data that transmitted to user n in t-th slot can be represented by $A_n(t)$. It is assumed that the packet arrival processes of users follow Bernoulli processes. In each time slot, a packet arrives with probability p_n, and no packet arrives with probability 1 - p_n. Packets for the different users are waiting to be scheduled in the corresponding queue, and each queue obeies the rule of the first-in-first-out (FIFO).

The radio resource allocation algorithm of BS's scheduler is designed to provide URLLC services with millisecond level end-to-end delay and nearly 100% reliability. The delay D_n of user n's packet comprises the queuing delay and transmission delay. The scheduler determines whether to schedule users and the number of RBs allocated to users according to the channel state information and the head-of-line (HoL) delays(i.e., queue delay of the first packets). According to the specification of 3GPP, the delay experienced by URLLC service's packets should higher than $D_{\min}$ and lower than $D_{\max}$ [14]. The reliability is defined according to the packet loss probability. If $D_n(t) \notin [D_{\min}, D_{\max}]$ or a specific packet can not be decoded correctly, the packet is lost [10]. The decoding error probability $\epsilon_n(t)$ of user n's packet can be approximated by [15]

$$\epsilon_n(t) \approx f_Q\left(\frac{-A_n ln2 + \Delta^t W K_n(t) ln[1+\phi_n(t)]}{\sqrt{\Delta^t W K_n(t) C_n(t)}}\right), \quad (1)$$

where W is the bandwidth of each RB, $K_n(t)$ is the number of RB allocated to the user n in the t-th slot, $\phi_n(t)$ is the downlink signal-to-noise (SNRs) of user n in t-th slot, and $C_n(t)$ is the channel dispersion.

To avoid the long transmission delay, retransmission can not be used to guarantee the reliability of URLLC. The decoding error probability of user n should not exceed the following threshold, i.e.,

$$\epsilon_n(t) \leq \epsilon_{\max}. \quad (2)$$

It can be seen that the decoding error probability in (1) decreases with the increase in the number of RBs. If each user is scheduled to transmit at most one packet each TTI, the corresponding minimum number of RB $K_n^*(t)$ that should be allocated to the n-th user can be obtained by binary research [16].

Due to the wireless resource is limited, the number of RBs allocated to the user n can be represented by [10]

$$K_n(t) = \begin{cases} x_n(t)K_n^*(t), \text{if} \sum_{n=1}^{N} x_n(t)K_n^*(t) \leq K \\ [\frac{x_n(t)K_n^*(t)}{\sum_{n=1}^{N} x_n(t)K_n^*(t)} N], \text{if} \sum_{n=1}^{N} x_n(t)K_n^*(t) > K. \end{cases} \quad (3)$$

When radio resources are sufficient, allocation is demand-driven, while resources are constrained, allocation is proportion-based.

Furthermore, the total number of packets successfully received by users in t-th slot is defined as

$$L(t) = \sum_{n=1}^{N} [x_n(t) \cdot \mathbf{1}_n^{\epsilon_n(t) \leq \epsilon_{\max}} \cdot \mathbf{1}_{D_n(t) \in [D_{\min}, D_{\max}]}]. \quad (4)$$

Specifically, if the transmission delay $D_n(t)$ of the packet for user n is within $[D_{\min}, D_{\max}]$, $1_{D_n(t) \in [D_{\min}, D_{\max}]} = 1$, else $1_{D_n(t) \in [D_{\min}, D_{\max}]} = 0$. If the packet for user n is decoded successfully, $1_n^{\epsilon_n(t) \leq \epsilon_{\max}} = 1$, else $1_n^{\epsilon_n(t) \leq \epsilon_{\max}} = 0$.

B. Problem Formulation

We consider a discrete time period $\mathbb{T} = \{1, 2, 3, ..., T\}$. The goal of this paper is to maximize the total number of successfully transmitted and decoded URLLC data packets for users over a long time scale by optimizing the resource allocation in each time slot. Hence, the overall problem can be formulated as

$$\mathbf{P1}: \max_{\{x_n(t)\}} \sum_{t\in\mathbb{T}} L(t), \tag{5}$$

$$\text{s.t.} \quad x_n(t) \in \{0, 1\}, \ \forall n \in \mathcal{N}, \tag{5a}$$

Note that problem $\mathbf{P1}$ is a sequential decision problem with 0-1 integer decision variables.

III. Proposed OGRL Architecture

Radio resource allocation is challenging given the dynamic situation in real-time wireless systems. A large number of status combinations make it difficult to select resource allocation algorithms as well as corresponding parameters. In this case, deep RL can be an excellent tool to approximate the interaction between resource allocation decisions and data transmission performances. Figure 1 illustrates the implementation of the proposed RL-based radio resource allocation framework. A detailed introduction to the implementation of the framework will be provided below.

A. MDP Formulation

The wireless channel fading in our model is assumed to be Markovian. Both time slot and HoL delays are discrete variables, and the HoL delay in the next slot is only related to the current state. So the optimization problem $\mathbf{P1}$ is a MDP problem, which can be solved through RL [17]. Specifically, a MDP can be represented by a tuple $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$, where $\mathcal{S}$ denotes the set of states, $\mathcal{A}$ denotes the set of actions, $\mathcal{P}$ denotes the transition probability from state $\mathbf{s} \in \mathcal{S}$ to state $\mathbf{s}' \in \mathcal{S}$, $\mathcal{R}$ denotes the reward received by agent, γ represents the discount factor. At each time slot, the resource scheduling policy executes an action $\mathbf{a}(t)$ that determines whether the user would be scheduled or not. The wireless network will transition to the next state $\mathbf{s}(t+1)$ and return a reward $r(t)$ which indicates the effect of $\mathbf{a}(t)$. The state, action and reward of resource scheduling are set in details as follows.

Action: The action $\mathbf{a}(t)$ is defined as $[x_1(t), ..., x_N(t)]$ [10], i.e., whether each active user in slot t is scheduled.

State: The packet delay and channel state are the key factors that impact the resource allocation for URLLC service. Thus the state of the system in slot t is given by $\mathbf{s}(t) = [\frac{D_1(t)}{D_{\max}}, ..., \frac{D_N(t)}{D_{\max}}, \frac{K_1^*(t)}{K}, ..., \frac{K_N^*(t)}{K}]$ [10], where $D_n(t)$ represents the HoL delays of user n , $K_n^*(t)$ represents the number of RBs allocated to the user n during the t-th slot.

Reward: The total reward of the system in t-th slot is defined as the number of packets that successfully decoded by all users, i.e., $r(t) = L(t)$.

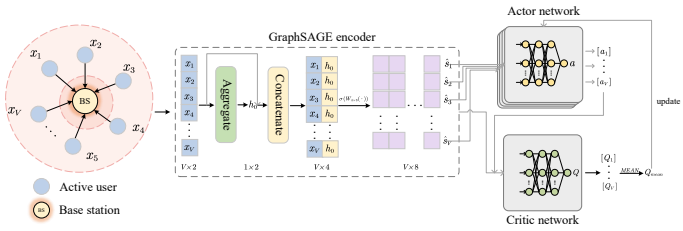

Fig. 1. The overall architecture of the proposed framework

B. GNN and Offline-RL Based Scheduling Framework

In the real wireless environment, the number of active users in the coverage range of the BS is constantly changing. Traditional RL-based resource allocation algorithms, often relying on fully connected neural networks to approximate the relationship between input states and output actions, necessitate retraining when confronted with untrained large-scale active users. Considering that communication networks can be modeled as graph structures, and GNNs can process graph information without restrictions on the size and shape of the graph, we introduce GNN to handle the scenario with the changing number of active users and improve the scalability and generalization of the resource allocation policy.

GNN network. The Graph SAmple and aggreGatE (GraphSAGE) algorithm is one of the most well-known GNN algorithms. This approach is independent of the graph's structure, such as the node degree distribution or batch size. The GraphSAGE algorithm operates on a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents a set of node, and $\mathcal{E}$ represents a set of edges between the nodes. The node features are denoted as $\{\mathbf{x}_v, \ \forall v \in \mathcal{V}\}$. The number of nodes is defined as $V = |\mathcal{V}|$. In In this paper, the relations among the BS and active users at each time slot can modeled by a certain graph. Specifically, the V-th node represents the base station, and nodes numbered from 1 to $V-1$ correspond to N active users. The initial features of v-th active user is set to $\mathbf{x}_v(t) = [\frac{D_v(t)}{D_{\max}}, \frac{K_v^*(t)}{K}]$, $v \in [1, V-1]$, and the node feature of the BS is set to [0, 0].

To make the resource allocation decision, the BS's scheduler should aggregate all the active users' states, which can be achieved by GraphSAGE. Given the graph consisting of BS and users, GraphSAGE uses an aggregation function called MEAN to aggregate the information from node neighbors and the corresponding weight matrix W to map the original features of nodes to a new representation space. Specifically, GraphSAGE can obtain an embedding vector that can represent the state of each user by following steps:

(i) BS aggregates the initial features of users into the neighborhood vector $\mathbf{h}^0$:

$$\mathbf{h}^0 = \text{MEAN}(\mathbf{x}_v, \ \forall v \in [1, V-1]), \tag{6}$$

the aggregate function MEAN means we just take the element-wise mean of the vectors in $\{\mathbf{x}_v, v = 1, ..., V-1\}$.

(ii) The users' original feature $\mathbf{x}_v$ is concatenated with vector $\mathbf{h}^0$, which is later fed into a fully connected layer with a nonlinear activation function σ, and the embedding vector of each user becomes:

$$\mathbf{h}_v = \sigma(W \cdot \mathrm{CONCAT}(\mathbf{x}_v, \mathbf{h}^0)). \quad (7)$$

The CONCAT(·) function concatenates inputs along a specified dimension. The final representation of active users' state $\hat{\mathbf{s}}_v = \mathbf{h}_v, \forall v \in N_{neighbor}(V)$.

Subsequently, the original state $\mathbf{s}(t)$ is replaced with $\hat{\mathbf{s}}(t)$ (i.e., $\{\hat{\mathbf{s}}_v, v \in [1, V-1]\}$) as the input state for resource allocation algorithm training. In each iteration, the state of user 1 to user V are input into the actor network in turn, and the corresponding scheduling actions are output. Then the Q_v corresponding to $(\hat{\mathbf{s}}_v, \mathbf{a}_v)$ is calculated sequentially through the critic network, and the final $Q((\hat{\mathbf{s}}, \mathbf{a}))$ is calculated by averaging $\{Q_1, ..., Q_{V-1}\}$.

Conservative Q Learning (CQL). The goal of reinforcement learning is to learn an optimal policy through interaction with the environment in order to obtain the maximum long-term reward in future decisions. Through $\hat{\mathbf{s}}(t)$ and $\mathbf{a}(t)$, the long-term reward is estimated by an state-action value function: $Q(\hat{\mathbf{s}}(t), \mathbf{a}(t)) = \mathbb{E}[\sum_{i=0}^{\infty} \gamma^i r(t+i)]$, where γ represents the discount parameter. CQL is a data-driven deep RL algorithm based on SAC that achieves the best-in-class results in offline RL problems.

The online SAC algorithm alternates between policy evaluation and policy improvement [11]. Specifically, the policy is evaluated by

$$\hat{Q}^{t+1} \leftarrow r(\hat{\mathbf{s}}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{a}^{t+1} \sim \pi^t(\mathbf{a}^{t+1}|\mathbf{s}^{t+1}])}[\hat{Q}^t(\hat{\mathbf{s}}^{t+1}, \mathbf{a}^{t+1})] \quad (8)$$

[11]. The policy π can be improved by

$$\hat{\pi}^{t+1} \leftarrow \arg\max_{\pi} \mathbb{E}_{\mathbf{a} \sim \pi^t(\mathbf{a}|\hat{\mathbf{s}})}[\hat{Q}^{t+1}(\hat{\mathbf{s}}, \mathbf{a}) - \tau log(\pi^t(\mathbf{a}|\hat{\mathbf{s}}))]. \quad (9)$$

The π^t is the learned policy, $\hat{\pi}^{t+1}$ is the learned policy in the $t+1$-th iteration, $\hat{Q}^{t+1}$ is the policy value in the $t+1$-th iteration, τ is the temperature parameter in SAC.

However, as introduced in Section I, this kind of online RL algorithm has the problems of the sim-to-real gap and low scalability and generalization performance in the resource allocation scenario. In contrast to the online RL, offline RL-based algorithms can effectively avoid these problem by learning from the real historical data of BS's operations. When the resource allocation policy conducts offline training, the datasets $\mathcal{D}$ collected by the behavior policy (i.e., the policy used to generate the dataset $\mathcal{D}$) are usually independent of the policy to be learned. Since π is trained to maximize Q-value, it may be biased towards out-of-distribution (OOD) actions (i.e., the distribution of data in the batch is different from the distribution under the current policy) with erroneously high Q-values. In online RL, this error can be corrected by trying an action in the environment and observing its actual value. However, offline RL can not correct the wrong actions in this way, thus the algorithm will learn the wrong policy due to the high Q value.

In order to deal with this problem, the CQL algorithm is adopted in this paper. The key idea of CQL is to bind the difference between the Q values evaluated at in-distribution actions (i.e., the distribution of data in the batch is the same as the distribution under the current policy) and OOD actions so that the policy update step will not exploit OOD actions. Specifically, CQL makes that the expected value of a policy under the learned Q-function lower-bounds its true value. A lower bound on the Q-value prevents the over-estimation that is common in offline RL settings due to OOD actions [11].

Algorithm 1 Offline Training of the OGRL

Initialize θ, ϕ, α, τ, dataset $\mathcal{D}$
for step t in {1,...,N} **do**
  for $v \in [1, V-1]$ **do**
    $\mathbf{h}^0 = \mathrm{MEAN}(\mathbf{x}_v, \ \forall v \in [1, V-1]$
    $\mathbf{h}_v = \sigma(W \cdot \mathrm{CONCAT}(\mathbf{x}_v, \mathbf{h}^0))$
  end for
  $\hat{\mathbf{s}} = \{\mathbf{h}_v, v \in [1, V-1]\}$
  for $v \in [1, V-1]$ **do**
    Calculate Q_v corresponding to each $(\hat{\mathbf{s}}_v, \mathbf{a})$
  end for
  $Q(\hat{\mathbf{s}}, \mathbf{a}) = MEAN\{Q_v, v \in [1, V-1]\}$
  Update critic θ with gradient descent via Eq. (10)
  Improve policy π_k through:
    $\phi_t = \phi_{t-1} + \eta_\pi \mathbb{E}_{\hat{\mathbf{s}} \sim \mathcal{D}, \mathbf{a} \sim \pi_k(\mathbf{a}|\hat{\mathbf{s}})}[\hat{Q}^{k+1}(\hat{\mathbf{s}}, \mathbf{a}) - \tau log \pi(\mathbf{a}|\hat{\mathbf{s}})]$
end for

The general form of the optimization problem in CQL can be represented by

$$\begin{aligned} &\min_Q \max_\pi \alpha(\mathbb{E}_{\hat{\mathbf{s}} \sim \mathcal{D}, \mathbf{a} \sim \pi}[Q(\hat{\mathbf{s}}, \mathbf{a})] \\ &- \mathbb{E}_{\hat{\mathbf{s}} \sim \mathcal{D}, \mathbf{a} \sim \hat{\pi}(\mathbf{a}|\hat{\mathbf{s}})}[Q(\hat{\mathbf{s}}, \mathbf{a})]) \\ &+ \frac{1}{2} \mathbb{E}_{\hat{\mathbf{s}}(t), \mathbf{a}, \hat{\mathbf{s}}' \sim \mathcal{D}}[(Q(\hat{\mathbf{s}}, \mathbf{a}) - \hat{B}^{\pi_k} \hat{Q}^k(\hat{\mathbf{s}}, \mathbf{a}))^2] + R(\mu), \end{aligned} \quad (10)$$

where π represents the target policy, α is the regularization scaling parameter in CQL [11], $\alpha(\mathbb{E}_{\hat{\mathbf{s}} \sim \mathcal{D}, \mathbf{a} \sim \pi}[Q(\hat{\mathbf{s}}, \mathbf{a})] - \mathbb{E}_{\hat{\mathbf{s}} \sim \mathcal{D}, \mathbf{a} \sim \hat{\pi}(\mathbf{a}|\hat{\mathbf{s}})}[Q(\hat{\mathbf{s}}, \mathbf{a})])$ denotes the regularization term used to push down the big Q-values, $\hat{\pi}$ represents the behavior policy, π_k represents the policy generated during iteration and $\hat{B}^{\pi_k}$ represents the empirical Bellman operator. The first and second items are used to constrain Q-function, where the first item is the penalty of the Q-function to obtain the lower bound of the Q-value, and the second item is used to maximize the Q-value and attain a tighter lower bound for V_π. The third item is the standard TD error which is the optimization goal for Q learning. To better estimate the Q-value, CQL adds the entropy of policy π as the fourth item.

The main parameters of OGRL and related settings are as follows: a) the learning rate of policy function η_π is set to 3×10^{-4}, b) the learning rate for Q functions η_c is set to 3×10^{-4}, c) the learning rate η_τ for temperature parameter of SAC is set to 3×10^{-4}, d) the initial α value α_{init} is set to 1, e) the initial τ value τ_{init} is set to 1, f) θ: parameters of Q-function, g) ϕ : parameters of target policy π, h) the dimension of the weight matrix W of GraphSAGE model is set to 4×8, i) each fully connected layer contains 256 neurons, j) all activation functions use RELU function. The training procedure of OGRL is shown in Algorithm 1.

IV. Simulation and Performance Evaluation

A. Simulation Settings

A 5G wireless network simulator is implemented. The environment is set that 20 users are distributed randomly in the coverage of the single cell, and the radius of the cell is 100

meters. The transmit power spectrum density of the BS $P_{\max}$ is 20 dBm/Hz, the noise power spectrum density N_0=-90 dBm/HZ. The path loss model is 45+30log(l) dB, in which l is the distance between the user and the BS. The value of TTI is 125 μs [10], which corresponding to the minimum value of the slot. The $D_{\min}, D_{\max}$ is set as 5ms and 6 ms, respectively. The URLLC packets for each user arrive at the BS following the Bernoulli process. The bandwidth of a RB is 180kHz, and the size of each packet L is 32 bytes. The packet arrival probability p varies from 0.1 to 0.5, the probability that a user becomes active in each time slot increases with the packet arrival rate.

B. Baseline Schemes

We compare the OGRL with the following baseline algorithms:

K-DDPG Algorithm: K-DDPG is proposed in [10] which is designed based on the online DDPG algorithm. This algorithm has demonstrated superior performance in handling resource allocation problems for URLLC compared to traditional resource allocation algorithms such as round-robin, MT, and EDF [10].

G-SAC Algorithm: The G-SAC is a resource allocation algorithm based on SAC [13]. The 'G' in G-SAC indicates that similar to OGRL, GraphSAGE is employed to process the original features, aimed at enhancing the scalability and generalization capabilities of the algorithm. At present, the G-SAC algorithm consistently outperforms all other online RL-based resource allocation policies.

K-BCQ Algorithm: K-BCQ is designed based on the offline RL algorithm BCQ. 'K' indicates that K-BCQ, like K-DDPG, is designed with the help of expert knowledge of scheduler design. The core idea of K-BCQ is to avoid distribution shift problem by introducing generative models that expand the experience pool and make more efficient use of offline dataset.

M-CQL Algorithm: The M-CQL is a resource allocation algorithm based on CQL. M-CQL does not add GraphSage and still uses the same fully connected neural network as K-DDPG. The 'M' means that M-CQL uses the multi-head neural network in the critic network that is also adopted by the K-DDPG so that it has multiple outputs to ensure each user's QoS.

C. Dataset Collection

We run these three schemes, i.e., the G-SAC scheme, the K-DDPG scheme, and the Random scheme on the 5G simulator to simulate the resource allocation process of the 5G BS's scheduler and collect the data to build the historical dataset. G-SAC and K-DDPG employ the optimal parameter settings obtained through numerous experiments. Implementing the Random scheme, the BS's action selection over the entire time horizon obeys a uniform probability distribution. Each strip of data in the dataset consists of a tuple $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} >$. At last, 5 datasets are generated with different behavior policy and dataset sizes, and the settings for the dataset are shown in Table I.

D. Performance Analysis

1) Convergence Validation: We assess the convergence of OGRL using datasets with different qualities and sizes. Dataset $D1$ is used to represent the expert dataset (i.e., the dataset was collected by an optimal online algorithm and covers most of the action-state space), D_4 is used to represent the expert dataset with a small size, D_3 is used to represent sub-optimal dataset (i.e., the dataset was collected by an sub-optimal online algorithm and covers a part of the action-state space), and D_5 is used to represent random dataset respectively. We set the maximum number of users and packet arrival rate in the test environment to 15 and 0.3. Fig.2 illustrates the variations in the total reward during the training of our proposed OGRL framework. We observed that in experiments conducted on four datasets of different qualities and sizes, the OGRL can converge to a stable state, within 400 iterations. This means that OGRL has a good convergence. Meanwhile, the algorithm trained on the expert dataset showed the fastest convergence and best results. This can be interpreted as the expert dataset providing a better quality training sample, enabling the algorithm to learn more quickly. The algorithm trained on suboptimal datasets shows relatively slow convergence rates and lower final performance than expert datasets. This validates the critical impact of dataset quality on offline algorithm training.

TABLE I
DATASET COMPOSITION

Dataset	Behavior policy	Collection Environment		Size
		Number of users	Packet arrival rate	
D_1	G-SAC	15	0.3	$5 \cdot 10^4$
D_2	G-SAC	15	0.3	$2 \cdot 10^5$
D_3	K-DDPG	15	0.1	$5 \cdot 10^4$
D_4	G-SAC	15	0.1	$5 \cdot 10^3$
D_5	Random	15	0.1	$2 \cdot 10^5$

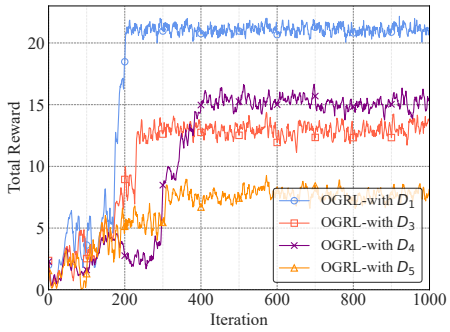

Fig. 2. Illustration of convergence speed of the proposed OGRL framework

2) Scalability and Generalization: We evaluate the scalability and generalization of the proposed OGRL framework and other baseline algorithms. We trained G-SAC and K-DDPG on the simulator in an online way until convergence with 15 total users and a 0.1 packet arrival rate. OGRL and M-CQL are trained in an offline way based on expert dataset D_2. Subsequently, the performance of different algorithms under two distinct scenarios is evaluated in the 5G simulator: 1) the total reward value that

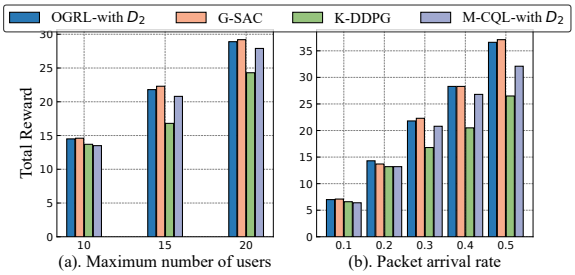

Fig. 3. Scalability and generalization.

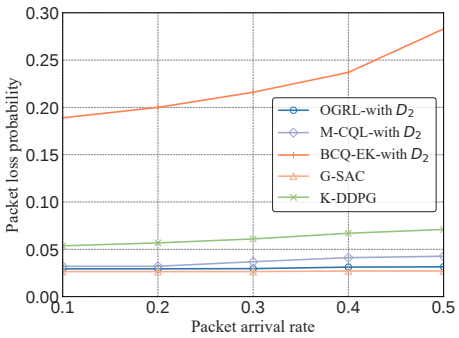

Fig. 4. Packet loss probability: N = 15

can be achieved under the same packet arrival rate with different user numbers as shown in Fig.3 (a), and 2) the total reward value that can be achieved under different packet arrival rates while maintaining the same total number of users as shown in Fig.3 (b). We can see that when the number of users is relatively small and the packet arrival rate is low, the performance of the four algorithms is relatively similar. However, as the number of users increases and the packet arrival rate rises, K-DDPG and M-CQL show slower performance growth, while OGRL can always maintain similar performance with the best online algorithm G-SAC. This is attributed to the GraphSage component in OGRL and G-SAC. In particular, although OGRL training is based on the dataset of 15 users with a packet arrival rate of 0.3, OGRL can still achieve excellent scalability and generalization when the number of users is larger than 15 and the packet arrival rate is larger than 0.3. The above results indicate that OGRL can generalize well to scenarios with total user numbers and packet arrival rates not encountered during the training process.

3) Performance Comparison With Baselines: Fig.4 shows the performance of the proposed OGRL framework and other baseline algorithms under different packet arrival rates in terms of mean packet loss probability. Both OGRL, M-CQL, and K-BCQ are trained by expert dataset D_2. G-SAC and K-DDPG are trained until convergence with 15 total users and a 0.3 packet arrival rate. The packet loss probabilities are evaluated over 1000 episodes. It can be observed that although OGRL is trained in an offline manner, it can achieve almost the same performance as the optimal online algorithm G-SAC. On the contrary, M-CQL and K-DDPG have worse performance. K-BCQ is also trained by expert dataset but performs poorly in QoS requirements assurance, which can be noted in Fig. 4. These results validate that the proposed OGRL is comparable to the best online algorithms and superior to other offline algorithms in terms of packet loss rate.

V. Conclusion

This paper proposed an effective OGRL framework to solve the resource allocation optimization problem in URLLC services. Unlike existing algorithms, the proposed framework exhibited a better generalization ability in various scenarios with different active users and packet arrival rates and can ensure the safety of policy learning. Specifically, a GraphSage-based processing network was presented to process the dynamically changing active users. In addition, a resource allocation algorithm based on offline RL is proposed to avoid the interaction with the environment during the training process, so as to ensure the safety of the policy learning process. Performance evaluation showed the effectiveness and superiority of the proposed OGRL compared with other baseline algorithms. Specifically, OGRL demonstrates comparable QoS and packet loss rate performance to the optimal online algorithm while outperforming other offline algorithms.

VI. Acknowledgment

This work was supported by the Beijing Natural Science Foundation (L202002 and L222003).

References

[1] ITU-R, "Draft New Recommendation ITU-R M.[IMT.VISION 2030 AND BEYOND]," June.2022.

[2] M. Farzanullah, H. V. Vu, and T. Le-Ngoc, "Deep reinforcement learning for joint user association and resource allocation in factory automation," in *IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 2059–2064, 2022.

[3] S. Wei, Y. Zou, X. Zhang, T. Zhang, and X. Li, "An integrated longitudinal and lateral vehicle following control system with radar and vehicle-to-vehicle communication," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1116–1127, 2019.

[4] M. Sereno, X. Wang, L. Besançon, M. J. McGuffin, and T. Isenberg, "Collaborative work in augmented reality: A survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 6, pp. 2530–2549, 2022.

[5] Y. Huang, S. Li, Y. T. Hou, and W. Lou, "Gpf: A GPU-based design to achieve ∼100 µs scheduling for 5G NR," in *Annual International Conference on Mobile Computing and Networking(MOBICOM)*, p. 207–222, Association for Computing Machinery, 2018.

[6] M. Andrews, "Probabilistic end-to-end delay bounds for earliest deadline first scheduling," in *International Conference on Computer Communications(INFOCOM)*, vol. 2, pp. 603–612 vol.2, 2000.

[7] H. Zhang, H. Wang, Y. Li, K. Long, and V. C. M. Leung, "Toward intelligent resource allocation on task-oriented semantic communication," *IEEE Wireless Communications*, vol. 30, no. 3, pp. 70–77, 2023.

[8] H. Zhang, W. Song, X. Liu, M. Sheng, W. Li, K. Long, and O. A. Dobre, "Intelligent channel prediction and power adaptation in leo constellation for 6g," *IEEE Network*, vol. 37, no. 2, pp. 110–117, 2023.

[9] W. R. Ghanem, V. Jamali, Y. Sun, and R. Schober, "Resource allocation for multi-user downlink miso ofdma-urllc systems," *IEEE Transactions on Communications*, vol. 68, no. 11, pp. 7184–7200, 2020.

[10] Z. Gu, C. She, W. Hardjawana, S. Lumb, D. McKechnie, T. Essery, and B. Vucetic, "Knowledge-assisted deep reinforcement learning in 5G scheduler design: From theoretical framework to implementation," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 7, pp. 2014–2028, 2021.

[11] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative Q-learning for offline reinforcement learning," *ArXiv*, vol. abs/2006.04779, 2020.

[12] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," *ArXiv*, vol. abs/1812.02900, 2018.

[13] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, "Soft actor-critic algorithms and applications," *ArXiv*, vol. abs/1812.05905, 2018.

[14] 3GPP, "Service Requirements for Cyber-Physical Control Applications in Vetical Domains; Protocol specification," Technical Specification (TS) 22.104, 3rd Generation Partnership Project (3GPP), 2018. Version 16.0.0.

[15] W. Yang, G. Durisi, T. Koch, and Y. Polyanskiy, "Quasi-static multiple-antenna fading channels at finite blocklength," *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 4232–4265, 2014.

[16] C. She, C. Yang, and T. Q. S. Quek, "Joint uplink and downlink resource configuration for ultra-reliable and low-latency communications," *IEEE Transactions on Communications*, vol. 66, no. 5, pp. 2266–2280, 2018.

[17] C. She, R. Dong, Z. Gu, Z. Hou, Y. Li, W. Hardjawana, C. Yang, L. Song, and B. Vucetic, "Deep learning for ultra-reliable and low-latency communications in 6G networks," *IEEE Network*, vol. 34, no. 5, pp. 219–225, 2020.