Joint User Scheduling, Phase Shift Control, and Beamforming Optimization in Intelligent Reflecting Surface-Aided Systems

Rui Huang, *Graduate Student Member, IEEE*, and Vincent W.S. Wong, *Fellow, IEEE*

***Abstract*—In this paper, we formulate a joint uplink scheduling, phase shift control, and beamforming optimization problem in intelligent reflecting surface (IRS)-aided systems. We consider maximizing the aggregate throughput and achieving the proportional fairness as objectives. We propose a deep reinforcement learning-based user scheduling, phase shift control, beamforming optimization (DUPB) algorithm to solve the joint problem. The proposed DUPB algorithm applies the neural combinatorial optimization (NCO) technique to solve the user scheduling subproblem, in which a stochastic user scheduling policy is learned by deep neural networks with attention mechanism. Curriculum learning with deep deterministic policy gradient (CL-DDPG) is used in the proposed DUPB algorithm to jointly optimize the phase shift control and beamforming vectors. The knowledge on the hidden convexity of the joint problem is exploited to facilitate the policy learning in CL-DDPG. Simulation results show that, with the maximum aggregate throughput as the objective, the proposed DUPB algorithm achieves an aggregate throughput that is higher than the alternating optimization (AO)-based algorithms. Moreover, the throughput fairness among the users is improved when proportional fairness is used as the objective. The proposed DUPB algorithm outperforms the AO-based algorithms in terms of runtime when the number of reflecting elements is large.**

***Index Terms*—Intelligent reflecting surface (IRS), deep reinforcement learning (DRL), neural combinatorial optimization (NCO), curriculum learning (CL), uplink scheduling.**

I. Introduction

Intelligent reflecting surface (IRS) is a reconfigurable planar surface with multiple passive reflecting elements. Each reflecting element can perform a phase shift to the incident signal independently and reflect the shifted signal to a receiver. An IRS-aided system serving four users to perform uplink transmissions is shown in Fig. 1. Apart from the direct channels between the base station and users, IRS introduces additional propagation channels. When the line-of-sight (LoS) links between the base station and users are blocked by obstacles, deploying an IRS can create virtual LoS channels to facilitate data transmission and improve the coverage of the base station. By properly control the phase shift of the reflecting elements on IRS, the base station can mitigate interference and allow multiple users to share a physical resource block (PRB) for uplink transmission. IRS can be categorized as a passive holographic multiple-input multiple-output surface (HMIMOS) since the reflecting elements can be powered by energy harvesting module [2]. IRS can be combined with other physical layer techniques, including full-duplex communications and energy harvesting communications [3]. Potential applications of IRS include unmanned aerial vehicle (UAV) networks and virtual reality [4], [5].

Existing research on resource allocation in IRS-aided systems mostly focus on the beamforming optimization at the base station and the phase shift control of IRS [6]–[15]. The beamforming and phase shift optimization for IRS-aided systems with single user is studied in [6], [7]. The authors in [9] determined the ergodic rate of an IRS-aided system with interference from a secondary user and proposed a parallel coordinate descent-based algorithm to optimize the phase shift. The authors in [10] studied multiuser beamforming with practical amplitude variation in an IRS-aided system. An alternating optimization (AO)-based algorithm is proposed to solve the joint optimization problem. The aggregate throughput maximization problem in IRS-aided full-duplex systems was investigated in [11]. The authors solved the joint phase shift control, power control, and beamforming optimization problem using AO with successive convex approximation (SCA). Moreover, the authors in [12] investigated the joint beamforming and phase shift control in an IRS-aided multiuser system under both perfect and imperfect channel information. The beam pattern and channel estimation in a terahertz massive multiple-input multiple-output (MIMO) system with holographic IRS was investigated in [13]. The joint beamforming and phase shift control for maximizing the physical layer security in IRS-aided systems has been studied in [14]. In addition, the authors in [15] used sequential fractional programming to solve the joint phase shift and power control problem for maximizing the energy efficiency of the IRS-aided systems. Fractional programming (FP) technique [16] was applied in [12] and [14] to develop low-complexity beamforming and phase shift control algorithms. Although the aforementioned works studied the optimization of beamforming and phase shift control, the uplink user scheduling problem in IRS-aided systems has not been investigated. For an IRS-aided system with multiple users, it is beneficial for the base station to

Manuscript received on November 21, 2020; revised on April 20, 2021, September 26, 2021, and January 8, 2022; accepted on March 6, 2022. This paper has been published in part in the *Proceedings of IEEE Global Communications Conference (GLOBECOM)*, Taipei, Taiwan, Dec. 2020 [1]. The editor coordinating the review of this paper and approving it for publication was Xiang Cheng. (*Corresponding author: Vincent W.S. Wong.*)

R. Huang and V. W.S. Wong are with the Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC, V6T 1Z4, Canada (e-mail: {ruihuang, vincentw}@ece.ubc.ca).

Color versions of one or more of the figures in this paper are available online at https://ieeexplore.ieee.org.

Digital Object Identifier xx.xxxx/TWC.xxxx.xxxxxxx

properly schedule the transmission of the users, such that the interference between the users can be mitigated. Moreover, in the existing AO-based approaches, the iterative optimization process has to be invoked whenever the base station observes a change in the channel states. This may lead to a high computational complexity.

Deep deterministic policy gradient (DDPG) [17] is a deep reinforcement learning (DRL) technique to solve decision problems where the action space is continuous. The authors in [18] proposed a DDPG-based phase shift control algorithm to maximize the throughput of an IRS-aided single-user multiple-input single-output (MISO) system. The authors in [19] jointly optimized the phase shift control and beamforming vectors with DDPG to maximize the aggregate throughput of an IRS-aided multiuser MISO system. While the aforementioned DDPG-based algorithms [18], [19] can obtain a high-quality suboptimal solution, results showed that conventional optimization approaches (e.g., FP and semidefinite relaxation (SDR)) can still achieve a higher throughput than the existing DDPG-based phase shift control algorithms. Hence, the performance of the existing DDPG-based algorithms may further be improved by exploiting the knowledge on the hidden convexity of the problem. In addition, as the vanilla DDPG is developed for the optimization of continuous variables, the aforementioned algorithms do not take the user scheduling, which involves the optimization of integer variables, into account. A new DRL framework needs to be designed for the IRS-aided systems where both integer and continuous variables are required to be optimized jointly.

Neural combinatorial optimization (NCO) is a model-free learning technique for solving combinatorial problems [20], [21]. In NCO, a stochastic policy for solving the combinatorial optimization problem is learned by training deep neural networks (DNNs) based on reinforcement learning (RL). For the user scheduling problem in IRS-aided systems, a DNN can be trained to learn the correlation between the users based on their channel information. The correlation between the users can be characterized by the DNNs with *attention mechanism* [22] to reflect how strongly scheduling a particular user will affect the uplink transmission of other users in the system. A stochastic policy can be obtained to determine the user scheduling that yields the desired system performance in an online manner. NCO has been applied to solve some classical combinatorial problems, including the travelling salesman problem and vehicle routing problem [20], [21], [23].

Recently, NCO has been applied to study the user scheduling and resource allocation problems in wireless communication systems [24], [25]. The authors in [24] proposed a channel assignment algorithm for power-domain non-orthogonal multiple access (NOMA) based on DNNs with attention mechanism. The proposed algorithm outperforms some of the existing channel assignment algorithms in the literature. The authors in [25] studied the user pairing in multicell power-domain NOMA, and showed that the NCO-based approach can achieve near-optimal performance. Compared with power-domain NOMA, the user scheduling problem in IRS-aided systems is more challenging due to the following reasons. While the channel information in power-domain NOMA only involves the signal power, it is necessary to take both the amplitude and phase of the uplink channel into account to determine the optimal user scheduling in IRS-aided systems. Moreover, the user scheduling is coupled with phase shift control of the IRS and beamforming at the base station.

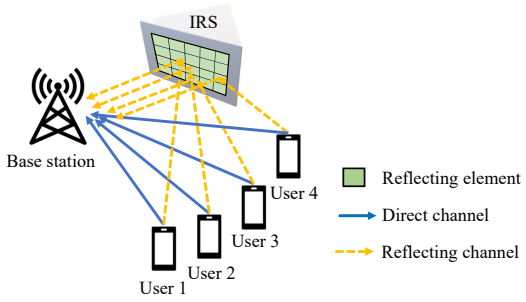


Fig. 1. An IRS-aided system with four users sharing one uplink PRB. The direct channels are denoted by solid blue lines, while the reflecting channels are denoted by dashed yellow lines.

To address the aforementioned issues, in this paper, we investigate the joint optimization of user scheduling, phase shift control, and beamforming optimization problem in IRS-aided systems. We consider both maximizing the aggregate throughput and achieving the proportional fairness as objectives. Obtaining the optimal solution of the joint optimization problem is challenging since the optimization variables are coupled and the formulated problem is nonconvex. To tackle the challenges, we propose a DRL-based user scheduling, phase shift control, beamforming optimization (DUPB) algorithm, in which we first use NCO to determine the user scheduling. The DNNs with attention mechanism are trained to learn a stochastic user scheduling policy based on RL. We then solve the phase shift control and beamforming subproblem by proposing a curriculum learning (CL) DDPG algorithm. We exploit DDPG and the hidden convexity of the joint problem to optimize the phase shift and beamforming variables. The contributions of this paper are as follows:

- We formulate a joint optimization problem for maximizing the aggregate throughput and achieving the proportional fairness in IRS-aided systems as a mixed-integer nonlinear optimization problem. We decompose the problem into a subproblem for user scheduling, and a subproblem for joint phase shift control and beamforming optimization.
- We use NCO to learn the stochastic policy for user scheduling in an online manner. We employ two DNN modules, i.e., an encoder module and a decoder module. The encoder module learns the high-dimensional representations of the channel information of the users, and these representations are used by the decoder module to determine the stochastic policy. The DNNs are trained based on RL without requiring the optimal solution of the problem during the training phase.
- We further propose a CL-DDPG algorithm to solve the joint phase shift control and beamforming optimization subproblem. The proposed CL-DDPG algorithm employs

an actor-critic method to learn a policy for optimizing the phase shift and beamforming variables. We improve the performance of the vanilla DDPG by exploiting the prior knowledge on the hidden convexity of the joint subproblem. Curriculum learning [26] is used to design the reward function such that the suboptimal solution obtained by a baseline algorithm can facilitate the learning in the early stage, which contributes to a better solution.
- Simulation results show that the proposed DUPB algorithm achieves an aggregate throughput that is higher than the AO-based algorithms and greedy scheduling. The throughput fairness among the users can be improved by using the proposed DUPB algorithm under the proportional fairness objective. Moreover, the proposed DUPB algorithm requires a lower runtime than the AO-based algorithms when the number of reflecting elements is large. We evaluate the impact of imperfect channel estimation on the achievable throughput of the proposed DUPB algorithm. Our results also show that both modules (i.e., the NCO algorithm for user scheduling and the CL-DDPG algorithm for phase shift control and beamforming optimization) in the proposed DUPB algorithm contribute to an aggregate throughput that is higher than the AO-based algorithm with FP.

The remainder of this paper is organized as follows. The system model and problem formulation are presented in Section II. In Section III, we apply NCO technique to solve the user scheduling subproblem. In Section IV, we present the CL-DDPG algorithm and the overall framework of the DUPB algorithm. Simulation results are shown in Section V. Conclusions are drawn in Section VI.

Notations: In this paper, we use upper-case and lower-case boldface letters to denote matrices and column vectors, respectively. $\mathbb{C}^{M\times N}$ denotes the set of $M\times N$ complex-valued matrices. $\boldsymbol{A}^H$ denotes the conjugate transpose of matrix $\boldsymbol{A}$. $\mathrm{diag}(\boldsymbol{x})$ returns a diagonal matrix where the diagonal elements are given by the elements of vector $\boldsymbol{x}$.

II. System Model

We consider an IRS-aided system with one base station, one IRS, and N users. The base station is equipped with K antennas, while each user equipment (UE) has only one antenna. The set of users is denoted by $\mathcal{N} = \{1, 2, \ldots, N\}$. Time is divided into intervals of equal duration. The time interval $[t, t+1)$ is referred to as time slot t, where $t \in \mathcal{T} = \{1, 2, \ldots\}$. We assume the base station is allocated with one PRB to serve the users in each time slot $t \in \mathcal{T}$.

An IRS with L_R reflecting elements is deployed to facilitate uplink transmission of the users. In each time slot $t \in \mathcal{T}$, the base station schedules M users to perform uplink transmission using one PRB. We use binary control variable $x_n(t) \in \{0, 1\}$ to indicate whether user $n \in \mathcal{N}$ is scheduled for uplink transmission in time slot t. We set $x_n(t) = 1$ if user n is scheduled in time slot t, and $x_n(t) = 0$ otherwise. We have

$$x_n(t) \in \{0, 1\},\ n \in \mathcal{N}, \tag{1}$$

$$\sum_{n\in\mathcal{N}} x_n(t) = M. \tag{2}$$

We use vector $\boldsymbol{x}(t) = (x_1(t), x_2(t), \ldots, x_N(t))$ to collect $x_n(t)$, $n \in \mathcal{N}$, in time slot t.

Let $\boldsymbol{h}_{D,n}(t) \in \mathbb{C}^K$ and $\boldsymbol{h}_{R,n}(t) \in \mathbb{C}^{L_R}$ denote the channel response between user $n \in \mathcal{N}$ and the base station (i.e., the direct channel) and the channel response between user n and the IRS (i.e., the reflecting channel) in time slot $t \in \mathcal{T}$, respectively. The channel response between the IRS and the base station in time slot t is denoted by matrix $\boldsymbol{G}(t) \in \mathbb{C}^{L_R\times K}$. We assume perfect channel estimation at the base station[1]. In addition, we assume block fading channels, where the block length is larger than the time period of the PRB. We use matrix $\boldsymbol{\Psi}(t)$ to denote the phase shift matrix of the IRS in time slot t. We have

$$\boldsymbol{\Psi}(t) = \mathrm{diag}(e^{j\psi_1(t)}, \cdots, e^{j\psi_{L_R}(t)}) \in \mathbb{C}^{L_R\times L_R}, \tag{3}$$

where $\psi_l(t)$, $l \in \{1, \ldots, L_R\}$ is the phase shift of the l-th reflecting element on the IRS in time slot t. We have the following constraint on the phase shift of the reflecting element

$$\psi_l(t) \in [0,\ 2\pi),\ \ l \in \{1, \ldots, L_R\}. \tag{4}$$

We assume the scheduled users always use the maximum transmit power $P^{\max}$ to transmit their signals. The received signal of user $n \in \mathcal{N}$ at the base station in time slot t is given by

$$\begin{aligned}\boldsymbol{y}_n(t) = x_n(t)\sqrt{P^{\max}}\big(&\boldsymbol{h}_{D,n}(t)\, s_n(t) \\ &+ \boldsymbol{G}^H(t)\boldsymbol{\Psi}(t)\boldsymbol{h}_{R,n}(t)s_n(t)\big) + \boldsymbol{f}_n(t) + \boldsymbol{\varrho},\end{aligned} \tag{5}$$

where $s_n(t) \in \mathbb{C}$ is the symbol of user n in time slot t with unit power, $\boldsymbol{\varrho}$ is the complex Gaussian noise with zero mean and variance σ^2, and $\boldsymbol{f}_n(t)$ is the interference from the remaining users to user n in time slot t. $\boldsymbol{f}_n(t)$ is given by

$$\begin{aligned}\boldsymbol{f}_n(t) = \sum_{j\in\mathcal{N}\setminus\{n\}} x_j(t)\sqrt{P^{\max}}\big(&\boldsymbol{h}_{D,j}(t)\, s_j(t) \\ &+ \boldsymbol{G}^H(t)\boldsymbol{\Psi}(t)\boldsymbol{h}_{R,j}(t)s_j(t)\big).\end{aligned} \tag{6}$$

The signal-to-interference-plus-noise ratio (SINR) of user n in time slot t is given by equation (7) below. In (7), $\boldsymbol{b}_n(t) \in \mathbb{C}^K$ is the beamforming vector of user n's signal in time slot

[1]Note that perfect channel information may be difficult to obtain in practical systems. The potential impact of imperfect channel estimation on the studied problem is evaluated in Section V-D.

$$\Gamma_n(\boldsymbol{x}(t), \boldsymbol{\Psi}(t), \boldsymbol{b}_n(t)) = \frac{x_n(t)P^{\max}\left|\boldsymbol{b}_n^H(t)\boldsymbol{h}_{D,n}(t) + \boldsymbol{b}_n^H(t)\boldsymbol{G}^H(t)\boldsymbol{\Psi}(t)\boldsymbol{h}_{R,n}(t)\right|^2}{\sum\limits_{j\in\mathcal{N}\setminus\{n\}} x_j(t)P^{\max}\left|\boldsymbol{b}_n^H(t)\boldsymbol{h}_{D,j}(t) + \boldsymbol{b}_n^H(t)\boldsymbol{G}^H(t)\boldsymbol{\Psi}(t)\boldsymbol{h}_{R,j}(t)\right|^2 + \sigma^2\left\|\boldsymbol{b}_n(t)\right\|_2^2}. \tag{7}$$

t at the base station. We use $\boldsymbol{b}(t) = (\boldsymbol{b}_1(t), \ldots, \boldsymbol{b}_N(t))$ to collect the beamforming vectors of all users in time slot t. The achievable throughput (bits/(time slot)/Hz) of user n can be determined as follows:

$$R_n(\boldsymbol{x}(t), \boldsymbol{\Psi}(t), \boldsymbol{b}_n(t)) = \log_2(1 + \Gamma_n(\boldsymbol{x}(t), \boldsymbol{\Psi}(t), \boldsymbol{b}_n(t))). \tag{8}$$

When proportional fairness is used as the objective, we determine the time average of the achievable throughput of user n up to time slot t based on the moving average [27] as shown in equation (9) at the bottom of this page, where T_c is the moving window size. We set $\overline{R}_n(1) = 1$, $n \in \mathcal{N}$, such that all users are considered to have equal average throughput in the first time slot $t = 1$.

In this paper, we consider the joint optimization problem of user scheduling, phase shift control, and beamforming. The optimization problem in time slot $t \in \mathcal{T}$ is formulated as follows:

$$\begin{aligned} \underset{\boldsymbol{x}(t),\, \boldsymbol{\Psi}(t),\, \boldsymbol{b}(t)}{\text{maximize}} \quad & \sum_{n \in \mathcal{N}} w_n(t)\, R_n(\boldsymbol{x}(t), \boldsymbol{\Psi}(t), \boldsymbol{b}_n(t)) \\ \text{subject to} \quad & \text{constraints } (1), (2), (4). \end{aligned} \tag{10}$$

The optimization problem with the maximum aggregate throughput objective can be obtained by setting $w_n(t) = 1$ for all $n \in \mathcal{N}$ in problem (10). For the proportional fairness objective, we set $w_n(t) = \frac{1}{\overline{R}_n(t)}$ for $n \in \mathcal{N}$ [16], [27]. The optimal solution of problem (10) is difficult to obtain since the problem is nonconvex due to the weighted fractional objective function and the phase shift constraint (4). Moreover, problem (10) has binary control variables $\boldsymbol{x}(t)$ and the optimization variables $\boldsymbol{x}(t)$, $\boldsymbol{\Psi}(t)$, $\boldsymbol{b}(t)$ are coupled.

In time slot $t \in \mathcal{T}$, we can decompose problem (10) into a user scheduling subproblem and a subproblem for joint phase shift control and beamforming optimization. In particular, given $\boldsymbol{\Psi}(t)$ and $\boldsymbol{b}(t)$, the subproblem for user scheduling in time slot t is given by

$$\begin{aligned} \underset{\boldsymbol{x}(t)}{\text{maximize}} \quad & \sum_{n \in \mathcal{N}} w_n(t)\, R_n(\boldsymbol{x}(t)) \\ \text{subject to} \quad & \text{constraints } (1) \text{ and } (2). \end{aligned} \tag{11}$$

Subproblem (11) is a combinatorial optimization problem with a total of $\binom{N}{M}$ feasible user scheduling selections in each time slot. Given the user scheduling vector $\boldsymbol{x}(t)$, the joint subproblem for phase shift control and beamforming optimization in time slot t is as follows:

$$\begin{aligned} \underset{\boldsymbol{\Psi}(t), \boldsymbol{b}(t)}{\text{maximize}} \quad & \sum_{n \in \mathcal{N}} w_n(t)\, R_n(\boldsymbol{\Psi}(t), \boldsymbol{b}(t)) \\ \text{subject to} \quad & \text{constraint } (4). \end{aligned} \tag{12}$$

Subproblem (12) is a nonconvex optimization problem with a multi-ratio fractional objective function. A suboptimal solution of problem (10) can be obtained by solving subproblem (12) for all $\binom{N}{M}$ user scheduling selections. This approach is computationally expensive, especially when problem (10) is required to be solved in each time slot and the number of users is large. In the following sections, we propose a DUPB algorithm to solve problem (10) with a lower computational complexity.

III. DUPB Algorithm: NCO-based Algorithm for User Scheduling

We first propose an NCO-based algorithm for determining the user scheduling in an online manner. In the proposed NCO-based algorithm, we solve the user scheduling subproblem using an RL framework. A stochastic user scheduling policy is learned by the DNNs with attention mechanism. For notational simplicity, we drop the time index t in the subsequent sections.

A. RL Framework and Stochastic Policy

We propose the following RL framework for solving the user scheduling subproblem (11). The states, actions, and rewards are defined as follows:

1) States: We first concatenate the rows of channel gain matrix $\boldsymbol{G}$ to obtain a vector of size KL_R, which is denoted by $\boldsymbol{g}$. Then, we use vector $\boldsymbol{v}_n$ to collect user n's channel information, along with the weight w_n. In particular, vector $\boldsymbol{v}_n$ is given by

$$\boldsymbol{v}_n = [\boldsymbol{h}_{D,n},\, \boldsymbol{h}_{R,n},\, \boldsymbol{g},\, w_n],\ n \in \mathcal{N}, \tag{13}$$

where $[\cdot, \cdot, \cdot]$ denotes the concatenation operator. The size of vector $\boldsymbol{v}_n$ is $K + L_R + KL_R + 1$.

State $\mathcal{S}$ consists of the channel information and the weights w_n of all N users. It is given by

$$\mathcal{S} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_N\}. \tag{14}$$

State $\mathcal{S}$ is a *set* that collects the global information [28], [29]. In the remainder of this paper, we use the terms state $\mathcal{S}$ and set $\mathcal{S}$ interchangeably.

2) Actions: The action is to schedule M users to perform uplink transmission. Given state $\mathcal{S}$, the action is equivalent to finding a subset $\mathcal{U}$ which consists of M different vectors (or elements) from $\mathcal{S}$. That is,

$$\mathcal{U} = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_M\}, \tag{15}$$

where $\boldsymbol{u}_l \in \mathcal{S}$ and $\boldsymbol{u}_l \neq \boldsymbol{u}_{l'}$, $l, l' \in \{1, \ldots, M\}$, and $l \neq l'$. Given subset $\mathcal{U}$, we obtain the corresponding user scheduling vector $\boldsymbol{x}^\star$ by setting $x_n = 1$ if $\boldsymbol{v}_n \in \mathcal{U}$. If $\boldsymbol{v}_n \notin \mathcal{U}$, then we set $x_n = 0$.

$$\overline{R}_n(t) = \begin{cases} 1, & \text{if } t = 1, \\ \frac{1}{t-1} \sum_{\tau=1}^{t-1} R_n(\boldsymbol{x}(\tau), \boldsymbol{\Psi}(\tau), \boldsymbol{b}_n(\tau)), & \text{if } t = \{2, 3, \ldots, T_c\}, \\ (1 - \frac{1}{T_c})\overline{R}_n(t-1) + \frac{1}{T_c} R_n(\boldsymbol{x}(t-1), \boldsymbol{\Psi}(t-1), \boldsymbol{b}_n(t-1)), & \text{otherwise.} \end{cases} \tag{9}$$

3) Rewards: After determining subset $\mathcal{U}$, the reward $r(\mathcal{U}) \in \mathbb{R}$ is determined by solving the joint phase shift and beamforming optimization problem in (12) for fixed $\boldsymbol{x}^\star$. We propose a CL-DDPG algorithm for solving problem (12). The details of the CL-DDPG algorithm will be presented in Section IV. We denote the phase shift matrix and beamforming vector that are obtained by solving problem (12) as $\boldsymbol{\Psi}^\star$ and $\boldsymbol{b}^\star$, respectively. Then, the reward $r(\mathcal{U})$ is determined as follows:

$$r(\mathcal{U}) = \sum_{n\in\mathcal{N}} w_n \, R_n(\boldsymbol{x}^\star, \boldsymbol{\Psi}^\star, \boldsymbol{b}_n^\star). \tag{16}$$

The stochastic policy for choosing the action, i.e., scheduling M users, can be determined by the conditional probability of selecting a particular subset $\mathcal{U}$ given state $\mathcal{S}$, i.e., $p(\mathcal{U}\,|\,\mathcal{S})$. Using the chain rule, this probability can be factorized as follows [21]:

$$p(\mathcal{U}\,|\,\mathcal{S}) = \prod_{l=1}^{M} p(\boldsymbol{u}_l\,|\,\mathcal{S}, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_{l-1}). \tag{17}$$

To obtain the optimal user scheduling, we aim to find the optimal stochastic policy that leads to the maximum expected reward. In particular, the optimal stochastic policy is given by

$$p^\star(\mathcal{U}\,|\,\mathcal{S}) = \arg\max \; \mathbb{E}_{\mathcal{U}'\sim p(\mathcal{U}\,|\,\mathcal{S})}\big[r(\mathcal{U}')\big]. \tag{18}$$

To this end, we propose an NCO-based algorithm to learn the optimal stochastic policy in (18). The details of the proposed algorithm are presented in the following subsections.

B. Encoder Module

We denote the learnable parameters of the DNN modules that are employed to learn the stochastic user scheduling policy as $\boldsymbol{\Phi}$. The parameterized stochastic user scheduling policy is denoted as $p_{\boldsymbol{\Phi}}(\mathcal{U}\,|\,\mathcal{S})$. We use an *encoder* DNN module to learn the underlying structures and abstraction of the information in set $\mathcal{S}$, which are referred to as the *embedding* of the users [21]. The DNN structure for the encoder module is shown in Fig. 2. For vector $\boldsymbol{v}_n \in \mathcal{S}$, we use an *initial embedding module* to obtain its embedding $\boldsymbol{v}_n^{\mathrm{E}}$. As shown on the right-hand side of Fig. 2, the initial embedding module consists of fully-connected (FC) layers and residual network (ResNet) modules. N_{Res} ResNet modules are stacked in order to increase the depth of the initial embedding module and provide a higher representation capacity. Each ResNet module has two FC layers, two rectified linear unit (ReLU) layers, and one residual connection. The residual connection is added to overcome the vanishing gradient issue due to the depth of the network [30], and to facilitate the learning by eliminating the singularities of FC layers [31].

In the initial embedding module, the first FC layer projects the input vector $\boldsymbol{v}_n$ to a d_{Res}-dimensional space, where d_{Res} is a constant. The output of the first FC layer is given by:

$$\boldsymbol{v}_n^{(0)} = \boldsymbol{W}^{\mathrm{FC1}}\,\boldsymbol{v}_n + \boldsymbol{b}^{\mathrm{FC1}}, \tag{19}$$

where the weights $\boldsymbol{W}^{\mathrm{FC1}} \in \mathbb{R}^{d_{\mathrm{Res}}\times d_i}$ and biases $\boldsymbol{b}^{\mathrm{E}} \in \mathbb{R}^{d_{\mathrm{Res}}}$ are learnable parameters, and $d_i = K + L_R + KL_R + 1$ is the size of vector $\boldsymbol{v}_n$. The vector $\boldsymbol{v}_n^{(0)}$ is then fed into the subsequent ResNet modules. We denote $\boldsymbol{W}_1^{(i)}$ and $\boldsymbol{W}_2^{(i)} \in \mathbb{R}^{d_{\mathrm{Res}}\times d_{\mathrm{Res}}}$ as the weights, and $\boldsymbol{b}_1^{(i)}$ and $\boldsymbol{b}_2^{(i)} \in \mathbb{R}^{d_{\mathrm{Res}}}$ as the biases of the first and second FC layers in the i-th ResNet module, respectively, for $i = 1, \ldots, N_{\mathrm{Res}}$. Given vector $\boldsymbol{v}_n^{(0)}$, $n \in \mathcal{N}$, the output of the i-th ResNet module is given by:

$$\begin{aligned}\boldsymbol{v}_n^{(i)} = \mathrm{ReLU}\Big(\boldsymbol{W}_2^{(i)}\big(\mathrm{ReLU}(\boldsymbol{W}_1^{(i)}\boldsymbol{v}_n^{(i-1)} + \boldsymbol{b}_1^{(i)})\big) + \boldsymbol{b}_2^{(i)}\Big) \\ + \boldsymbol{v}_n^{(i-1)}, \qquad i = 1, \ldots, N_{\mathrm{Res}},\end{aligned} \tag{20}$$

where $\mathrm{ReLU}(\cdot)$ denotes the ReLU activation function. The output of the N_{Res}-th (i.e., the last) ResNet module is passed through another FC layer with weights $\boldsymbol{W}^{\mathrm{FC2}} \in \mathbb{R}^{d_h\times d_{\mathrm{Res}}}$ and biases $\boldsymbol{b}^{\mathrm{FC2}} \in \mathbb{R}^{d_h}$ to obtain the initial embedding $\boldsymbol{v}_n^{\mathrm{E}}$. We have $\boldsymbol{v}_n^{\mathrm{E}} = \boldsymbol{W}^{\mathrm{FC2}}\,\boldsymbol{v}_n^{(N_{\mathrm{Res}})} + \boldsymbol{b}^{\mathrm{FC2}}$. The learnable parameters of the initial embedding module $\boldsymbol{\Phi}^{\mathrm{IEM}}$ are given by

$$\begin{aligned}\boldsymbol{\Phi}^{\mathrm{IEM}} = \Big(\boldsymbol{W}^{\mathrm{FC1}}, \boldsymbol{W}_1^{(i)}, \boldsymbol{W}_2^{(i)}, \boldsymbol{W}^{\mathrm{FC2}}, \boldsymbol{b}^{\mathrm{FC1}}, \boldsymbol{b}_1^{(i)}, \boldsymbol{b}_2^{(i)}, \boldsymbol{b}^{\mathrm{FC2}}\Big), \\ i = 1, \ldots, N_{\mathrm{Res}}.\end{aligned} \tag{21}$$

After obtaining the initial embedding, we use *attention mechanism* [22] to capture the user interference and the combinatorial structure of the user scheduling subproblem. The attention mechanism can be considered as an information exchange process between the embeddings of vectors. By exchanging the information, the embedding $\boldsymbol{v}_n^{\mathrm{E}}$ not only provides the multidimensional representations of the channel information and weight w_n of user n, but also shows how it relates to the other users. We generate three additional vectors, namely, *key* $\boldsymbol{k}_n$, *query* $\boldsymbol{q}_n$, and *value* $\boldsymbol{z}_n$, for each vector based on the embedding $\boldsymbol{v}_n^{\mathrm{E}}$ as follows:

$$\boldsymbol{k}_n = \boldsymbol{W}_{\mathrm{en}}^{\mathrm{K}}\,\boldsymbol{v}_n^{\mathrm{E}}, \;\; \boldsymbol{q}_n = \boldsymbol{W}_{\mathrm{en}}^{\mathrm{Q}}\,\boldsymbol{v}_n^{\mathrm{E}}, \;\; \boldsymbol{z}_n = \boldsymbol{W}_{\mathrm{en}}^{\mathrm{Z}}\,\boldsymbol{v}_n^{\mathrm{E}}, \tag{22}$$

where matrices $\boldsymbol{W}_{\mathrm{en}}^{\mathrm{K}}, \boldsymbol{W}_{\mathrm{en}}^{\mathrm{Q}} \in \mathbb{R}^{d_k\times d_h}$, and $\boldsymbol{W}_{\mathrm{en}}^{\mathrm{Z}} \in \mathbb{R}^{d_z\times d_h}$ are learnable parameters, whereas d_k and d_z are constants. Using the attention mechanism, the embedding of a vector may receive values from the embeddings of other vectors. We compute the *compatibility* $\delta_{n,j} \in \mathbb{R}$ of the two vectors as $\delta_{n,j} = \frac{\boldsymbol{q}_n^T\boldsymbol{k}_j}{\sqrt{d_k}}$. The attention weights $a_{n,j} \in [0,\,1]$ can be obtained using the softmax function as $a_{n,j} = \frac{e^{\delta_{n,j}}}{\sum_{j'\in\mathcal{N}} e^{\delta_{n,j'}}}$. Let $\boldsymbol{z}_n'$ denote the value that embeddings $\boldsymbol{v}_n^{\mathrm{E}}$ received from the embeddings of other vectors. We have

$$\boldsymbol{z}_n' = \sum_{j\in\mathcal{N}} a_{n,j}\boldsymbol{z}_j. \tag{23}$$

We construct a new embedding $\hat{\boldsymbol{v}}_n^{\mathrm{E}}$ of vector $\boldsymbol{v}_n$ by combining $\boldsymbol{v}_n^{\mathrm{E}}$ with the received $\boldsymbol{z}_n'$, which is $\hat{\boldsymbol{v}}_n^{\mathrm{E}} = \boldsymbol{v}_n^{\mathrm{E}} + \boldsymbol{z}_n'$. The final

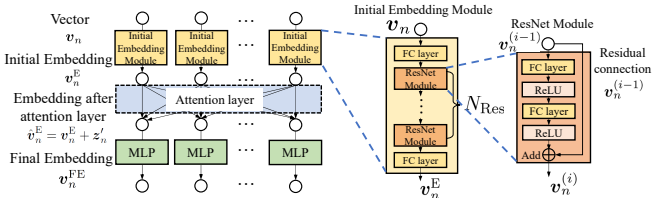


Fig. 2. The network structure of the encoder module. The encoder module learns the embeddings of the input vectors with an initial embedding module, an attention layer [22], and an MLP layer.

embedding of vector $\boldsymbol{v}_n$, which is denoted by $\boldsymbol{v}_n^{\mathrm{FE}}$, is obtained by passing the embedding $\hat{\boldsymbol{v}}_n^{\mathrm{E}}$ through a multilayer perceptron (MLP) module. The dimension of the output layer of the MLP module is d_h. We denote the learnable parameters of MLP modules as $\boldsymbol{\Phi}^{\mathrm{MLP}}$. The parameters in the encoder module $\boldsymbol{\Phi}_{\mathrm{en}}$ are given by $\boldsymbol{\Phi}_{\mathrm{en}} = (\boldsymbol{\Phi}^{\mathrm{IEM}}, \boldsymbol{W}_{\mathrm{en}}^{\mathrm{K}}, \boldsymbol{W}_{\mathrm{en}}^{\mathrm{Q}}, \boldsymbol{W}_{\mathrm{en}}^{\mathrm{Z}}, \boldsymbol{\Phi}^{\mathrm{MLP}})$. The outputs of the encoder module are the final embeddings of the users.

C. NCO-based Algorithm: Context Embedding and Decoder

In the decoder module, we first combine the final embeddings of all vectors in state $\mathcal{S}$ to obtain the aggregate embedding using the weighted summation $\boldsymbol{v}_{\mathrm{G}}^{\mathrm{E}} = \sum_{n\in\mathcal{N}} \eta_n \boldsymbol{v}_n^{\mathrm{FE}}$, where $\eta_n \in (0,\,1)$ is the weight of the final embedding of user n. Ideally, the weight η_n should be adjusted flexibly to reflect the contribution of embedding $\boldsymbol{v}_n^{\mathrm{E}}$ (i.e., user n) to the reward. Inspired by the feature aggregation in deep multiple instance learning [29], [32], we employ an embedding aggregation layer to determine the weight as $\eta_n = \frac{e^{\beta_n}}{\sum_{n'\in\mathcal{N}} e^{\beta_{n'}}}$, where $\beta_n = \boldsymbol{\nu}^T \tanh(\boldsymbol{W}^{\mathrm{A}} \boldsymbol{v}_n^{\mathrm{FE}})$ [32]. Vector $\boldsymbol{\nu} \in \mathbb{R}^{d_a}$ and matrix $\boldsymbol{W}^{\mathrm{A}} \in \mathbb{R}^{d_a \times d_h}$ are learnable parameters of the embedding aggregation layer, and d_a is a constant. Using the aggregate embedding $\boldsymbol{v}_{\mathrm{G}}^{\mathrm{E}}$, the decoder module constructs a *context embedding* for generating the conditional probability $p_{\boldsymbol{\Phi}}(\boldsymbol{u}_l \,|\, \mathcal{S}, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_{l-1})$, which is given by

$$\boldsymbol{v}_{\mathrm{c}}^{\mathrm{E}} = \begin{cases} [\boldsymbol{v}_{\mathrm{G}}^{\mathrm{E}},\, \boldsymbol{v}_{\boldsymbol{u}_0}^{\mathrm{FE}},\, \boldsymbol{w},\, M], & \text{if } l = 1, \\ [\boldsymbol{v}_{\mathrm{G}}^{\mathrm{E}},\, \boldsymbol{v}_{\mathcal{U}_{l-1}}^{\mathrm{E}},\, \boldsymbol{w},\, M-l+1\,], & \text{if } l > 1, \end{cases} \tag{24}$$

where $\boldsymbol{v}_{\mathcal{U}_{l-1}}^{\mathrm{E}}$ is the aggregate embedding of the previously selected vectors. It is given by

$$\boldsymbol{v}_{\mathcal{U}_{l-1}}^{\mathrm{E}} = \frac{1}{l-1} \sum_{n\in\{n' \,|\, \boldsymbol{v}_{n'}^{\mathrm{E}} \in \{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_{l-1}\}\}} \boldsymbol{v}_n^{\mathrm{FE}}, \tag{25}$$

and vector $\boldsymbol{w} = (w_1, \ldots, w_N)$. $\boldsymbol{v}_{\boldsymbol{u}_0}^{\mathrm{FE}}$ serves as a placeholder to maintain the constant size of $\boldsymbol{v}_{\mathrm{c}}^{\mathrm{E}}$ when the base station determines the first scheduled user. The last element in (24), i.e., $M-l+1$, is the number of remaining vectors that can be added into the subset. The size of the context embedding is given by $d_{\mathrm{c}} = 2d_h + N + 1$.

To obtain the stochastic policy, we compute the compatibility of the context embedding in (24) with the final embedding of each of the remaining users that can potentially be scheduled. We obtain the query of the context embedding, the keys and values of the vectors as follows:

$$\boldsymbol{q}_{\mathrm{c}} = \boldsymbol{W}_{\mathrm{de}}^{\mathrm{Q}} \boldsymbol{v}_{\mathrm{c}}^{\mathrm{E}}, \;\; \boldsymbol{k}_n = \boldsymbol{W}_{\mathrm{de}}^{\mathrm{K}} \boldsymbol{v}_n^{\mathrm{FE}}, \;\; \boldsymbol{z}_n = \boldsymbol{W}_{\mathrm{de}}^{\mathrm{Z}} \boldsymbol{v}_n^{\mathrm{FE}}, \tag{26}$$

where matrices $\boldsymbol{W}_{\mathrm{de}}^{\mathrm{Q}} \in \mathbb{R}^{d_h \times d_{\mathrm{c}}}$, $\boldsymbol{W}_{\mathrm{de}}^{\mathrm{K}} \in \mathbb{R}^{d_h \times d_k}$, and $\boldsymbol{W}_{\mathrm{de}}^{\mathrm{Z}} \in \mathbb{R}^{d_h \times d_z}$ project the context embedding and the final embeddings of the vectors to d_h dimensions. The compatibilities of context embedding with the final embedding of the remaining users can be determined by

$$\delta_{\mathrm{c},j} = \begin{cases} \delta^{\max} \tanh\left(\frac{\boldsymbol{q}_{\mathrm{c}}^T \boldsymbol{k}_j}{\sqrt{d_h}}\right), & \text{if } \boldsymbol{v}_j \text{ has not been selected}, \\ -\infty, & \text{otherwise}, \end{cases} \tag{27}$$

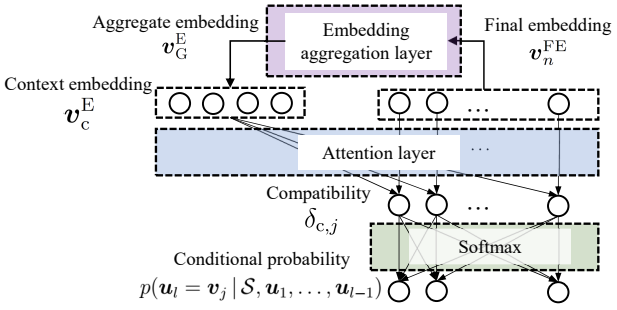


Fig. 3. The network structure of the decoder module. The decoder module generates the conditional probabilities based on the final embeddings provided by the encoder module.

where $\delta^{\max}$ is a constant. The conditional probability for selecting user j as the l-th scheduled user in the subset is given by

$$p(\boldsymbol{u}_l = \boldsymbol{v}_j \,|\, \mathcal{S}, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_{l-1}) = \frac{e^{\delta_{\mathrm{c},j}}}{\sum_{j'\in\mathcal{N}} e^{\delta_{\mathrm{c},j'}}}. \tag{28}$$

With the conditional probability in (28), the stochastic policy $p_{\boldsymbol{\Phi}}(\mathcal{U} \,|\, \mathcal{S})$ can be obtained based on (17). The learnable parameters in the decoder module is given by $\boldsymbol{\Phi}_{\mathrm{de}} = (\boldsymbol{\nu}, \boldsymbol{W}^{\mathrm{A}}, \boldsymbol{W}_{\mathrm{de}}^{\mathrm{K}}, \boldsymbol{W}_{\mathrm{de}}^{\mathrm{Q}}, \boldsymbol{W}_{\mathrm{de}}^{\mathrm{Z}})$. The overall learnable parameters are collected as $\boldsymbol{\Phi} = (\boldsymbol{\Phi}_{\mathrm{en}},\, \boldsymbol{\Phi}_{\mathrm{de}})$.

D. Learning Algorithm based on REINFORCE

The stochastic policy generated by the encoder and decoder modules is characterized by the learnable parameters $\boldsymbol{\Phi}$, i.e., $p_{\boldsymbol{\Phi}}(\mathcal{U} \,|\, \mathcal{S})$. We aim to find the parameters $\boldsymbol{\Phi}$, such that the expected reward of policy $p_{\boldsymbol{\Phi}}(\mathcal{U} \,|\, \mathcal{S})$ is maximized. This leads to the following optimization problem:

$$\min_{\boldsymbol{\Phi}} \; \mathcal{L}(\boldsymbol{\Phi} \,|\, \mathcal{S}) \triangleq \mathbb{E}_{\mathcal{U}'\sim p_{\boldsymbol{\Phi}}(\mathcal{U} \,|\, \mathcal{S})}\big[-r(\mathcal{U}')\big], \tag{29}$$

where $\mathcal{L}(\boldsymbol{\Phi} \,|\, \mathcal{S})$ is referred to as the *loss function*. We use the REINFORCE algorithm [33] to perform gradient descent and optimize $\boldsymbol{\Phi}$. The gradient estimation is given by

$$\nabla \mathcal{L}(\boldsymbol{\Phi} \,|\, \mathcal{S}) = \mathbb{E}_{\mathcal{U}'\sim p_{\boldsymbol{\Phi}}(\mathcal{U} \,|\, \mathcal{S})}\big[(b(\mathcal{S}) - r(\mathcal{U}'))\, \nabla \log p_{\boldsymbol{\Phi}}(\mathcal{U}' \,|\, \mathcal{S})\big], \tag{30}$$

where $b(\mathcal{S})$ is the reward obtained by a baseline policy on set $\mathcal{S}$. We use the best policy learned by the DNN modules so far during the training phase to serve as the baseline. We compare the latest learned policy $p_{\boldsymbol{\Phi}}(\mathcal{U} \,|\, \mathcal{S})$ with the baseline every N_e training iterations, where N_e is a positive integer. The baseline will be replaced by the latest policy $p_{\boldsymbol{\Phi}}(\mathcal{U} \,|\, \mathcal{S})$ if $p_{\boldsymbol{\Phi}}(\mathcal{U} \,|\, \mathcal{S})$ achieves a higher expected reward compared with the previous baseline. We refer to the N_e training iterations between two baseline updates as a *training episode*. With the gradient in (30), we use Adam optimizer [34] to solve problem (29) and update the learnable parameters $\boldsymbol{\Phi}$.

The training algorithm in each training episode is shown in Algorithm 1. In each training iteration, we train the DNN module using one minibatch, which includes B different channel realizations of the users. For each of the channel realizations in the minibatch, we determine the corresponding set $\mathcal{S}'$. We use set $\mathcal{S}_B$ to collect all the B vector sets in the

Algorithm 1 NCO-based User Scheduling Algorithm: Training Phase

1: **for** Training iteration counter $= 1, \ldots, N_e$ **do**
2: Obtain a minibatch consisting of B channel realizations, and determine the corresponding vector set $\mathcal{S}_B$.
3: **for** each $\mathcal{S}' \in \mathcal{S}_B$ **do**
4: Feed the vectors in $\mathcal{S}'$ into the DNN modules and determine the subset $\mathcal{U}$.
5: Determine the reward $r(\mathcal{U})$ by solving subproblem (12).
6: Given state $\mathcal{S}'$, determine the reward of the baseline policy $b(\mathcal{S}')$.
7: Determine the loss function as $\mathcal{L}(\mathbf{\Phi} \,|\, \mathcal{S}') \leftarrow b(\mathcal{S}') - r(\mathcal{U})$.
8: **end for**
9: Determine the aggregate loss over the minibatch as $\mathcal{L}(\mathbf{\Phi} \,|\, \mathcal{S}_B) \leftarrow \frac{1}{B} \sum_{\mathcal{S}' \in \mathcal{S}_B} \mathcal{L}(\mathbf{\Phi} \,|\, \mathcal{S}')$.
10: Determine the gradient based on (30), and update $\mathbf{\Phi}$ by solving problem (29) using Adam optimizer [34].
11: **end for**
12: Update the baseline policy if the current learned policy outperforms the previous baseline.
13: **return** The updated learnable parameters $\mathbf{\Phi}$.

Algorithm 2 Online Execution of the Proposed NCO-based User Scheduling Algorithm in Time Slot $t \in \mathcal{T}$

1: Determine the weights $w_n(t)$, $n \in \mathcal{N}$, based on (9).
2: Obtain vectors $\boldsymbol{v}_n(t)$, $n \in \mathcal{N}$ and state $\mathcal{S}(t)$ based on the channel information and the weights $w_n(t)$, $n \in \mathcal{N}$ in time slot t.
3: Feed the vectors in $\mathcal{S}(t)$ into the DNN modules, and determine the subset $\mathcal{U}(t)$ based on the learned policy $p_{\mathbf{\Phi}}(\mathcal{U}(t) \,|\, \mathcal{S}(t))$.
4: Obtain the corresponding user scheduling selection $\boldsymbol{x}^\star(t)$ in time slot t from $\mathcal{U}(t)$.

minibatch. That is, set $\mathcal{S}'$ is an element in set $\mathcal{S}_B$. The loss function is first determined for each $\mathcal{S}' \in \mathcal{S}_B$, and is then averaged over the whole minibatch. After each episode, the performance of the learned policy and the baseline policy is evaluated over B_e new channel realizations.

E. Online Algorithm

Algorithm 2 shows our proposed NCO-based algorithm, which determines the user scheduling given the online information in time slot $t \in \mathcal{T}$. In Line 1, we first determine the weights of all users in time slot t. In Line 2, we obtain state $\mathcal{S}(t)$ based on the channel information and the weights of the users. We then feed the vectors in $\mathcal{S}(t)$ into the pre-trained DNN modules, and determine the subset $\mathcal{U}(t)$ based on the learned policy $p_{\mathbf{\Phi}}(\mathcal{U}(t) \,|\, \mathcal{S}(t))$. With subset $\mathcal{U}(t)$, we can obtain the user scheduling vector $\boldsymbol{x}^\star(t)$. Now, with the given $\boldsymbol{x}^\star(t)$, we jointly optimize the phase shift matrix and beamforming vectors, and obtain $\mathbf{\Psi}^\star(t)$ and $\boldsymbol{b}^\star(t)$ using the CL-DDPG algorithm. The details of the CL-DDPG algorithm will be presented in the next section.

IV. DUPB Algorithm: CL-DDPG Algorithm for Phase Shift Control and Beamforming Optimization

Given the user scheduling vector, we propose a CL-DDPG algorithm to solve the joint phase shift control and beamforming optimization subproblem.

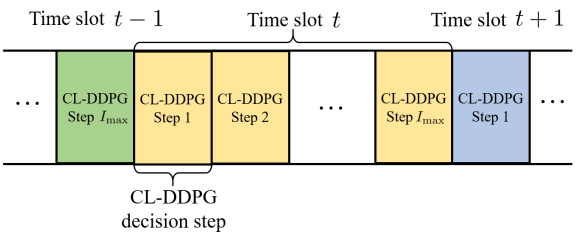


Fig. 4. Illustration of the sequential decision process of the CL-DDPG for joint phase shift control and beamforming optimization.

A. CL-DDPG Algorithm: State, Action, and Reward

Using our proposed CL-DDPG algorithm, the joint phase shift control and beamforming optimization in an arbitrary time slot $t \in \mathcal{T}$ can be formulated as a sequential decision process with a maximum of $I_{\max}$ decision steps, as shown in Fig. 4. Since the channel information and weights of the users are constant within a particular time slot, we drop the time index t for notation simplicity. We define the state in the i-th decision step as follows:

$$\boldsymbol{s}^{\text{PB}}(i) = [\boldsymbol{s}_1^{\text{PB}}(i), \boldsymbol{s}_2^{\text{PB}}(i), \ldots, \boldsymbol{s}_M^{\text{PB}}(i), \boldsymbol{g}\,], \quad i = 1, 2, \ldots, I_{\max}, \tag{31}$$

where $\boldsymbol{s}_n^{\text{PB}}(i) = [\boldsymbol{h}_{D,n},\, \boldsymbol{h}_{R,n},\, w_n R_n(\boldsymbol{a}^{\text{PB}}(i-1))]$, $n \in \mathcal{M}$, $\boldsymbol{a}^{\text{PB}}(i-1)$ is the action taken in the previous decision step, and $\boldsymbol{a}^{\text{PB}}(0)$ is the initialized action. The action vector is defined as

$$\boldsymbol{a}^{\text{PB}}(i) = [\boldsymbol{b}(i), \boldsymbol{\psi}(i)], \quad i = 1, 2, \ldots, I_{\max}, \tag{32}$$

where $\boldsymbol{\psi}(i) = (\psi_1(i), \ldots, \psi_{L_R}(i))$.

Remark: Since the elements in action vector $\boldsymbol{a}^{\text{PB}}(i)$ are continuous variables, we choose DDPG [17] as the foundation of our algorithm design. However, the vanilla DDPG suffers from the exploration inefficiency when the dimensionality of the state and action space is large. For the problem studied in this paper, as the dimensionality of the state and action space increases substantially with the number of reflecting elements L_R, the performance of the vanilla DDPG may degrade. To address these challenges and improve the efficiency of the vanilla DDPG, in our proposed CL-DDPG algorithm, curriculum learning is employed to design the reward function, such that the learning in the early stage is facilitated by exploiting the knowledge on the structure of the joint problem.

Curriculum learning [26] is a technique for modifying the system transition history observed by the learning agent such that the performance of learning algorithm can be improved. The rationale of curriculum learning is to begin the learning process with a simple objective and gradually change the objective towards a more difficult one, with the intuition that the knowledge learned with the simple objective can facilitate the learning with a more difficult objective [35], [36]. In our proposed CL-DDPG algorithm, during the early stage of the learning process, we choose an objective of reducing the *distance* between the suboptimal solution. We employ the following average distance measurement:

$$\frac{||\boldsymbol{a}^{\text{PB}}(i) - \boldsymbol{a}^{\text{PBSub}}||_2^2}{MK + L_R}, \tag{33}$$

where $\boldsymbol{a}^{\text{PBSub}} = \left[\boldsymbol{b}^{\text{sub}}, \boldsymbol{\psi}^{\text{sub}}\right]$ denotes the suboptimal solution of the joint problem (12), which can be obtained using an AO-based algorithm (e.g., [1], [11], [15]). The squared ℓ_2 norm is divided by $MK + L_R$ to obtain the average results for each element in $\boldsymbol{a}^{\text{PB}}(i)$.

Following the curriculum learning approach, we progressively change the objective towards the primal objective, which is to maximize the weighted aggregate throughput. To this end, the reward function in the proposed CL-DDPG algorithm is given by:

$$r^{\text{PB}}(i) = \sum_{n\in\mathcal{M}} w_n R_n(\boldsymbol{a}^{\text{PB}}(i)) - \beta(i)\frac{||\boldsymbol{a}^{\text{PB}}(i) - \boldsymbol{a}^{\text{PBSub}}||_2^2}{MK + L_R}, \tag{34}$$

where $\beta(i)$ is the weight representing the importance of minimizing the average distance measurement. The value $\beta(i)$ needs to be adjusted properly throughout the training phase, such that the second term in (34) can facilitate the learning and meanwhile the performance of the learning algorithm will not be limited by the suboptimal solution $\boldsymbol{a}^{\text{PBSub}}$. To achieve such goals in the proposed CL-DDPG algorithm, the value of $\beta(i)$ will be decayed by multiplying with $1-\theta$, where θ is a positive constant, if either of the following two conditions is satisfied:

- Condition 1: The distance measurement between the learned solution and the suboptimal solution is less than or equal to a given threshold ϵ. That is

$$\frac{||\boldsymbol{a}^{\text{PB}}(i) - \boldsymbol{a}^{\text{PBSub}}||_2^2}{MK + L_R} \leq \epsilon. \tag{35}$$

- Condition 2: The learned solution achieves a weighted aggregate throughput that is higher than that obtained from the suboptimal solution. That is

$$\sum_{n\in\mathcal{N}} w_n R_n(\boldsymbol{a}^{\text{PB}}(i)) > \sum_{n\in\mathcal{N}} w_n R_n(\boldsymbol{a}^{\text{PBSub}}). \tag{36}$$

If neither of the aforementioned two conditions is satisfied, we increment the value of β by multiplying with $1+\theta$ to offer a higher reward for approaching a suboptimal solution. Finally, we clip the value of $\beta(i)$ to be within $(0, \beta_{\max}]$ to avoid numerical issues. The overall mechanism for adjusting the value of β is shown in the following equation:

$$\beta(i+1) = \begin{cases} (1-\theta)\,\beta(i), & \text{if inequality (35) or} \\ & \text{(36) is satisfied,} \\ \min\{(1+\theta)\beta(i), \beta_{\max}\}, & \text{otherwise.} \end{cases} \tag{37}$$

B. Actor-Critic Method and Learning Algorithm

We employ the actor-critic method to learn the policy for solving the joint phase shift control and beamforming optimization problem. The actor learns a policy $\pi_{\boldsymbol{\Phi}_{\text{act}}}$, which is parameterized by the learnable parameters $\boldsymbol{\Phi}_{\text{act}}$. The policy $\pi_{\boldsymbol{\Phi}_{\text{act}}}$ defines a mapping from a state to an action. That is $\boldsymbol{a}^{\text{PB}}(i) = \pi_{\boldsymbol{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}}(i))$. In addition, the critic learns a state-action value function $Q_{\boldsymbol{\Phi}_{\text{crt}}}$, which is parameterized by $\boldsymbol{\Phi}_{\text{crt}}$. The state-action value function $Q_{\boldsymbol{\Phi}_{\text{crt}}}(\boldsymbol{s}^{\text{PB}}(i), \boldsymbol{a}^{\text{PB}}(i))$ estimates the discounted future reward of selecting action $\boldsymbol{a}^{\text{PB}}(i)$ under state $\boldsymbol{s}^{\text{PB}}(i)$. That is,

$$\begin{aligned} &Q_{\boldsymbol{\Phi}_{\text{crt}}}(\boldsymbol{s}^{\text{PB}}(i), \boldsymbol{a}^{\text{PB}}(i)) \\ &= \mathbb{E}_{\boldsymbol{s}^{\text{PB}}\sim p_{\pi_{\boldsymbol{\Phi}_{\text{act}}}}, \boldsymbol{a}^{\text{PB}}\sim\pi_{\boldsymbol{\Phi}_{\text{act}}}}\left[\sum_{\ell=i}^{T_{\max}} \gamma^{\ell-i} r^{\text{PB}}(\boldsymbol{s}^{\text{PB}}(\ell), \boldsymbol{a}^{\text{PB}}(\ell))\right], \end{aligned} \tag{38}$$

where $p_{\pi_{\boldsymbol{\Phi}_{\text{act}}}}$ denotes the discounted state visitation distribution which depends on the actor's policy $\pi_{\boldsymbol{\Phi}_{\text{act}}}$, $T_{\max}$ is the decision horizon, and $\gamma \in [0,1]$ is the discount factor.

The goal of the actor-critic method is to learn the actor policy which maximizes the discounted future reward [17]. We have

$$\begin{aligned} &\underset{\boldsymbol{\Phi}_{\text{act}}}{\text{maximize}}\ J(\boldsymbol{\Phi}_{\text{act}}) \\ &\triangleq \mathbb{E}_{\boldsymbol{s}^{\text{PB}}\sim p_{\pi_{\boldsymbol{\Phi}_{\text{act}}}}, \boldsymbol{a}^{\text{PB}}\sim\pi_{\boldsymbol{\Phi}_{\text{act}}}}\left[\sum_{\ell=1}^{T_{\max}} \gamma^{\ell-1} r^{\text{PB}}(\boldsymbol{s}^{\text{PB}}(\ell), \boldsymbol{a}^{\text{PB}}(\ell))\right]. \end{aligned} \tag{39}$$

To solve problem (39), we use the deterministic policy gradient algorithm [37] to update the learnable parameters of the actor $\boldsymbol{\Phi}_{\text{act}}$ with the following gradient:

$$\begin{aligned} &\nabla J(\boldsymbol{\Phi}_{\text{act}}) \\ &= \mathbb{E}_{\boldsymbol{s}^{\text{PB}}\sim p_{\pi_{\boldsymbol{\Phi}_{\text{act}}}}}\left[\nabla Q_{\boldsymbol{\Phi}_{\text{crt}}}(\boldsymbol{s}^{\text{PB}}, \boldsymbol{a}^{\text{PB}})\,|_{\,\boldsymbol{a}^{\text{PB}}=\pi_{\boldsymbol{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}})} \nabla\pi_{\boldsymbol{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}})\right]. \end{aligned} \tag{40}$$

Moreover, the learnable parameters of the critic $\boldsymbol{\Phi}_{\text{crt}}$ is updated based on the temporal difference (TD) error of value approximation [38, Ch. 6]. In particular, we first determine the target of the state-action value function approximation as

$$\begin{aligned} y(\boldsymbol{s}^{\text{PB}}, \pi_{\boldsymbol{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}})) = r^{\text{PB}}(\boldsymbol{s}^{\text{PB}}, \pi_{\boldsymbol{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}})) \\ + \gamma\, Q_{\boldsymbol{\Phi}_{\text{crt}}}(\hat{\boldsymbol{s}}^{\text{PB}}, \pi_{\boldsymbol{\Phi}_{\text{act}}}(\hat{\boldsymbol{s}}^{\text{PB}})), \end{aligned} \tag{41}$$

where $\hat{\boldsymbol{s}}^{\text{PB}}$ is the next state as a result of taking action $\pi_{\boldsymbol{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}})$ in state $\boldsymbol{s}^{\text{PB}}$. Then, $\boldsymbol{\Phi}_{\text{crt}}$ is updated by solving the following minimization problem [38, Ch. 6]:

$$\begin{aligned} &\underset{\boldsymbol{\Phi}_{\text{crt}}}{\text{minimize}}\ \mathcal{L}(\boldsymbol{\Phi}_{\text{crt}}) \\ &\triangleq \mathbb{E}_{\boldsymbol{s}^{\text{PB}}\sim p_{\pi_{\boldsymbol{\Phi}_{\text{act}}}}, \boldsymbol{a}^{\text{PB}}\sim\pi_{\boldsymbol{\Phi}_{\text{act}}}}\Big[\big(Q_{\boldsymbol{\Phi}_{\text{crt}}}(\boldsymbol{s}^{\text{PB}}, \pi_{\boldsymbol{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}})) \\ &\qquad - y(\boldsymbol{s}^{\text{PB}}, \pi_{\boldsymbol{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}}))\big)^2\Big]. \end{aligned} \tag{42}$$

We can update $\boldsymbol{\Phi}_{\text{crt}}$ using gradient descent with the following gradient:

$$\begin{aligned} &\nabla\mathcal{L}(\boldsymbol{\Phi}_{\text{crt}}) \\ &= \mathbb{E}_{\boldsymbol{s}^{\text{PB}}\sim p_{\pi_{\boldsymbol{\Phi}_{\text{act}}}}, \boldsymbol{a}^{\text{PB}}\sim\pi_{\boldsymbol{\Phi}_{\text{act}}}}\Big[2\nabla Q_{\boldsymbol{\Phi}_{\text{crt}}}(\boldsymbol{s}^{\text{PB}}, \pi_{\boldsymbol{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}})) \\ &\qquad \big(Q_{\boldsymbol{\Phi}_{\text{crt}}}(\boldsymbol{s}^{\text{PB}}, \pi_{\boldsymbol{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}})) - y(\boldsymbol{s}^{\text{PB}}, \pi_{\boldsymbol{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}}))\big)\Big]. \end{aligned} \tag{43}$$

To overcome the overestimation issue [39] of the conventional actor-critic method, we introduce two critics, where each of them performs an independent estimation of the state-action value function. We denote the learnable parameters of the two critics as $\boldsymbol{\Phi}_{\text{crt1}}$ and $\boldsymbol{\Phi}_{\text{crt2}}$, respectively. We

Algorithm 3 CL-DDPG: Training Algorithm

1: Set episode counter $k \leftarrow 0$.
2: Initialize the threshold δ.
3: Initialize the learnable parameters $\mathbf{\Phi}_{\text{act}}$, $\mathbf{\Phi}_{\text{crt1}}$, and $\mathbf{\Phi}_{\text{crt2}}$.
4: Conduct $T_{\text{warm-up}}$ episodes of exploration and store the transition history.
5: **while** $k \leq T_{\max}$ **do**
6: Observe new channel realizations and the weights of the users.
7: Initialize $i \leftarrow 0$ and $\beta(i) \leftarrow \beta_0$.
8: Determine the suboptimal solution as $\boldsymbol{a}^{\text{PBSub}}$ using a baseline algorithm.
9: **while** $i \leq I_{\max}$ **do**
10: Determine the action $\boldsymbol{a}^{\text{PB}}(i) \leftarrow \pi_{\mathbf{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}}(i)) + \boldsymbol{\varrho}_{\text{epl}}$.
11: Determine the reward $r^{\text{PB}}(i)$ based on (34).
12: Update $\beta(i)$ based on (37) and store the transition history.
13: Sample a mini-batch of $\widehat{B}$ transition tuples.
14: Update $\mathbf{\Phi}_{\text{act}}$, $\mathbf{\Phi}_{\text{crt1}}$, and $\mathbf{\Phi}_{\text{crt2}}$ using the Adam optimizer and (44).
15: $i \leftarrow i + 1$.
16: **end while**
17: $k \leftarrow k + 1$
18: **end while**

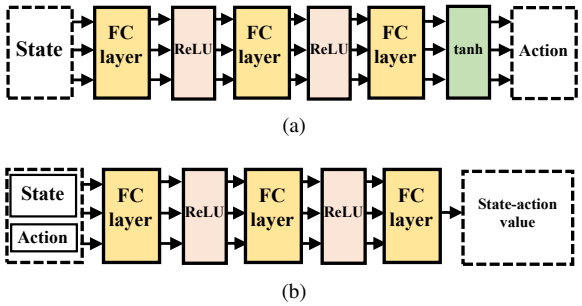


Fig. 5. The network structure of (a) the actor and (b) the critic. The actor network determines the action based on the state. The critic network approximates the state-action value given the state and action.

upper-bound the approximated state-action value function by the minimum approximation of the two critics to alleviate the impact of overestimation. This results in the following modification of the target of the expected discounted reward approximation in (41):

$$y(\boldsymbol{s}^{\text{PB}}, \pi_{\mathbf{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}})) = R(\boldsymbol{s}^{\text{PB}}, \pi_{\mathbf{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}})) + \gamma \min\Big\{Q_{\mathbf{\Phi}_{\text{crt1}}}(\hat{\boldsymbol{s}}^{\text{PB}}, \pi_{\mathbf{\Phi}_{\text{act}}}(\hat{\boldsymbol{s}}^{\text{PB}})), Q_{\mathbf{\Phi}_{\text{crt2}}}(\hat{\boldsymbol{s}}^{\text{PB}}, \pi_{\mathbf{\Phi}_{\text{act}}}(\hat{\boldsymbol{s}}^{\text{PB}}))\Big\}.$$

The overall learning algorithm is shown in Algorithm 3. The learning is performed in an episodic fashion. The learning progress consists of $T_{\max}$ episodes. Each episode has $I_{\max}$ decision steps. The same channel realization is used within an episode, while different (and independent) channel realizations are used across different episodes. To improve the exploration of the action space, we introduce a warm-up stage that consists of $T_{\text{warm-up}}$ episodes. Within the warm-up stage, the actions are chosen by sampling from a uniform distribution over the feasible actions. New channel realization and the weights of the users are observed at the beginning of each episode. In Line 8, we determine a suboptimal solution of the joint problem using a baseline algorithm, e.g., the AO-based algorithms [1], [11], [15]. In the i-th decision step, the action is determined as $\boldsymbol{a}^{\text{PB}}(i) = \pi_{\mathbf{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}}(i)) + \boldsymbol{\varrho}_{\text{epl}}$, where $\boldsymbol{\varrho}_{\text{epl}}$ is the Gaussian noise with zero mean and variance σ^2_{epl}. In Lines 10−12, after selecting an action, we update the value of $\beta(i)$ based on (37). The transition tuple $(\boldsymbol{s}^{\text{PB}}(i), \boldsymbol{a}^{\text{PB}}(i), r^{\text{PB}}(i), \boldsymbol{s}^{\text{PB}}(i+1))$ is stored in the experience replay. In Line 13, we sample a mini-batch of $\widehat{B}$ transition tuples from the experience replay. After determining the gradient for each transition tuple within the mini-batch, we obtain the new learnable parameters $\mathbf{\Phi}'_{\text{act}}$, $\mathbf{\Phi}'_{\text{crt1}}$, and $\mathbf{\Phi}'_{\text{crt2}}$ by performing one step of gradient descent (and ascent) using the Adam optimizer. We then update the learnable parameters using the following soft update:

$$\begin{aligned}\mathbf{\Phi}_{\text{act}} &= \kappa\mathbf{\Phi}'_{\text{act}} + (1-\kappa)\mathbf{\Phi}_{\text{act}},\\ \mathbf{\Phi}_{\text{crt1}} &= \kappa\mathbf{\Phi}'_{\text{crt1}} + (1-\kappa)\mathbf{\Phi}_{\text{crt1}},\\ \mathbf{\Phi}_{\text{crt2}} &= \kappa\mathbf{\Phi}'_{\text{crt2}} + (1-\kappa)\mathbf{\Phi}_{\text{crt2}},\end{aligned} \tag{44}$$

where κ is a constant and is between zero and one.

C. Network Structure of the Actor and Critic

The proposed actor network structure is shown in Fig. 5(a). The actor network consists of three FC layers. The first and second FC layers are followed by the ReLU activation layers. The output of the last FC layer is passed through a `tanh` activation layer to ensure that each of the output value is always within the interval $(-1, 1)$. We note that setting a minimum and a maximum value for the action facilitates the learning of both the actor and critic networks. For the output of the actor network that corresponds to the phase shift variables, we multiply the output value by $(1+\lambda)\pi$, where $\lambda > 0$, to obtain the phase shift value[2]. The sizes of the input and output of the actor network are given by $d_{\text{act\_in}} = M(K + L_R + KL_R + 1)$ and $d_{\text{act\_out}} = MK + L_R$, respectively. The sizes of the three FC layers in the actor network are $d_{\text{act\_in}} \times d_{\text{fc1}}$, $d_{\text{fc1}} \times d_{\text{fc2}}$, and $d_{\text{fc2}} \times d_{\text{act\_out}}$, respectively. The remaining outputs are directly used as the real and imaginary parts of the beamforming vectors. This is because the magnitude of the beamforming vector at the receiver does not affect the optimality of the solution. An arbitrary beamforming vector $\boldsymbol{b}_n$ can be scaled as $\frac{\boldsymbol{b}_n}{(1+\lambda)||\boldsymbol{b}_n||_2}$ so that the real and imaginary parts of each element of the vector are within the interval $(-1, 1)$. Both critics use the network structure as shown in Fig. 5(b). Each critic network has three FC layers and two ReLU activation layers. The input of the critic network is the concatenation of the state and action. The size of the input of the critic networks is given by $d_{\text{crt\_in}} = M(K + L_R + KL_R + 1) + MK + L_R$. Given the state and action, the output of the critic networks is $Q_{\mathbf{\Phi}_{\text{crt}}}(\boldsymbol{s}^{\text{PB}}, \pi_{\mathbf{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}}))$. The sizes of the three FC layers in the critic networks are $d_{\text{crt\_in}} \times d_{\text{fc1}}$, $d_{\text{fc1}} \times d_{\text{fc2}}$, and $d_{\text{fc2}} \times 1$, respectively.

[2]We choose the multiplier $(1+\lambda)\pi > \pi$ since the output value of the `tanh` activation layer cannot be equal to either 1 or -1. Using a multiplier that is less than or equal to π will affect the optimality of the learned solution when the optimal solution is either $\psi_l^\star = \pi$ or $\psi_l^\star = -\pi$.

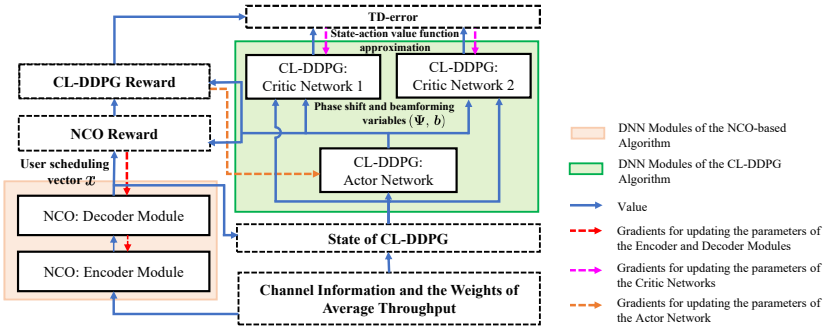


Fig. 6. The overall architecture of the proposed DUPB algorithm. The proposed DUPB algorithm consists of (i) the encoder and decoder modules employed in the NCO-based user scheduling algorithm as indicated by the orange box in the figure, and (ii) the actor and critic networks, i.e., the green box in the figure, for determining the phase shift and beamforming variables.

Algorithm 4 CL-DDPG: Online Execution Algorithm

1: Observe the channel realizations and the weights of the scheduled users.
2: Determine the suboptimal solution $\boldsymbol{a}^{\text{PBSub}}$ using a baseline algorithm.
3: Initialize the threshold δ.
4: Initialize $i \leftarrow 0$ and $\beta(i) \leftarrow \beta_0$.
5: Initialize the maximum achievable weighted aggregate throughput $R_{\max} \leftarrow 0$.
6: Conduct $I_{\text{warm-up}}$ decision steps of exploration and store the transition history.
7: **while** $i \leq I_{\max}$ **do**
8: Determine the action $\boldsymbol{a}^{\text{PB}}(i) \leftarrow \pi_{\boldsymbol{\Phi}_{\text{act}}}(\boldsymbol{s}^{\text{PB}}(i)) + \boldsymbol{\varrho}_{\text{epl}}$.
9: Determine the reward $r^{\text{PB}}(i)$ based on (34).
10: Update $\beta(i)$ based on (37) and store the transition history.
11: Sample a mini-batch and update $\boldsymbol{\Phi}_{\text{act}}$, $\boldsymbol{\Phi}_{\text{crt1}}$, and $\boldsymbol{\Phi}_{\text{crt2}}$ using the Adam optimizer and (44).
12: **if** $\sum_{n\in\mathcal{M}} w_n(t) R_n(\boldsymbol{a}^{\text{PB}}(i)) \geq R_{\max}$ **then**
13: $\boldsymbol{a}^{\text{out}} \leftarrow \boldsymbol{a}^{\text{PB}}(i)$.
14: $R_{\max} \leftarrow \sum_{n\in\mathcal{M}} w_n(t) R_n(\boldsymbol{a}^{\text{PB}}(i))$.
15: **end if**
16: $i \leftarrow i + 1$.
17: **end while**
18: Return $\boldsymbol{a}^{\text{out}}$.

D. CL-DDPG: Online Algorithm

The online algorithm of CL-DDPG algorithm is shown in Algorithm 4. In Line 2, given the channel information and the weights of the scheduled users, we obtain the suboptimal solution $\boldsymbol{a}^{\text{PBSub}}$ using a baseline algorithm. In Line 6, we conduct $I_{\text{warm-up}}$ decision steps of exploration to collect system transition tuples. In Lines 7−17, we update the learnable parameters for $I_{\max}$ decision steps based on the online information to further improve the learned policy. In Lines 12−15, the action that achieves the maximum weighted aggregate throughput throughout the $I_{\max}$ decision steps of the online execution is chosen as the output.

E. Overall Framework of the Proposed DUPB Algorithm

The overall framework of the proposed DUPB algorithm is shown in Fig. 6. The forward propagations for determining the rewards, state-action value function approximation, and TD-error are denoted by blue solid arrows. The backpropagations of the gradients are denoted by dashed arrows. Note that, both the reward functions used in the NCO-based and CL-DDPG algorithms depend on the weighted aggregate throughput, which corresponds to the objective of the primal problem (10). With the reward function in (16), we use the REINFORCE algorithm to determine the gradient for updating the learnable parameters of the encoder and decoder modules in the NCO-based algorithm. Recall that for the reward function in (16), the solutions of phase shift control $\boldsymbol{\Psi}^\star$ and beamforming vectors $\boldsymbol{b}_n^\star$, $n \in \mathcal{N}$, are determined by the phase shift control and beamforming policy $\pi_{\boldsymbol{\Phi}_{\text{act}}}$ that is learned by the CL-DDPG algorithm. Therefore, $b(\mathcal{S})$ and $r(\mathcal{U}')$ in (30) depend on policy $\pi_{\boldsymbol{\Phi}_{\text{act}}}$. This indicates that the gradient for updating the user scheduling policy depends on the phase shift control and beamforming policy $\pi_{\boldsymbol{\Phi}_{\text{act}}}$. Thus, the learning of user scheduling policy depends on $\pi_{\boldsymbol{\Phi}_{\text{act}}}$. We then use the deterministic policy gradient and TD-error to obtain the gradients for the training of the actor and critic networks in the CL-DDPG algorithm, respectively.

For the proposed DUPB algorithm, the computational complexity of obtaining the user scheduling vector depends on the computational complexity of forward propagation through the encoder and decoder modules. The forward propagation through the initial embedding module in the encoder module has a complexity of $\mathcal{O}\Big(N\big((K + L_R + KL_R + 1)d_{\text{Res}} + N_{\text{Res}} d_{\text{Res}}^2 + d_{\text{Res}} d_h\big)\Big)$. The forward propagation through the attention layers, MLP layers, and the embedding aggregate layer incurs a complexity of $\mathcal{O}(Nd_z d_h + Nd_k d_h + N^2 d_k + Nd_h^2 + Nd_a d_h + d_c d_h + Nd_h)$. Hence, obtaining the user scheduling incurs the following computational complexity:

$$\begin{aligned}\mathcal{O}_{\text{US}} = \mathcal{O}\Big(N\big((K + L_R + KL_R + 1)d_{\text{Res}} \\ + N_{\text{Res}} d_{\text{Res}}^2 + d_{\text{Res}} d_h\big) + Nd_z d_h + Nd_k d_h \\ + N^2 d_k + Nd_h^2 + Nd_a d_h + d_c d_h + Nd_h\Big).\end{aligned}$$

For the ease of analysis, given the M scheduled users, we assume that the AO-based algorithm with FP is employed to obtain a suboptimal solution of the joint phase shift and beamforming optimization problem. This incurs a computational complexity of $\mathcal{O}(C_{\text{AO}}(C_{\text{PS}} L_R^{4.5} \log(1/\epsilon_{\text{SDR}}) + MK^3))$, where C_{AO} is the number of iterations to solve the joint phase shift and beamforming optimization problem with fixed user

scheduling vector, C_{PS} is the number of iterations to solve the phase shift subproblem using FP and SDR, and ϵ_{SDR} is the solution accuracy of SDR. With the obtained suboptimal solution, the online execution of the CL-DDPG algorithm incurs $\widehat{B}I_{\max}$ forward propagations and $I_{\max}$ backpropagations. Each forward propagation has a computational complexity of $\mathcal{O}_{\text{PB\_forward}} = \mathcal{O}\Big((d_{\text{act\_in}} + d_{\text{crt\_in}})d_{\text{fc1}} + d_{\text{fc1}}d_{\text{fc2}} + (d_{\text{act\_out}} + 1)d_{\text{fc2}}\Big)$, while each backpropagation incurs a computational complexity of

$$\mathcal{O}_{\text{PB\_backprop}} = \mathcal{O}\Big((d_{\text{act\_in}} + d_{\text{crt\_in}} + d_{\text{fc2}} + 1)d_{\text{fc1}} + d_{\text{fc2}} + (d_{\text{act\_out}} + 1)d_{\text{fc2}} + d_{\text{act\_out}}\Big). \tag{45}$$

The overall computational complexity of the proposed DUPB algorithm is given by:

$$\mathcal{O}_{\text{US}} + \mathcal{O}(C_{\text{AO}}(C_{\text{PS}}L_R^{4.5}\log(1/\epsilon_{\text{SDR}}) + MK^3)) + \widehat{B}I_{\max}\mathcal{O}_{\text{PB\_forward}} + I_{\max}\mathcal{O}_{\text{PB\_backprop}}. \tag{46}$$

V. Performance Evaluation

We simulate an IRS-aided system where the distance between the IRS and the base station is 200 meters. The users are randomly and uniformly distributed within [10, 50] meters of the IRS. The simulation setup is shown in Fig. 7. We assume the direct channels between the users and the base station are blocked [8]. Note that when the direct channels are present, the input dimensionality of the linear projection layers in the encoder module should be increased accordingly to incorporate the extra channel information. The reflecting channels follow Rician fading distribution. We set the Ricean K-factor to 6 with path loss exponent equal to 3.5 [40]. The maximum transmit power $P^{\max}$ is set to 30 dBm, and the noise power is set to -90 dBm. The window size of the exponential moving average T_C in equation (9) is set to 20. The parameters for the proposed DUPB algorithm are shown in Table I. We use PyTorch [41] for simulation. In our simulation, the channel realizations for DNN training and evaluation are generated independently. We conduct the simulation using a Linux-based computing server with an Intel E5-2683 Broadwell @ 2.1GHz CPU, and an NVIDIA Tesla P100 Pascal GPU with 12 GB memory. The average training time per episode of the proposed DUPB algorithm on the computing server is 132.7 min for the system setting $N = 10$, $M = K = 4$, and $L_R = 80$.

The performance of the proposed DUPB algorithm is evaluated after 200 training episodes. We first compare the performance of the proposed DUPB algorithm with the following three algorithms:

- AO-based algorithm with FP [1]: In this algorithm, we first relax the binary user scheduling variables, and then solve the user scheduling, phase shift control, and beamforming subproblems iteratively using FP and SDR.
- AO-based algorithm with SCA [11]: Compared with the AO-based algorithm with FP, in this algorithm, we solve the phase shift control subproblem using the SCA approach [11].

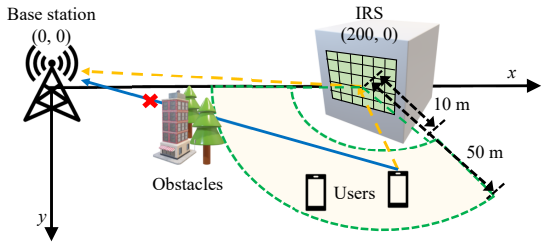


Fig. 7. Simulation setup. The base station is located at $(0, 0)$, while the IRS is located at $(200, 0)$. The users are distributed within a 120° annulus sector where the IRS is located as the center of the circle.

TABLE I
Simulation Parameters for the Proposed DUPB Algorithm

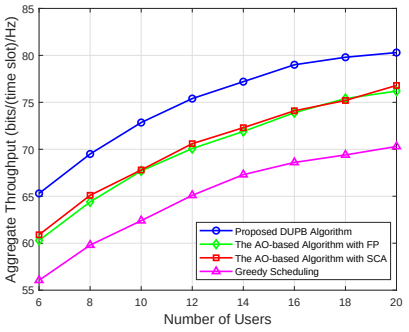
Parameter	Value
Dimensionalities of the DNN layers in NCO d_k, d_z, d_a, d_h	$\max\{256, 4d_i\}$
Number of training iterations in each episode of NCO N_e	25
Dimensionalities of the DNN layers in CL-DDPG d_{fc1}, d_{fc2}	$\max\{256, 2d_{\text{crt\_in}}\}$
Maximum number of decision steps in CL-DDPG $I_{\max}$	1000
Learning rate	0.005
Minibatch sizes B, $\widehat{B}$	64, 128
Constant α for clipping the compatibility	10
Number of channel realizations used for evaluation B_e	5000
Reward function parameters β_0, $\beta_{\max}$, ϵ	10, 10, 0.1
Results scaling factors θ, λ	0.01, 0.01
Number of episodes and decision steps for warm-up $T_{\text{warm-up}}$, $I_{\text{warm-up}}$	10, 1000

- Greedy scheduling: In this algorithm, the base station schedules those M users with the largest value of $\left|\boldsymbol{h}_{D,n}(t) + \boldsymbol{G}^H(t)\boldsymbol{h}_{R,n}(t)\right|^2$, $n \in \mathcal{N}$ and then employs the AO-based algorithm with FP to obtain the phase shift matrix and beamforming vectors.

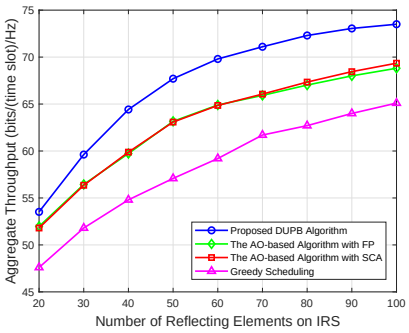
A. Aggregate Throughput

We vary the number of users and evaluate the aggregate throughput with the maximum aggregate throughput objective in Fig. 8(a). We set $M = K = 4$ and $L_R = 80$. We observe that the aggregate throughput of all considered algorithms increase with the number of users. When N is equal to 20, the proposed DUPB algorithm achieves an aggregate throughput which is 4.6%, 5.4% and 14.2% higher than the AO-based algorithm with SCA, the AO-based algorithm with FP, and the greedy scheduling algorithm, respectively.

Fig. 8(b) shows the aggregate throughput versus the number of reflecting elements on the IRS with the maximum aggregate throughput objective. We set $M = K = 4$ and $N = 10$. The results show that the aggregate throughput of all considered algorithms increase with the number of reflecting elements. The reason is that, having more reflecting elements on the IRS offers a higher degree of freedom for controlling the phase shift of the reflecting channel. By properly controlling the phase shift of the IRS and beamforming at the base station, we can reap the benefits of the degree of freedom.

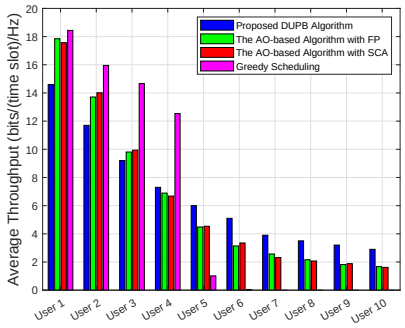


(a)

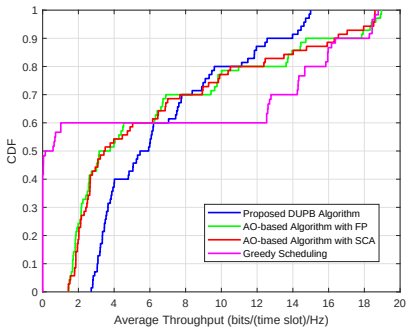


(b)

Fig. 8. Aggregate throughput versus (a) the number of users and (b) the number of reflecting elements when the objective is to maximize the aggregate throughput.



(a)



(b)

Fig. 9. (a) Average throughput per user and (b) the cumulative distribution function (CDF) of the average throughput per user for $M = K = 4$, $N = 10$, and $L_R = 80$.

When L_R is equal to 100, the proposed DUPB algorithm achieves an aggregate throughput that is 6.0%, 7.3%, and 12.9% higher than the AO-based algorithm with SCA, the AO-based algorithm with FP, and the greedy scheduling algorithm, respectively.

B. Fairness Among the Users

We evaluate the average throughput per user under the proportional fairness objective in Fig. 9(a). We sort the users in an ascending order of their distances to the base station and index them accordingly. That is, the closest user to the base station is referred to as user 1, and the farthest user to the base station is referred to as user 10. All algorithms are evaluated after running 300 consecutive time slots. With proportional fairness objective, the users are allocated with appropriate amount of network resources to maintain a balance between fairness and aggregate throughput. This is because the reward of scheduling a particular user is weighted based on its average throughput. Scheduling the users with low average throughput can sometimes lead to a higher reward. Hence, the network resources will not congregate at those users with better channel condition. We also observe from Fig. 9(a) that the distribution of throughput per user varies between different algorithms. With the greedy scheduling algorithm, the users with good channel conditions (e.g., users 1 and 2) obtain a higher throughput, while the users with poor channel conditions (e.g., users 9 and 10) are not scheduled by the base station. In the proposed DUPB algorithm, those users with poor channel conditions are scheduled more frequently and hence obtain a higher throughput.

In Fig. 9(b), we plot the cumulative distribution function (CDF) of the average throughput of the users. We observe that, for those users with relatively low average throughput, particularly lower than 4 bits/(time slot)/Hz, they are allocated with more resources under the proposed DUPB algorithm and hence achieve a higher average throughput. The results also show the difference in the tail distribution of the average throughput per user. With the proportional fairness objective, the users with preferable channel conditions may achieve the highest average throughput under the greedy scheduling algorithm. This is consistent with the results in Fig. 9(a). Moreover, the proposed DUPB algorithm reduces the standard deviation of the throughput distribution among the users by 29.7%, 33.4%, and 51.1% when compared with the AO-based algorithm with SCA, the AO-based algorithm with FP, and the greedy scheduling algorithm, respectively.

C. Runtime Comparison

We compare the online execution runtime of different algorithms for 10 consecutive time slots on the same computing server. The results are shown in Table II. When L_R is equal to 100, the runtime of the proposed DUPB algorithm is 37.9% and 43.5% lower than the AO-based algorithm with SCA and the AO-based algorithm with FP, respectively. We also observe that the AO-based algorithm with FP incurs the longest runtime when L_R is large, mainly due to the

TABLE II
ONLINE EXECUTION RUNTIME COMPARISON FOR DIFFERENT ALGORITHMS

Parameter Settings	$N = 10$, $M = K = 4$, $L_R = 40$	$N = 10$, $M = K = 4$, $L_R = 80$	$N = 10$, $M = K = 4$, $L_R = 100$	$N = 20$, $M = K = 4$, $L_R = 80$
Proposed DUPB Algorithm	79.4 min	116.9 min	159.6 min	124.7 min
The AO-based Algorithm with FP	68.5 min	175.6 min	282.7 min	332.8 min
The AO-based Algorithm with SCA	81.2 min	168.7 min	256.8 min	320.6 min
Greedy Scheduling Algorithm	27.6 min	56.9 min	84.5 min	57.1 min

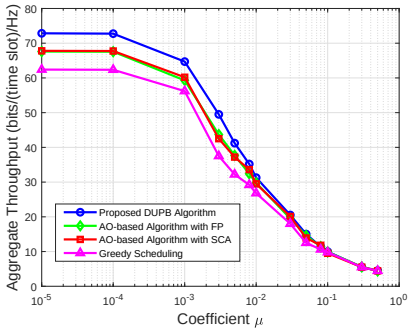


Fig. 10. Aggregate throughput under imperfect channel estimation. We set $M = K = 4$, $L_R = 80$, and $N = 10$.

computational complexity of using SDR to obtain the rank-one matrix. In all considered settings, the greedy scheduling algorithm requires the shortest runtime as the user scheduling is determined by a pre-defined heuristic.

D. Effect of Imperfect Channel Estimation

We evaluate the impact of imperfect channel estimation on the performance of the proposed algorithms. The estimated channel gain between the base station and the IRS is given by $\widehat{\boldsymbol{G}}(t) = \boldsymbol{G}(t) + \Delta\boldsymbol{G}(t)$, where $\Delta\boldsymbol{G}(t) \in \mathbb{C}^{L_R \times K}$ is channel estimation error of $\boldsymbol{G}(t)$. We consider the statistical channel estimation error [42], [43], where the elements in $\Delta\boldsymbol{G}(t)$ are assumed to follow complex Gaussian distribution with zero mean and variance $\mu^2\|\boldsymbol{g}(t)\|_2^2$. The constant $\mu \in [0, 1)$ measures the significance of estimation error. The estimated channel between user n and the IRS is given by $\widehat{\boldsymbol{h}}_{R,n}(t) = \boldsymbol{h}_{R,n}(t) + \Delta\boldsymbol{h}_{R,n}(t)$. The elements in $\Delta\boldsymbol{h}_{R,n}(t)$ follow complex Gaussian distribution with zero mean and variance $\mu^2\|\boldsymbol{h}_{R,n}(t)\|_2^2$. In Fig. 10, we evaluate the aggregate throughput under imperfect channel estimation with the maximum aggregate throughput as objective. We observe performance degradations in all considered algorithms due to imperfect channel estimation. When μ is equal to 0.005, the proposed DUPB algorithm can retain 56.6% of the aggregate throughput under perfect channel estimation. The performance degradation of the proposed DUPB algorithm and the other baseline algorithms due to imperfect channel estimation are similar.

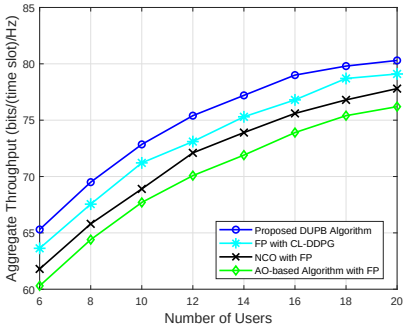
E. Performance Gain of Each Module

In order to provide additional insights on the throughput improvement obtained from the proposed DUPB algorithm, we evaluate the performance of the following two algorithms:

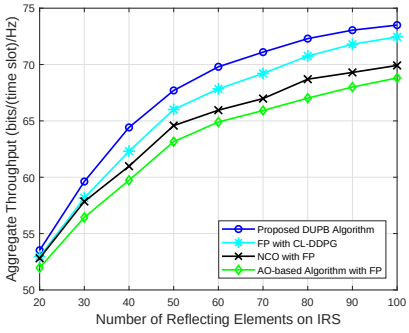
1) NCO-based user scheduling with FP-based phase shift control and beamforming optimization: In this algorithm, we use the proposed NCO-based algorithm to learn the user scheduling policy, while the joint phase shift control and beamforming subproblem is solved using the FP and SDR-based approach. We refer to this algorithm as NCO with FP.

2) FP-based user scheduling with CL-DDPG-based phase shift control and beamforming optimization: In this algorithm, we use the FP-based approach to solve the user scheduling subproblem. Given the user scheduling vector, the joint phase shift control and beamforming optimization subproblem is solved using the proposed CL-DDPG algorithm. We refer to this algorithm as FP with CL-DDPG.

We set $M = K = 4$ and $L_R = 80$. The performance comparison results are shown in Fig. 11. Results show that both the FP with CL-DDPG and NCO with FP algorithms achieve better performance than the AO-based algorithm. This indicates that both modules (i.e., the NCO algorithm for



(a)



(b)

Fig. 11. Aggregate throughput versus (a) the number of users and (b) the number of reflecting elements when the objective is to maximize the aggregate throughput.

user scheduling and the CL-DDPG algorithm for phase shift control and beamforming optimization) contribute to the improved performance of our proposed DUPB algorithm. While the optimization-based approaches can obtain high quality suboptimal solutions, the following issues may still affect the optimality of these approaches due to the nonconvexity of the studied problem. First, the quality of the converged solution of the FP-based user scheduling algorithm can be highly sensitive to the initialization of the relaxed user scheduling variables. When the relaxed user scheduling variable of user n (i.e., $x_n \in [0, 1]$) is initialized to be close to 0, the local gradient will encourage the algorithm to further reduce the value of x_n. Therefore, the FP-based user scheduling algorithm would converge to the stationary point in which user n is not being scheduled, i.e., $x_n = 0$. Thus, it can difficult for the FP-based user scheduling algorithm to avoid being trapped by the local optimal solutions. When the user scheduling subproblem is solved using the NCO algorithm, we can explore the solution space better than the FP-based algorithm since (a) a large number of feasible actions are being explored by the NCO algorithm during the exploration phase and the training phase, (b) the NCO algorithm learns a stochastic user scheduling policy which tends to explore more feasible actions during the early phase of training, and (c) with the help of the REINFORCE algorithm, the policy learning in the NCO algorithm is designed to find the best solution within the explored solution space. The aforementioned features allow the NCO algorithm to efficiently explore the solution space and obtain a higher aggregate throughput.

We also observe from Fig. 11 that, when compared with the AO-based algorithm with FP, the CL-DDPG algorithm provides a higher performance gain than the NCO-based user scheduling algorithm. When the joint phase shift control and beamforming subproblem is solved using the AO-based algorithm with FP, since Gaussian randomization process is used in the SDR to recover the rank-one phase shift control matrix, it may lead to performance degradation. In our proposed CL-DDPG algorithm, since we can determine the phase shift variables that satisfy the unit-modulus constraint directly from the learned policy, the performance degradation due to Gaussian randomization can be mitigated.

VI. Conclusion

In this paper, we studied the joint user scheduling, phase shift control, and beamforming design in IRS-aided systems. We decomposed the formulated problem into two subproblems: a combinatorial optimization subproblem for user scheduling, and a nonconvex subproblem for joint phase shift control and beamforming optimization. We proposed a DUPB algorithm, in which NCO technique is used to learn the stochastic policy for determining the user scheduling. In addition, we proposed a CL-DDPG algorithm to jointly optimize the phase shift control and the beamforming vectors. Our curriculum learning approach facilitates the policy learning by exploiting the knowledge on the hidden convexity of the joint problem. Results showed that the proposed DUPB algorithm outperforms the AO-based algorithms and greedy scheduling algorithm under both maximum aggregate throughput and proportional fair objectives. The runtime of the proposed DUPB algorithm is lower than that of the proposed AO-based algorithms when the number of reflecting elements is large. For future work, we will consider the multicell scenario and and optimizing the transmit power.

References

[1] R. Huang and V. W. S. Wong, "Neural combinatorial optimization for throughput maximization in IRS-aided systems," in *Proc. of IEEE Global Commun. Conf. (GLOBECOM)*, Taipei, Taiwan, Dec. 2020.

[2] C. Huang, S. Hu, G. C. Alexandropoulos, A. Zappone, C. Yuen, R. Zhang, M. D. Renzo, and M. Debbah, "Holographic MIMO surfaces for 6G wireless networks: Opportunities, challenges, and trends," *IEEE Wireless Commun.*, vol. 27, no. 5, pp. 118–125, Oct. 2020.

[3] V. W. S. Wong, R. Schober, D. W. K. Ng, and L. Wang, *Key Technologies for 5G Wireless Systems*. Cambridge University Press, 2017.

[4] S. Gong, X. Lu, D. T. Hoang, D. Niyato, L. Shu, D. I. Kim, and Y. Liang, "Toward smart wireless communications via intelligent reflecting surfaces: A contemporary survey," *IEEE Commun. Surveys & Tuts.*, vol. 22, no. 4, pp. 2283–2314, Fourth Quarter 2020.

[5] C. Chaccour, M. N. Soorki, W. Saad, M. Bennis, and P. Popovski, "Risk-based optimization of virtual reality over terahertz reconfigurable intelligent surfaces," in *Proc. of IEEE Int'l Conf. Commun. (ICC)*, Jun. 2020.

[6] Q. Wu and R. Zhang, "Intelligent reflecting surface enhanced wireless network: Joint active and passive beamforming design," in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, Abu Dhabi, UAE, Dec. 2018.

[7] X. Yu, D. Xu, and R. Schober, "MISO wireless communication systems via intelligent reflecting surfaces : (invited paper)," in *Proc. of IEEE/CIC Int'l Conf. Commun. in China (ICCC)*, Chengdu, China, Aug. 2019.

[8] S. Zhang and R. Zhang, "Intelligent reflecting surface aided multiple access: Capacity region and deployment strategy," in *Proc. of IEEE Int'l Workshop Signal Process. Advances Wireless Commun.*, May 2020.

[9] Y. Jia, C. Ye, and Y. Cui, "Analysis and optimization of an intelligent reflecting surface-assisted system with interference," in *Proc. of IEEE Int'l Conf. Commun. (ICC)*, Jun. 2020.

[10] S. Abeywickrama, R. Zhang, Q. Wu, and C. Yuen, "Intelligent reflecting surface: Practical phase shift model and beamforming optimization," *IEEE Trans. Commun.*, vol. 68, no. 9, pp. 5849–5863, Sept. 2020.

[11] D. Xu, X. Yu, Y. Sun, D. W. K. Ng, and R. Schober, "Resource allocation for IRS-assisted full-duplex cognitive radio systems," *IEEE Tran. Commun.*, vol. 68, no. 12, pp. 7376–7394, Dec. 2020.

[12] X. Ma, S. Guo, H. Zhang, Y. Fang, and D. Yuan, "Joint beamforming and reflecting design in reconfigurable intelligent surface-aided multi-user communication systems," *IEEE Trans. Wireless Commun.*, vol. 20, pp. 3269–3283, May 2021.

[13] Z. Wan, Z. Gao, F. Gao, M. D. Renzo, and M.-S. Alouini, "Terahertz massive MIMO with holographic reconfigurable intelligent surfaces," *IEEE Trans. Commun.*, vol. 69, no. 7, pp. 4732–4750, Jul. 2021.

[14] K. Feng, X. Li, Y. Han, S. Jin, and Y. Chen, "Physical layer security enhancement exploiting intelligent reflecting surface," *IEEE Commun. Lett.*, vol. 25, no. 3, pp. 734–738, Mar. 2021.

[15] C. Huang, A. Zappone, G. C. Alexandropoulos, M. Debbah, and C. Yuen, "Reconfigurable intelligent surfaces for energy efficiency in wireless communication," *IEEE Trans. Wireless Commun.*, vol. 18, no. 8, pp. 4157–4170, Aug. 2019.

[16] K. Shen and W. Yu, "Fractional programming for communication systems - Part I: Power control and beamforming," *IEEE Trans. Signal Process.*, vol. 66, no. 10, pp. 2616–2630, May 2018.

[17] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *Proc. of Int'l Conf. Learning Representations, (ICLR)*, San Juan, Puerto Rico, May 2016.

[18] K. Feng, Q. Wang, X. Li, and C.-K. Wen, "Deep reinforcement learning based intelligent reflecting surface optimization for MISO communication systems," *IEEE Wireless Commun. Lett.*, vol. 9, no. 5, pp. 745–749, May 2020.

[19] C. Huang, R. Mo, and C. Yuen, "Reconfigurable intelligent surface assisted multiuser MISO systems exploiting deep reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 8, pp. 1839–1850, Aug. 2020.

[20] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio, "Neural combinatorial optimization with reinforcement learning," *CoRR*, vol. abs/1611.09940, Jan. 2017. [Online]. Available: http://arxiv.org/abs/1611.09940

[21] W. Kool, H. Van Hoof, and M. Welling, "Attention, learn to solve routing problems!" in *Proc. of Int'l Conf. Learn. Representations (ICLR)*, New Orleans, LA, May 2019.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, Long Beach, CA, Dec. 2017.

[23] M. Nazari, A. Oroojlooy, M. Takáč, and L. V. Snyder, "Reinforcement learning for solving the vehicle routing problem," in *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, Montréal, Canada, Dec. 2018.

[24] C. He, Y. Hu, Y. Chen, and B. Zeng, "Joint power allocation and channel assignment for NOMA with deep reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2200–2210, Aug. 2019.

[25] M. Ma and V. W. S. Wong, "Joint user pairing and association for multicell NOMA: A pointer network-based approach," in *Proc. of IEEE Int'l Conf. Commun. (ICC) Workshop*, Jun. 2020.

[26] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proc. of Int'l Conf. Machine Learn. (ICML)*, Montreal, Canada, Jun. 2009.

[27] V. K. N. Lau, "Proportional fair space-time scheduling for wireless communications," *IEEE Trans. Commun.*, vol. 53, no. 8, pp. 1353–1360, Aug. 2005.

[28] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, "Deep sets," in *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, Long Beach, CA, Dec. 2017.

[29] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh, "Set transformer: A framework for attention-based permutation-invariant neural networks," in *Proc. of Int'l Conf. Machine Learn. (ICML)*, Long Beach, CA, Jun. 2019.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, Jun. 2016.

[31] A. E. Orhan and X. Pitkow, "Skip connections eliminate singularities," in *Proc. of Int'l Conf. Learn. Representations (ICLR)*, Vancouver, Canada, May 2018.

[32] M. Ilse, J. Tomczak, and M. Welling, "Attention-based deep multiple instance learning," in *Proc. of Int'l Conf. Machine Learn. (ICML)*, Stockholm, Sweden, Jul. 2018.

[33] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3-4, pp. 229–256, May 1992.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. of Int'l Conf. Learning Representations, (ICLR)*, San Diego, CA, May 2015.

[35] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone, "Curriculum learning for reinforcement learning domains: A framework and survey," *Journal of Machine Learning Research*, vol. 21, pp. 1–50, Jul. 2020.

[36] G. Hacohen and D. Weinshall, "On the power of curriculum learning in training deep networks," in *Proc. of Int'l Conf. Machine Learn. (ICML)*, Long Beach, CA, Jun. 2019.

[37] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. of Int'l Conf. Machine Learn. (ICML)*, Beijing, China, Jun. 2014.

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

[39] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. of Int'l Conf. Machine Learn. (ICML)*, Stockholm, Sweden, Jun. 2018.

[40] 3GPP TR 38.901 V16.1.0, "Technical specification group radio access network; Study on channel model for frequencies from 0.5 to 100 GHz (Release 16)," Dec. 2019.

[41] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, Dec. 2019.

[42] G. Zhou, C. Pan, H. Ren, K. Wang, and A. Nallanathan, "A framework of robust transmission design for IRS-aided MISO communications with imperfect cascaded channels," *IEEE Trans. Signal Process.*, vol. 68, pp. 5092–5106, Aug. 2020.

[43] J. Zhang, M. Kountouris, J. G. Andrews, and R. W. Heath, "Multi-mode transmission for the MIMO broadcast channel with imperfect channel state information," *IEEE Trans. Commun.*, vol. 59, no. 3, pp. 803–814, Mar. 2011.

Rui Huang (S'19) received the B.Eng. degree from Chongqing University, Chongqing, China, in 2015, and the M.Eng. degree from Shanghai Jiao Tong University (SJTU), Shanghai, China, in 2018. He is currently a Ph.D. Candidate in the Department of Electrical and Computer Engineering, The University of British Columbia (UBC), Vancouver, Canada. He has been a recipient of the Four Year Doctoral Fellowship (4YF) at UBC since 2018. His research interests include Internet of Things (IoT) and machine learning for optimization in wireless communication systems.

Vincent W.S. Wong (S'94, M'00, SM'07, F'16) received the B.Sc. degree from the University of Manitoba, Winnipeg, MB, Canada, in 1994, the M.A.Sc. degree from the University of Waterloo, Waterloo, ON, Canada, in 1996, and the Ph.D. degree from the University of British Columbia (UBC), Vancouver, BC, Canada, in 2000. From 2000 to 2001, he worked as a systems engineer at PMC-Sierra Inc. (now Microchip Technology Inc.). He joined the Department of Electrical and Computer Engineering at UBC in 2002 and is currently a Professor. His research areas include protocol design, optimization, and resource management of communication networks, with applications to wireless networks, smart grid, mobile edge computing, and Internet of Things. He received the Best Paper Award at the *IEEE GLOBECOM* 2020. Currently, Dr. Wong is the Chair of the Executive Editorial Committee of *IEEE Transactions on Wireless Communications*, an Area Editor of *IEEE Transactions on Communications* and *IEEE Open Journal of the Communications Society*, and an Associate Editor of *IEEE Transactions on Mobile Computing*. He has served as a Guest Editor of *IEEE Journal on Selected Areas in Communications*, *IEEE Internet of Things Journal*, and *IEEE Wireless Communications*. He has also served on the editorial boards of *IEEE Transactions on Vehicular Technology* and *Journal of Communications and Networks*. He was a Tutorial Co-Chair of *IEEE GLOBECOM*'18, a Technical Program Co-chair of *IEEE VTC*2020-*Fall* and *IEEE SmartGridComm*'14, as well as a Symposium Co-chair of *IEEE ICC*'18, *IEEE SmartGridComm* ('13, '17) and *IEEE GLOBECOM*'13. He is the Chair of the IEEE Vancouver Joint Communications Chapter and has served as the Chair of the IEEE Communications Society Emerging Technical Sub-Committee on Smart Grid Communications. He was an IEEE Communications Society Distinguished Lecturer (2019 - 2020).