Deep Reinforcement Learning for Radio Resource Allocation in NOMA-based Remote State Estimation

Gaoyang Pang, Wanchun Liu*, Yonghui Li, and Branka Vucetic
School of Electrical and Information Engineering, The University of Sydney, Australia
Emails: {gaoyang.pang, wanchun.liu, yonghui.li, branka.vucetic}@sydney.edu.au.

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10001377

Abstract—**Remote state estimation, where many sensors send their measurements of distributed dynamic plants to a remote estimator over shared wireless resources, is essential for mission-critical applications of Industry 4.0. Most of the existing works on remote state estimation assumed orthogonal multiple access and the proposed dynamic radio resource allocation algorithms can only work for very small-scale settings. In this work, we consider a remote estimation system with non-orthogonal multiple access. We formulate a novel dynamic resource allocation problem for achieving the minimum overall long-term average estimation mean-square error. Both the estimation quality state and the channel quality state are taken into account for decision making at each time. The problem has a large hybrid discrete and continuous action space for joint channel assignment and power allocation. We propose a novel action-space compression method and develop an advanced deep reinforcement learning algorithm to solve the problem. Numerical results show that our algorithm solves the resource allocation problem effectively, presents much better scalability than the literature, and provides significant performance gain compared to some benchmarks.**

Index Terms—**Remote state estimation, radio resource allocation, NOMA, deep reinforcement learning, task-oriented communications.**

I. Introduction

Wireless networked control systems (WNCSs), consisting of spatially distributed plants, sensors, machines, actuators and controllers, play an essential role in the era of Industry 4.0 [1]. In particular, remote state estimators for monitoring dynamic plant status in a real-time manner are critical in WNCSs to enable high-quality closed-loop control. In Industry 4.0, massive wireless sensors are deployed for remote state estimation of spatially distributed plants. Thus, it is essential to manage the limited wireless radio resources for remote state estimation of many distributed plants.

Existing works on wireless resource allocation mainly focused on data-oriented communications, and the design targets are transmission throughput, latency, and reliability [2]. Advanced data-driven machine learning approaches, such as supervised learning and reinforcement learning, have been adopted when resource allocation problems cannot be solved effectively by conventional model-based methods [3]. Different from data-oriented communications, the resource allocation design in a remote estimation system should be task-oriented as the goal is to minimize the long-term average remote estimation mean-square error (MSE) of the dynamic plant states [2].

Significant efforts have been devoted to the wireless resource allocation of remote estimation systems with orthogonal multiple access (OMA) (see [4], [5] and the references therein), where each frequency channel (i.e., a subcarrier) can only be assigned to a single sensor to avoid inter-user interference completely. Non-orthogonal multiple access (NOMA) [6] allows simultaneous transmission of multiple sensors' packets at the same frequency channel and offers higher transmission capacity than OMA. Each sensor packet can be detected by processing received superimposed signals.

However, existing works on dynamic resource allocation in NOMA-based remote estimation only considered the simple multi-sensor-single-channel setting, and focused on either power allocation [7]–[9] or channel assignment problems [10], rather than joint design ones. Moreover, the developed policy optimization methods for resource allocation can only handle very small-scale systems, e.g., a five-sensor-single-channel setting [8], due to the curse of dimensionality in policy optimization. We aim to address these limitations of NOMA-based remote estimation. The novel contributions of our work are summarized below.

1) New system model. We investigate a NOMA-based remote estimation system with multiple sensors and frequency channels. It requires a joint design of channel assignment and power control of each sensor for achieving the optimal overall remote estimation performance. Such a system has not been investigated before.

2) Novel problem formulation. We formulate the dynamic resource allocation problem into a Markov decision process (MDP) problem. It takes into account both estimation quality states and channel quality states for decision making. To the best of our knowledge, such a problem has not been considered in the literature of remote state estimation. The action space of the formulated MDP is a multi-dimensional hybrid one consisting of discrete channel assignment and continuous power allocation, while classical MDP solutions only work for discrete action spaces. The new (enlarged) state space and the hybrid action space make the decision-making problem difficult to solve, especially in large systems.

3) Advanced dynamic resource allocation algorithm with large state and action spaces. To handle the optimal resource allocation problem, we develop an advanced data-driven deep reinforcement learning (DRL) algorithm that generates low-dimensional continuous virtual actions, and propose a novel action mapping scheme to map virtual actions into real hybrid actions in resource allocation. Extensive simulation results illustrate that the proposed DRL algorithm can effectively solve

*Wanchun Liu is the corresponding author.

978-1-6654-3540-6/22/$31.00 © 2022 IEEE

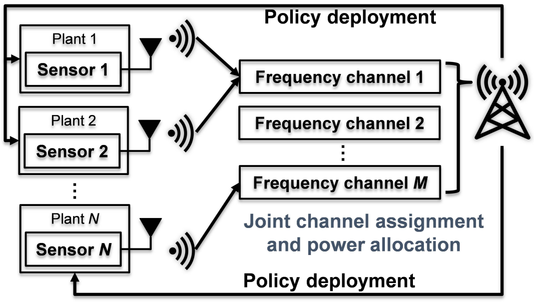

Fig. 1. A NOMA-based N-sensor-M-channel remote estimation system.

the dynamic resource allocation problem with a much larger scale than the literature, and provides significant performance gain compared with some benchmark policies, especially when the system scale is large.

II. System Model of Remote State Estimation

The studied remote estimation system consists of N dynamic plants, each monitored by a sensor, and a remote estimator, as shown in Fig. 1. The N sensors send the measurement data to the remote estimator over M frequency channels (i.e., subcarriers), where $M < N$. Each wireless device is equipped with a single antenna. The remote estimator applies a dynamic resource allocation policy for channel assignment and power control of each sensor.

A. Local State Estimation

We consider discrete-time linear time-invariant (LTI) systems, where the model of plant n is given as [5], [11]

$$\begin{aligned}\mathbf{x}_n(t+1) &= \mathbf{A}_n\mathbf{x}_n(t) + \mathbf{w}_n(t)\\ \mathbf{y}_n(t) &= \mathbf{C}_n\mathbf{x}_n(t) + \mathbf{v}_n(t)\end{aligned} \tag{1}$$

where $\mathbf{x}_n(t) \in \mathbb{R}^{l_n}$ is the plant state vector; $\mathbf{A}_n \in \mathbb{R}^{l_n \times l_n}$ is the plant state transition matrix; $\mathbf{y}_n(t) \in \mathbb{R}^{r_n}$ is the sensor measurement vector; $\mathbf{C}_n \in \mathbb{R}^{r_n \times l_n}$ is the measurement matrix; $\mathbf{w}_n(t) \in \mathbb{R}^{l_n}$ and $\mathbf{v}_n(t) \in \mathbb{R}^{r_n}$ are the plant disturbance and sensing measurement noise vectors, respectively. These noise vectors are independent and identically distributed (i.i.d.) zero-mean Gaussian processes with covariance matrices $\mathbf{W}_n$ and $\mathbf{V}_n$, respectively.

Due to the measurement distortion and noise in (1), each sensor has a local estimator to pre-estimate the plant state $\mathbf{x}_n(t)$ based on the raw measurement $\mathbf{y}_n(t)$ before sending it to the remote estimator. We adopt the classical Kalman filter (KF) for generating the estimated state $\mathbf{x}_n^s(t)$ as it is the optimal estimator of LTI systems in the average estimation MSE [5]. The definition of local estimation error covariance $\mathbf{P}_n^s(t)$ is given as

$$\mathbf{P}_n^s(t) \triangleq \mathbb{E}\left[\left(\mathbf{x}_n^s(t) - \mathbf{x}_n(t)\right)\left(\mathbf{x}_n^s(t) - \mathbf{x}_n(t)\right)^{\mathrm{T}}\right]. \tag{2}$$

Since our work focuses on the remote state estimation, each local estimator is assumed to be stable and operate in the steady state, i.e., the estimation error covariance of the local KF is a constant $\mathbf{P}_n^s(t) \triangleq \bar{\mathbf{P}}_n, \forall t \in \mathbb{N}$, where $\mathbb{N}$ is the set of positive integers [4], [5], [8], [11].

B. Wireless Channel Model and Communications

We consider finite-state Markov block-fading channels [12]. The overall channel power gain state matrix is denoted as $\mathbf{G}(t)$, where each column vector $\mathbf{g}_n(t) \triangleq (g_{n,1}(t), \ldots, g_{n,M}(t))^{\mathrm{T}}$ represents the channel power gain between sensor n and the remote estimator over M channels. Each channel power gain has H states, i.e., $g_{n,m}(t) \in \mathcal{G} \triangleq \{h_1, h_2, \ldots, h_H\}$. The channel state vector $\mathbf{g}_n(t) \in \mathcal{G}^M \triangleq \{\tilde{\mathbf{g}}_1, \tilde{\mathbf{g}}_2, \ldots, \tilde{\mathbf{g}}_{H^M}\}$ is modeled as a multi-state Markov chain. Since sensors are dislocated and have different radio propagation environments, we assume that their Markov channel states are independent. The remote estimator has the knowledge of channel state information obtained via standard channel estimation techniques. The channel state transition matrices are unknown to the remote estimator, because the estimation of a multi-dimensional Markov chain model is computationally intensive [13].

We adopt short-packet communications for sensor packet transmissions [14]. Given the packet length l (i.e., the number of symbols per packet), the number of data bits b, the signal-to-noise ratio (SNR) γ_n of sensor n, we have the Shannon capacity $\mathcal{C}(\gamma_n) = \log_2(1+\gamma_n)$ and the channel dispersion $\mathcal{V}(\gamma_n) = (1-(1+\gamma_n)^{-2})(\log_2 e)^2$. Then, the decoding failure probability of sensor n's packet can be approximated as [15]

$$\varepsilon(\gamma_n) \approx \mathcal{Q}\left(\frac{\mathcal{C}(\gamma_n) - \frac{b}{l}}{\sqrt{\frac{\mathcal{V}(\gamma_n)}{l}}}\right),$$

where $\mathcal{Q}(x) = (\frac{1}{\sqrt{2\pi}})\int_x^\infty e^{-\frac{t^2}{2}}\mathrm{d}t$ is the Gaussian Q-function.

C. Multiple-Access Scheme

In the NOMA scheme, each sensor takes at most one channel for transmission, while each channel can be allocated to multiple sensors at the same time. At each time slot, the remote estimator needs to determine both the sensor-to-channel assignment and the sensor transmission power for interference management. Let $\mathbf{d}_n(t) \triangleq (d_{n,1}(t), d_{n,2}(t), \ldots, d_{n,M}(t))^{\mathrm{T}} \in \{0,1\}^M$ denotes the binary channel selection action of sensor n at the M channels, where $\sum_{m=1}^M d_{n,m}(t) \leq 1$. Let $\mathbf{p}^{\mathrm{tx}}(t) \triangleq (P_1^{\mathrm{tx}}(t), P_2^{\mathrm{tx}}(t), \ldots, P_N^{\mathrm{tx}}(t))^{\mathrm{T}} \in \mathbb{P}^N$ denotes the transmission power of N sensors at time t, where $\mathbb{P} \triangleq [0, P_{max}]$. Then, the received signal power of sensor n is $P_n^{\mathrm{rx}}(t) = P_n^{\mathrm{tx}}(t)\left((\mathbf{d}_n(t))^{\mathrm{T}}\mathbf{g}_n(t)\right)$.

To decode sensor signals at the same channel, the remote estimator performs successive interference cancellation (SIC) with the decreasing order of the received sensor signal power [6], i.e., the strongest/weakest sensor signal is decoded first/last. The first sensor packet is decoded by treating all other sensor signals as interference. Once it is decoded successfully, the sensor signal can be reconstructed perfectly and thus removed from the received signal. Then, the second sensor packet will be decoded without the interference of the first one. The decoder stops once a decoding failure occurs or the last sensor packet has been successfully decoded. Assuming that

$P_1^{\mathrm{rx}}(t) \geq P_2^{\mathrm{rx}}(t) \geq \ldots \geq P_N^{\mathrm{rx}}(t)$, the signal-to-interference-plus-noise ratio (SINR) for decoding sensor n's packet is

$$\gamma_n(t) = \frac{P_n^{\mathrm{rx}}(t)}{\sum_{i=n+1}^{N} \left(\mathbf{d}_n(t)\right)^{\mathrm{T}} \mathbf{d}_i(t) P_i^{\mathrm{rx}}(t) + \sigma^2},$$

where σ^2 is the receiving noise power. The decoding failure probability of sensor n can be obtained as

$$\hat{\varepsilon}_n(t) = \begin{cases} \varepsilon\left(\gamma_1(t)\right), & n = 1 \\ U_{n,1} + \sum_{k=2}^{n} \left(U_{n,k} \prod_{i=1}^{k-1} \left(1 - U_{k,i}\right)\right), & n > 1. \end{cases}$$

where $U_{n,n} = \varepsilon\left(\gamma_n(t)\right)$ and $U_{n,i} = \left(\mathbf{d}_n(t)\right)^{\mathrm{T}} \mathbf{d}_i(t) \hat{\varepsilon}_i(t), \forall i \leq n$.

D. Remote State Estimation

Due to transmission scheduling and packet detection errors, sensor n's packet may not be received by the remote estimator at each time slot. Let $\zeta_n(t) = 1$ denote the successful packet detection of sensor n at t. To provide real-time state estimation of all plants, the remote estimator knows plant state transition matrices of all plants and employs a minimum mean-square error (MMSE) state estimation for each plant [5], [11]

$$\hat{\mathbf{x}}_n(t) = \begin{cases} \mathbf{x}_n^s(t) & \text{if } \zeta_n(t) = 1 \\ \mathbf{A}_n \hat{\mathbf{x}}_n(t-1) & \text{otherwise.} \end{cases} \tag{3}$$

Thus, the remote estimation error covariance is

$$\mathbf{P}_n(t) \triangleq \mathbb{E}\left[\left(\hat{\mathbf{x}}_n(t) - \mathbf{x}_n(t)\right)\left(\hat{\mathbf{x}}_n(t) - \mathbf{x}_n(t)\right)^{\mathrm{T}}\right] \tag{4}$$

$$= \begin{cases} \bar{\mathbf{P}}_n & \text{if } \zeta_n(t) = 1 \\ \mathbf{A}_n \mathbf{P}_n(t-1) \mathbf{A}_n^{\mathrm{T}} + \mathbf{W}_n & \text{otherwise} \end{cases} \tag{5}$$

where (5) is obtained by taking (3) and (2) into (4). Recall that $\bar{\mathbf{P}}_n$ is the local estimation error covariance.

Now we define $\tau_n(t)$ as the age of information (AoI) of sensor n at time slot t, representing the time interval since the last successful transmission of sensor n to the remote estimator. Therefore, the AoI state of sensor n has the updating rule below

$$\tau_n(t) = \begin{cases} 1 & \text{if } \zeta_n(t-1) = 1 \\ \tau_n(t-1) + 1 & \text{otherwise.} \end{cases} \tag{6}$$

A larger AoI indicates that the remote estimation is less accurate. Jointly using (5) and (6), the remote estimation error covariance can be concisely rewritten as a function of AoI as

$$\mathbf{P}_n(t) = f_n^{\tau_n(t)}\left(\bar{\mathbf{P}}_n\right),$$

where $f_n^1(\mathbf{X}) = \mathbf{A}_n \mathbf{X} \mathbf{A}_n^{\mathrm{T}} + \mathbf{W}_n$ and $f_n^{\tau_n}(\mathbf{X}) = f_n^1\left(f_n^{\tau_n - 1}(\mathbf{X})\right)$.

To quantify the remote estimation quality of sensor n at time t, we define the *estimation cost function*, i.e., the sum estimation MSE of the plant vector state, as

$$\begin{aligned} J_n(t) &\triangleq \mathbb{E}\left[\left(\hat{\mathbf{x}}_n(t) - \mathbf{x}_n(t)\right)^{\mathrm{T}}\left(\hat{\mathbf{x}}_n(t) - \mathbf{x}_n(t)\right)\right] \\ &= \mathrm{Tr}\left(\mathbf{P}_n(t)\right) = \mathrm{Tr}\left(f_n^{\tau_n(t)}\left(\bar{\mathbf{P}}_n\right)\right), \end{aligned}$$

where $\mathrm{Tr}(\cdot)$ is the matrix trace operator. Thus, the AoI state and the local estimation error covariance jointly determine the remote estimation quality. A smaller $J_n(t)$ indicates that the remote state estimation is more accurate. As proved in our previous work [11], if the spectral radius of plant n's state transition matrix $\mathbf{A}_n$ is greater than 1, the increasing AoI gives rise to the exponential growth of estimation cost.

III. Problem Formulation

We aim to design a deterministic and stationary resource allocation policy denoted as $\pi(\cdot) \in \Pi$ that generates channel allocation and power control actions of all sensors at each time slot, for achieving the optimal discounted long-term average estimation quality [5], i.e.,

$$J^* = \min_{\pi(\cdot) \in \Pi} \lim_{T \to \infty} \mathbb{E}\left[\sum_{t=0}^{T-1} \sum_{n=1}^{N} \lambda^t J_n(t)\right]$$

where $\lambda \in (0,1)$ is a discount factor and $\mathbb{E}[\cdot]$ is the expectation operator. We formulate such a sequential decision-making problem into an MDP.

A. MDP Formulation

In general, the MDP takes both the channel quality state and the AoI state of all sensors as inputs and generates the resource allocation action at each time. The Markovian property holds directly due to the Markov channel modeling and the AoI state updating rule.

State: Given the channel state matrix $\mathbf{G}(t) \in \mathcal{G}^{M \times N}$, and the AoI state $\boldsymbol{\tau}(t) = (\tau_1(t), \tau_2(t), \ldots, \tau_N(t)) \in \mathbb{N}^N$, the state of the MDP is defined as $\mathbf{s}(t) \triangleq \{\mathbf{G}(t), \boldsymbol{\tau}(t)\} \in \mathcal{G}^{M \times N} \times \mathbb{N}^N$.

Action: The hybrid action $\mathbf{a}(t) \triangleq \{\mathbf{D}(t), \mathbf{p}^{\mathrm{tx}}(t)\}$ takes into account both the discrete channel allocation $\mathbf{D}(t)$ and the continuous power control $\mathbf{p}^{\mathrm{tx}}(t) \in \mathbb{P}^N$. The discrete action space is denoted as $\mathcal{A}$ with the cardinality of $|\mathcal{A}| = (M+1)^N$, as each sensor has $M+1$ options for channel selection. The hybrid action space is denoted as $\mathcal{A} \times \mathbb{P}^N$.

Transition: The state-transition probability of MDP consists of the channel state transition and the AoI state transition. Since the former does not depend on the latter, the state-transition probability from state $\mathbf{s}(t)$ to state $\mathbf{s}(t+1)$ under a particular action $\mathbf{a}(t)$ can be written as $\Pr[\mathbf{s}(t+1)|\mathbf{s}(t), \mathbf{a}(t)] = \Pr[\mathbf{G}(t+1)|\mathbf{G}(t)] \Pr[\boldsymbol{\tau}(t+1)|\mathbf{s}(t), \mathbf{a}(t)]$. Recall that the knowledge of $\Pr[\mathbf{G}(t+1)|\mathbf{G}(t)]$ is unavailable (Section II-B).

Policy: The policy is a mapping between the state and the action as $\mathbf{a}(t) = \pi(\mathbf{s}(t))$, where $\pi(\cdot) \in \Pi$.

Reward: We define the reward of the MDP as the negative sum estimation cost, i.e., $r(t) = -J(t) = -\sum_{n=1}^{N} J_n(t)$. Thus, one needs to find a resource allocation policy for maximizing the discounted long-term average reward $\lim_{T \to \infty} \mathbb{E}\left[\sum_{t=0}^{T-1} \lambda^t r(t)\right]$.

B. Challenges for Solving the MDP

In the absence of state transition probabilities, conventional model-based MDP algorithms cannot work (e.g., value and policy iteration algorithms), and thus data-driven reinforcement learning approaches are preferable. Due to the curse of dimensionality introduced by the large state space $\mathcal{G}^{N \times M} \times \mathbb{N}^N$, conventional data-driven reinforcement learning

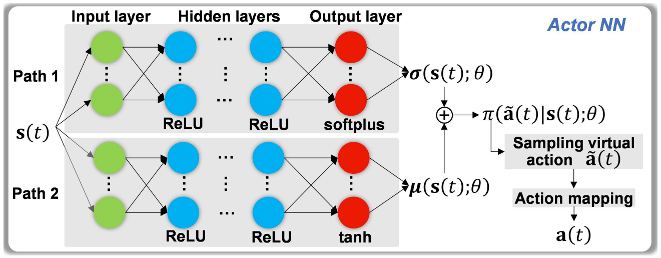
Input layer
Hidden layers
Output layer
Actor NN
Path 1
Path 2
$\mathbf{s}(t)$
ReLU
ReLU
softplus
ReLU
ReLU
tanh
$\boldsymbol{\sigma}(\mathbf{s}(t);\theta)$
$\boldsymbol{\mu}(\mathbf{s}(t);\theta)$
$\pi(\tilde{\mathbf{a}}(t)|\mathbf{s}(t);\theta)$
Sampling virtual action $\tilde{\mathbf{a}}(t)$
Action mapping
$\mathbf{a}(t)$

Fig. 2. The framework for generating resource allocation actions based on a trained actor NN (Section IV-B) and the action mapping scheme (Section IV-A).

algorithms, such as Q-learning, are not applicable. To deal with the large state space issue, we resort to more advanced DRL algorithms adopting deep neural networks (DNNs) for value function approximation, though there are several challenges remaining.

1) Large and hybrid action space: The MDP has a large action space, even with relatively small numbers of sensors and channels. For example, when $N = 10, M = 5$, there are $(M+1)^N = 60466176$ discrete actions for channel assignment. For the continuous action part, a commonly adopted method is action discretization. In the simplest 2-level quantization scenario, there are $2^N = 1024$ actions for power control and thus $1024 \times 60466176 \approx 6 \times 10^{10}$ discrete actions in total for joint channel assignment and power control. However, deep Q-network (DQN), the most popular DRL algorithm adopted for solving MDPs with discrete actions, cannot handle such an MDP. The large action space entails large DNNs and makes the DQN difficult to train, requiring a huge storage space and a tremendous amount of computing power. Solving MDPs with large action spaces are challenging.

2) Training difficulties: In our MDP problem, the channel state is of high dynamics and the AoI state is also a stochastic function of the channel state and the action. In addition, as mentioned in Section II-D, the absolute value of reward grows exponentially fast with respect to the AoI state, which means a DRL agent needs to deal with a fairly large range of rewards, inducing a highly fluctuated training process that is difficult to converge. This feature is different from many existing works applying DRL in wireless communications problems [3], where the reward/cost commonly grows up in linear or log scale with the increasing state. Therefore, the highly stochastic states and the large reward range can lead to unstable training process and make a DRL agent difficult to converge to an optimal policy.

IV. Deep Reinforcement Learning Algorithm

To solve the challenges above, first, we design low-dimensional continuous virtual actions of the resource allocation problem and propose a novel scheme that maps virtual action $\tilde{\mathbf{a}}(t)$ to real hybrid discrete and continuous action $\mathbf{a}(t)$. Second, we choose a proper DRL framework with a continuous action space, which provides a stable training process to learn the optimal policy with virtual actions. The framework is illustrated in Fig. 2.

A. Virtual Action Design and Action Mapping Scheme

Considering the hybrid action feature of resource allocation, it is natural to design a multi-dimensional continuous virtual action of each sensor to represent the hybrid action. However, how to design a continuous virtual action and map it to a discrete channel selection action for effective DRL is highly non-trivial. A naive method is to adopt a scalar continuous virtual action for each sensor and discretize it linearly into $M+1$ levels for channel selection. Level 0 means no channel selection and Level $m > 0$ denotes the selection of channel m. At first glance, the method compresses the M-size discrete action space by a one-dimensional continuous one. However, the method implicitly converts the original non-Euclidean action space into the Euclidean one. This potentially requires the DRL agent to learn a highly discontinuous multi-level-multi-stage piecewise policy function. Ideally, a slight change of state can make a significant change in action output. This makes the DRL difficult to converge to an optimal one.

To solve the issue of discontinuous function approximation, we propose to convert the decimal channel selection action (i.e., from 0 to M) of each sensor to a binary sequence with a length of $\lceil \log_2(M+1) \rceil$. Thus, we design the virtual action for each sensor with $\lceil \log_2(M+1) \rceil$ elements. Each element is a real number, and the positive and negative values are mapped to '1' and '0', respectively, for real action mapping. Now, the virtual action has $N\lceil \log_2(M+1) \rceil$ dimension for channel selection of all sensors. Each element of the virtual action only handles the selection of two discrete actions. Changing of a single virtual action entry from negative to positive represents the reduced and increased likelihood of one action and the other, respectively. The continuity can make the DRL easy to train. The original $(M+1)^N$-size discrete action space is compressed by the $N\lceil \log_2(M+1) \rceil$-dimension continuous one, which scales linearly and logarithmically in N and M, respectively. One dimensional virtual action $\tilde{P}_n^{\text{tx}}(t) \in \mathbb{R}$ is added for each sensor's transmit power control, and the mapping between the virtual to real power control action is $P_n^{\text{tx}}(t) = P_{\max}(\text{clamp}_{-1}^{1}[\tilde{P}_n^{\text{tx}}(t)] + 1)/2$, where the operator $\text{clamp}_{-1}^{1}[\cdot]$ is for truncating a variable to values between -1 and 1. Thus, the virtual action $\tilde{\mathbf{a}}(t)$ has $N\lceil \log_2(M+1) \rceil + N$ elements in total.

B. Policy Optimization with Virtual Actions

There are many DRL frameworks with various features developed during the past decade. To match the key problem features in Section III-B, we adopt the proximal policy optimization (PPO) [16], a *policy-based* DRL method that learns a *stochastic policy* via *on-policy learning*, for solving the MDP. The reasons include: 1) compared with off-policy DRL methods, on-policy ones are more stable in highly stochastic environments due to the guaranteed monotonic performance improvement based their policy updating mechanism [17]; 2) existing research shows that stochastic policy based DRL algorithms lead to smoother function approximations [18], and thus provide higher speed for training convergence and better training stability than deterministic ones when the training environment is of high dynamics [19]; 3) policy-based DRL generates multi-dimensional continuous actions directly and can deal with MDPs with continuous action spaces.

The PPO agent has a pair of actor and critic neural networks (NNs), where the NN parameters, including weights and biases, are denoted by θ and φ, respectively. Each of the NNs has an input layer, multiple hidden layers, an output layer, and adopts a fully connected and feedforward structure. The input layers of the actor and critic NNs receive the MDP state $\mathbf{s}(t)$. All hidden layers adopt the activation function of rectified linear unit (ReLU) for nonlinear function approximation.

We design the actor NN for generating a multi-dimensional *continuous virtual action* $\tilde{\mathbf{a}}(t)$ with stochastic policy $\tilde{\pi}(\tilde{\mathbf{a}}(t)|\mathbf{s}(t);\theta)$, which is a probability density function of $\tilde{\mathbf{a}}(t)$ given the current state $\mathbf{s}(t)$. The virtual action is then mapped to the real action $\mathbf{a}(t)$ for radio resource allocation as discussed in Section IV-A to obtain the next state $\mathbf{s}(t+1)$ and reward $r(t+1)$. The actor output layer generates mean $\boldsymbol{\mu}(\mathbf{s}(t);\theta)$ and standard deviation $\boldsymbol{\sigma}(\mathbf{s}(t);\theta)$ of $\tilde{\mathbf{a}}(t)$. We adopt tanh and softplus as the activation functions for the mean and the standard deviation outputs, respectively. The former is to bound the mean within $[-1,1]$, while the latter is to guarantee a positive standard deviation. The critic NN estimates the state-value function $V(\mathbf{s}(t);\varphi)$ given the actor's policy $\tilde{\pi}(\tilde{\mathbf{a}}(t)|\mathbf{s}(t);\theta)$, i.e.,

$$V(\mathbf{s}(t);\varphi)\approx\mathbb{E}\left[\sum_{k=0}^{\infty}\lambda^k r(t+k)\middle|\tilde{\pi}(\tilde{\mathbf{a}}(t+k)|\mathbf{s}(t+k);\theta),\forall k\geq 0\right].$$

Thus, the critic NN has a single output.

The training of the PPO agent alternates between experience generation and policy update. In the following, we only present the key steps, and the detailed algorithm can be found in [16].

1) Experience generation. By using current policy $\tilde{\pi}(\cdot|\cdot;\theta_{\text{old}})$, the PPO agent samples data $(\mathbf{s}(t),\tilde{\mathbf{a}}(t),r(t))$ with length of T through interacting with the environment. By leveraging the generated experience, the advantage function $A(t)$ and the reward-to-go function $R(t)$ for each $t = 0,\dots,T-1$ can be calculated as

$$A(t) = \sum_{k=t}^{T-1}(\lambda\alpha)^{k-t}(r(k)+\lambda V(\mathbf{s}(k+1);\varphi)-V(\mathbf{s}(k);\varphi))$$

and

$$R(t) = r(t) + \lambda V(\mathbf{s}(t+1);\varphi)$$

respectively, where α is a hyper-parameter named as the smoothing factor.

2) Policy update. The PPO agent creates a sample batch by randomly sampling B experiences from the generated T-length experience earlier, i.e., $\{(\mathbf{s}(t_i),\tilde{\mathbf{a}}(t_i),A(t_i),R(t_i))\}$ where $t_i\in\{t_1,\dots,t_B\}\subset\{0,\dots,T-1\}$.

The loss function for updating the critic NN is defined as

$$L_{\text{C}}(\varphi) = \frac{1}{B}\sum_{i=1}^{B}\left(R(t_i)-V\left(\mathbf{s}(t_i);\varphi\right)\right)^2,$$

which is named as the temporal difference error, specifying difference of the state-value estimations based on time steps t_i and t_i+1. The loss function for updating the actor NN has been elaborately designed for achieving high training stability and is much more complex:

$$L_{\text{A}}(\theta)=$$
$$\frac{1}{B}\sum_{i=1}^{B}(\min\{p(\mathbf{s}(t_i);\theta)A(t_i),c\left(\mathbf{s}(t_i);\theta\right)A(t_i)\}+w\hbar_{\theta_{\text{old}}}(\tilde{\mathbf{a}}(t_i)))$$

where $p(\mathbf{s}(t_i);\theta) = \frac{\tilde{\pi}(\tilde{\mathbf{a}}(t_i)|\mathbf{s}(t_i);\theta)}{\tilde{\pi}(\tilde{\mathbf{a}}(t_i)|\mathbf{s}(t_i);\theta_{\text{old}})}$ is a density ratio, and $c(\mathbf{s}(t_i);\theta) = \max\left\{\min\left\{p(\mathbf{s}(t_i);\theta),1+\omega\right\},1-\omega\right\}$ is a clip function with a hyper-parameter ω. $\hbar_{\theta_{\text{old}}}(\tilde{\mathbf{a}}(t_i))$ is the entropy loss function of $\tilde{\mathbf{a}}(t_i)$. Since $\tilde{\mathbf{a}}(t_i)$ follows a Gaussian distribution, its entropy loss can be directly obtained based on its standard deviation $\boldsymbol{\sigma}(\mathbf{s}(t_i);\theta_{\text{old}})$. w denotes entropy loss weight factor. Then, the critic NN and the actor NN parameters φ and θ can be updated by optimizing the loss functions $L_{\text{C}}(\varphi)$ and $L_{\text{A}}(\theta)$, respectively, using the widely adopted Adam optimizer.

After training, the virtual action $\tilde{\mathbf{a}}(t)$ can be generated deterministically based on the maximum likelihood method for online deployment, i.e., $\tilde{\mathbf{a}}(t)=\boldsymbol{\mu}(\mathbf{s}(t);\theta)$.

V. Numerical Experiments

A. Experiment Setup

Our numerical experiments are implemented on a computing platform with two Intel Xeon Gold 6256 CPUs @ 3.60 GHz and a 192 GB RAM. Each of the actor and critic NNs of the PPO-based DRL agent has three hidden layers with sizes of $\lceil 70K\rceil,\lceil 50K\rceil,\lceil 30K\rceil$, respectively, where $K=\sqrt{N/M}\log_2(M+1)$. The state input of each NN has $N(M+1)$ dimensions, which is the same as the MDP. The output size of the critic NN is 1, while that of the actor NN is discussed in Section IV-B. The dynamic system matrices $\mathbf{A}_n$ are randomly generated by leveraging the method presented in [5], where the spectrum radius is drawn uniformly from the range of $(1,1.3)$. The channel transition matrices are generated randomly. Table I summarizes the details of the remote estimation system parameters and the DRL parameters.

We adopt the DQN-based channel assignment of a remote estimation system with OMA as a benchmark. As shown in [5], the DQN-based algorithm performs better than the heuristic algorithms (e.g., the greedy and the round-robin policies). Thus, we only need to compare our algorithm with the DQN-based one. In addition, we use the naive action mapping based PPO (discussed in Section IV-A) as a benchmark of our proposed novel action mapping scheme.

B. Performance Evaluation

In Table II, we compare the remote estimation performance, i.e., the average sum MSE of all plants, between the proposed DRL-based algorithm and the benchmarks with various system scales and sensor-to-channel ratios (SCRs). The remote estimation system with a smaller average estimation MSE has

TABLE I
SUMMARY OF EXPERIMENT SETUP

Items	Value
Remote state estimation system parameters	
Transmit power budget [dBm], P_{max}	23
Receiving noise power [dBm], σ^2	-60
Code rate [bps], b/l	2
Block length [symbols], l	200
Markov channel power gain states, $\mathcal{G}$	$\{10^{-8}, 10^{-7}, \cdots, 10^{-1}\}$
Channel state transition matrix	Randomly generalized
Traning parameters	
Episode number, E	$\left\lceil 250 \times \frac{N}{M} \times \sqrt{NM} \right\rceil$
Maximum time steps per episode, T	128
Learning rate of actor NN	0.0001
Learning rate of critic NN	0.001
Mini-batch size, B	128
Discount factor, λ	0.95
Learning agent parameters of this work (PPO)	
The smoothing factor, α	0.95
Entropy loss weight factor, w	0.01
Clip factor, ω	0.2
Learning agent parameters of the benchmark (DQN)	
Initial epsilon for exploring action space	1
Epsilon decay rate	0.999
Minimum epsilon	0.01
Experience buffer length	$1000NM$

TABLE II
PERFORMANCE COMPARISON OF THE PROPOSED ALGORITHM AND THE BENCHMARKS IN TERMS OF AVERAGED ESTIMATION MSE

System scale	OMA	NOMA	
(N, M, SCR)	DQN	Naive action mapping	This work
(6, 3, 2)	46.6243	39.0462	38.0663
(10, 5, 2)	–	65.0766	63.0768
(20, 10, 2)	–	154.9627	125.4118
(30, 15, 2)	–	297.5236	218.3914
(40, 20, 2)	–	397.5561	286.1228
(50, 25, 2)	–	486.9513	360.9214
(10, 4, 2.5)	–	78.8600	77.6075
(20, 8, 2.5)	–	172.9613	157.2169
(30, 12, 2.5)	–	318.8591	252.8929
(40, 16, 2.5)	–	404.8909	345.0570
(50, 20, 2.5)	–	563.4724	434.4692
(15, 5, 3)	–	159.5629	129.8431
(24, 8, 3)	–	275.5904	230.7575
(33, 11, 3)	–	438.7793	343.1833
(42, 14, 3)	–	578.4324	448.6448
(51, 17, 3)	–	765.4839	592.1996

a better performance. Average estimation MSEs in Table II are calculated by 10000-step simulations.

We see that the DQN algorithm with OMA only works for the 6-sensor-3-channel setting and does not even converge for larger systems. The proposed DRL algorithm with NOMA can scale up to 50 sensors and 25 channels. We also see that the proposed action mapping scheme-based algorithm can provide a 25% average estimation MSE reduction than the naive mapping scheme-based one when the system is large (e.g., $N \geq 50$). The performance gap increase with the growing system scale. We also see that the estimation performance decreases with the increasing SCR as expected, due to the decreasing amount of wireless resource.

VI. CONCLUSION

We have proposed a practical remote estimation system with the NOMA scheme. We have developed an advanced DRL algorithm for resource allocation with large hybrid state and action spaces. Our experiments have showcased that the proposed DRL algorithm is able to effectively address the resource allocation problem and provide significant performance gain than the benchmarks. For future work, we will investigate distributed DRL algorithms for resource allocation of large-scale systems and compare them with the present centralized allocation scheme.

REFERENCES

[1] P. Park, S. Coleri Ergen, C. Fischione, C. Lu, and K. H. Johansson, "Wireless network design for control systems: A survey," *IEEE Commun. Surv. Tutor.*, vol. 20, no. 2, pp. 978–1013, May 2018.

[2] E. Uysal *et al.*, "Semantic communications in networked systems: A data significance perspective," *arXiv preprint (accepted by IEEE Network)*, Mar. 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2103.05391

[3] A. Zappone, M. Di Renzo, and M. Debbah, "Wireless networks design in the era of deep learning: Model-based, AI-based, or both?" *IEEE Trans. Commun.*, vol. 67, no. 10, pp. 7331–7376, Jun. 2019.

[4] W. Liu, K. Huang, D. E. Quevedo, B. Vucetic, and Y. Li, "Deep reinforcement learning for wireless scheduling in distributed networked control," *arXiv preprint*, Sep. 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2109.12562

[5] A. S. Leong, A. Ramaswamy, D. E. Quevedo, H. Karl, and L. Shi, "Deep reinforcement learning for wireless sensor scheduling in cyber-physical systems," *Automatica*, vol. 113, pp. 1–8, Mar. 2020. Art. no. 108759.

[6] S. M. R. Islam, N. Avazov, O. A. Dobre, and K.-s. Kwak, "Power-domain non-orthogonal multiple access (NOMA) in 5G systems: Potentials and challenges," *IEEE Commun. Surv. Tutor.*, vol. 19, no. 2, pp. 721–742, Oct. 2016.

[7] Y. Li, A. Mehr, and T. Chen, "Multi-sensor transmission power control for remote estimation through a SINR-based communication channel," *Automatica*, vol. 101, pp. 78–86, Mar. 2019.

[8] A. Forootani, R. Iervolino, M. Tipaldi, and S. Dey, "Transmission scheduling for multi-process multi-sensor remote estimation via approximate dynamic programming," *Automatica*, vol. 136, pp. 1–14, Feb, 2022. Art. no. 110061.

[9] M. Pezzutto, L. Schenato, and S. Dey, "Transmission scheduling for remote estimation with multi-packet reception under multi-sensor interference," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 2628–2633, Apr. 2021.

[10] ——, "Transmission power allocation for remote estimation with multi-packet reception capabilities," *Automatica*, vol. 140, pp. 1–13, Jun. 2022. Art. no. 110257.

[11] W. Liu, D. E. Quevedo, Y. Li, K. H. Johansson, and B. Vucetic, "Remote state estimation with smart sensors over Markov fading channels," *IEEE Trans. Autom. Control*, early access, Jun. 2021, doi: 10.1109/TAC.2021.3090741.

[12] P. Sadeghi, R. A. Kennedy, P. B. Rapajic, and R. Shams, "Finite-state Markov modeling of fading channels - a survey of principles and applications," *IEEE Signal Process. Mag.*, vol. 25, no. 5, pp. 57–80, Sep. 2008.

[13] Y. He *et al.*, "Deep-reinforcement-learning-based optimization for cache-enabled opportunistic interference alignment wireless networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 11, pp. 10 433–10 445, Sep. 2017.

[14] Y. Polyanskiy, H. V. Poor, and S. Verdu, "Channel coding rate in the finite blocklength regime," *IEEE Trans. Inform. Theory*, vol. 56, no. 5, pp. 2307–2359, May 2010.

[15] W. Liu, G. Nair, Y. Li, D. Nesic, B. Vucetic, and H. V. Poor, "On the latency, rate, and reliability tradeoff in wireless networked control systems for IIoT," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 723–733, Jan. 2021.

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint*, Aug. 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1707.06347

[17] T. Yang, L. Zhao, W. Li, and A. Y. Zomaya, "Reinforcement learning in sustainable energy and electric systems: A survey," *Annu. Rev. Control*, vol. 49, pp. 145–163, Apr. 2020.

[18] M. Morales, *Grokking deep reinforcement learning*, 1st ed. USA: Manning Publications, 2020, ch. 11, pp. 342–343.

[19] Y. Xiao, J. Liu, J. Wu, and N. Ansari, "Leveraging deep reinforcement learning for traffic engineering: A survey," *IEEE Commun. Surv. Tutor.*, vol. 23, no. 4, pp. 2064–2097, Aug. 2021.