

Resource Allocation in Wireless Control Systems via Deep Policy Gradient

Vinicius Lima, Mark Eisen, Konstantinos Gatsis and Alejandro Ribeiro

Abstract—In wireless control systems, remote control of plants is achieved through closing of the control loop over a wireless channel. As wireless communication is noisy and subject to packet dropouts, proper allocation of limited resources, e.g. transmission power, across plants is critical for maintaining reliable operation. In this paper, we formulate the design of an optimal resource allocation policy that uses current plant states and wireless channel conditions to assign resources used to send control actuation information back to plants. While this problem is challenging due to its infinite dimensionality and need for explicit system model and state knowledge, we propose the use of deep reinforcement learning techniques to find data-driven resource allocation policies. In particular, we use model-free policy gradient methods to directly learn continuous power allocation policies without knowledge of plant dynamics or communication models. Numerical simulations demonstrate the strong performance of learned policies relative to baseline resource allocation methods.

I. INTRODUCTION

Wireless communication networks are frequently used to exchange data between plants, sensors and actuators in control systems. The use of wireless networks in lieu of wired communication makes the installation of components easier and maintenance more flexible, but also adds particular challenges to the design of control and communication policies [1], [2]. Wireless networks are in general more noisy than their wired counterparts [3], while reliable operation of control systems demands rapid communication and low message error rates — requirements in turn constrained by the limited resources available in that network. It is natural in this setting to look for an optimal way to distribute the resources available in the network among the plants sharing that communication medium.

In wireless networks, resource allocation aims to balance power consumption, latency and reliability while taking into account stochastic noise and rapid variability [4], [5]. That involves optimizing a performance metric over a function, resulting in an infinite dimensional problem that is usually hard to solve. The formulation of the resource allocation problem, however, resembles a statistical learning problem [5], which allows one to treat resource allocation in a data-driven fashion [6], [7]. When control plants share a common communication medium, resource allocation policies should also take into account the dynamics of each plant. For a

recent overview of issues and algorithms in network design of wireless control systems, we refer the reader to [2]; resource allocation and scheduling for control systems are also studied in [8]–[10], to name a few.

In this setting resources are allocated depending upon both the state of the control plants and fading conditions of the wireless channel — finding solutions invariably requires accurate model and state information. Recent advances in machine learning and the search for model-free allocation policies have motivated the use of data-driven approaches for scheduling [11]–[13]. In [11], [12], the authors propose a scheduling scheme based on the Deep Q-Network (DQN) algorithm. Value-based methods are adequate for discrete scheduling actions, but unsuitable for learning the continuous action spaces of resource allocation problems we consider in this paper. In [13] the authors explore actor-critic algorithms to learn communication and control policies in event-triggered wireless control systems. Such existing works, however, only consider discrete scheduling actions.

Here we propose the use of policy gradient methods to find optimal resource allocation policies for wireless control systems (WCS) under power constraints. We cast the resource allocation problem in WCSs as a reinforcement learning problem. This can be solved in continuous action spaces via policy gradient methods that respect wireless resource constraints. We further propose the use of neural networks to parameterize a policy that uses current plant and wireless fading state information to allocate wireless resources. Policy gradient methods allow us to model continuous allocation functions; hence our focus on this class of algorithms. Numerical results demonstrate the strong performance of such resource allocation policies over benchmark solutions.

II. RESOURCE ALLOCATION IN CONTROL SYSTEMS

Consider a system made up by m independent plants sharing a common wireless medium as in Figure 1. At each time instant a plant samples its state and sends this information to a common access point (AP) containing a remote controller. We assume the dynamics of each plant i can be approximately represented by a linear model affected by some random noise standing for eventual disturbances or unmodeled dynamics, such that

$$x_{t+1}^{(i)} = A^{(i)}x_t^{(i)} + B^{(i)}u_t^{(i)} + w_t^{(i)} \quad (1)$$

with the state vector $x^{(i)} \in \mathbb{R}^p$, control input $u^{(i)} \in \mathbb{R}^r$ and the random disturbance a Gaussian noise $w_t^{(i)} \in \mathbb{R}^p$ with covariance matrix $W^{(i)}$. We assume that the pairs $(A^{(i)}, B^{(i)})$ are controllable but $A^{(i)}$ is not necessarily stable.

Supported by Intel Science and Technology Center for Wireless Autonomous Systems and ARL DCIST CRA W9111NF-17-2-0181. V. Lima and A. Ribeiro are with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA. M. Eisen is with Intel Corporation, Hillsboro, OR. K. Gatsis is with the Department of Engineering Science, University of Oxford, Parks Road, Oxford, OX1 3PJ, UK.

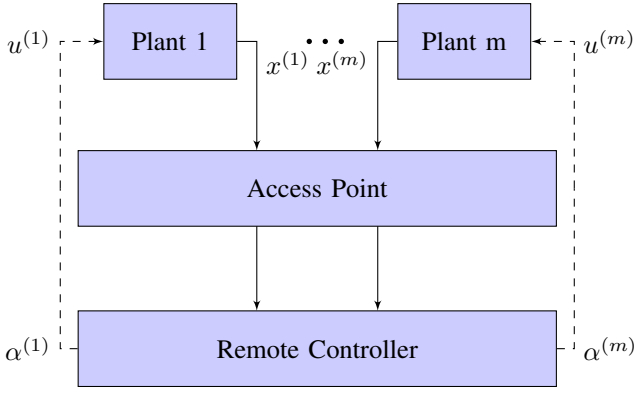


Fig. 1. Wireless control system architecture. Each plant i has state $x^{(i)}$ and its dynamics is independent of the others. The wireless communication network is made up by different channels with fading state $h^{(i)}$, and an access manager oversees the communication network. The plants transmit their current states to the access point, which should then distribute the resources available in the network to the corresponding communication channels. When the feedback loop is closed, the plant receives the control signal and executes the corresponding control action. When the feedback loop is not closed the plant relies on an estimate of the control signal instead.

In wireless control systems, the access point manages the access of each plant to the shared wireless medium. This medium is inherently noisy and prone to packet drops. When the plant is able to successfully receive the control signal, the feedback loop is closed, and the plant can execute the ensuing control action. To stress the importance of communication in this setting, we assume that the plant does not execute any control action when that transmission fails.

The reliability of the communications channel is dependent upon the resource or power level with which a plants sends its information as well as a random channel state known as wireless fading. Let then $h^{(i)} \in \mathcal{H} \subset \mathbb{R}_+$ a random variable drawn from a distribution $m(h)$ that represents the fading state experienced by plant i and denote by $\alpha^{(i)} \in \mathbb{R}_+$ the resource used by plant i . Further define a function $v : \mathbb{R}_+ \times \mathbb{R}_+ \rightarrow [0, 1]$ that, given a channel state and resource allocation, returns the probability of successfully receiving the packet. The system dynamics is then given by

$$x_{t+1}^{(i)} = \begin{cases} A^{(i)}x_t^{(i)} + B^{(i)}u_t^{(i)} + w_t & \text{w.p. } v(h_t^{(i)}, \alpha_t^{(i)}) \\ A^{(i)}x_t^{(i)} + w_t & \text{w.p. } 1 - v(h_t^{(i)}, \alpha_t^{(i)}) \end{cases} \quad (2)$$

As can be seen in (2), allocating more power to a plant will increase the reliability of the wireless channel, thus increasing probability of closing its control loop and experiencing more favorable dynamics. In most practical systems, however, we do not have unlimited power to allocate between the communication channels. The resource allocation problem consists of properly allocating resources available while keeping all plants in desirable states. We are interested in a resource allocation function $\alpha(h, x)$ that, given current channel conditions $h_t := [h_t^{(1)}; \dots; h_t^{(m)}]$ and plant states $x_t := [x_t^{(1)}; \dots; x_t^{(m)}]$, distributes resources among the plants without violating a maximum power constraint p_{max} . As current resource allocation decisions impact future states,

we consider as performance metric a quadratic cost of the plants states evaluated over a finite horizon T . The resource allocation problem takes the form

$$\begin{aligned} \min_{\alpha(h, x)} \mathbb{E}_{x_0}^{\alpha(h, x)} \left[\sum_{t=0}^T x_t^T Q x_t \right] \\ \text{s. t. } \alpha(h, x) \in \mathcal{A}; \mathcal{A} = \left\{ \alpha(h, x) : \sum_{i=1}^m \alpha^{(i)} \leq p_{max} \right\}, \end{aligned} \quad (3)$$

with $Q \geq 0$ and $\alpha^{(i)}$ the resource allocated to plant i . At each time t , the AP uses power $\alpha_t^{(i)} = [\alpha(h_t, x_t)]_i$ to send the control signal $u^{(i)}$ back to plant i . The communication exchange subsequently occurs with success rate given by $v(h_t^{(i)}, \alpha_t^{(i)})$ and plant i evolves via (2) accordingly.

In (3), the objective involves finding the resource allocation function $\alpha(x, h)$ that results in the minimum operation cost of the plants while satisfying the resource constraints. This results in an infinite-dimensional optimization problem that is generally hard to solve. Moreover, finding an optimal policy directly in (3) necessarily requires explicit knowledge of the plant dynamics and communication models in (2), which are often unavailable in practice. This search for model-free, data-driven policies motivate the use of reinforcement learning (RL) for resource allocation in WCSs.

III. RL FOR RESOURCE ALLOCATION

Reinforcement learning can be seen as a mathematical representation of the idea that learning comes from interaction with the environment [14]. RL problems are formalized with the use of Markov decision processes (MDP), a standard mathematical description of a sequential decision making process [14]. Formally, a MDP consists in a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P} \rangle$ with \mathcal{S} a set of states, \mathcal{A} a set of actions and $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ a state transition probability kernel that assigns to each triplet (s, a, s') the probability of moving from state s to s' if action a is chosen. A transition from a state s_t to s_{t+1} incurs a cost per stage r_t , and the agent takes actions according to a stochastic policy $\pi(a|s)$. That policy corresponds to a distribution that gives the probability of the agent to choose an action a when its current state is s . The agent's objective is to minimize some cumulative cost

$$J(\pi, s) := \mathbb{E}_s^\pi [R_t] = \mathbb{E}_s^\pi \left[\sum_{k=1}^T \gamma^k r_{k+t+1} \right] \quad (4)$$

starting from s and following $\pi(\cdot)$ with $\gamma \in (0, 1]$ a given discount factor [14], [15]. The value function is given by

$$J^*(s) = \inf_{\pi \in \Pi} J(\pi, s), s \in \mathcal{S}, \quad (5)$$

and we want to find the policy π^* achieving this minimum.

To cast the scheduling problem of section II in a reinforcement learning framework, we take the AP as a centralized agent. The actions here correspond to resource allocation vectors, defined on the corresponding action space \mathbb{R}_+^m and taken according to the allocation policy $\alpha(h, x)$. The possible

states of the system are made up by variables describing the state of the plants and channel conditions,

$$s_t = [h_t; x_t] = [h_t^{(1)}; \dots; h_t^{(m)}; x_t^{(1)}; \dots; x_t^{(m)}] \in \mathbb{R}^{m+m \times p}.$$

The performance of the resource allocation function is a quadratic cost in the control states, cf. (3). Furthermore we parameterize the resource allocation function $\alpha(h, x)$ with some stochastic policy $\pi(p|s; \theta)$ that is fully specified with a parameter vector $\theta \in \mathbb{R}^m$, i.e.

$$\alpha(h, x) = \pi(p|s; \theta). \quad (6)$$

Naturally, the above parameterization incurs a loss of optimality, but multilayer neural networks satisfy the so-called universal approximation theorem, meaning that these functions can arbitrarily approximate a continuous function when sufficiently large [16]. We can now rewrite the resource allocation problem in (3) as

$$\begin{aligned} \min_{\theta} \mathbb{E}_{x_0}^{\pi(h, x; \theta)} \left[\sum_{t=0}^T \gamma^t x_t^T Q x_t \right] \\ \pi(h, x; \theta) \in \mathcal{P}; \mathcal{P} = \left\{ \pi(h, x; \theta) : \sum_{i=1}^m \pi^{(i)} \leq p_{\max} \right\} \end{aligned} \quad (7)$$

Note that the cost function here will depend on the parameters θ , allowing us to take

$$J(\theta) = \mathbb{E}_{x_0}^{\pi(h, x; \theta)} \left[\sum_{t=0}^T \gamma^t x_t^T Q x_t \right]. \quad (8)$$

That is an example of a model-free reinforcement learning problem: the agent does not know the dynamical model of the control plants or the distribution of the communication channel states nor tries to learn it. Within the model-free RL framework, an immediate distinction is made between *value-based* methods, that is, methods that learn or estimate the action-value function and calculate the corresponding policy based on this approximation; and *policy-based* methods, where the algorithm learns a parameterized policy directly. Parameterizing a policy directly allows us to learn policies in continuous actions spaces as in the resource allocation problem considered here [14, ch. 13]. At each iteration, policy gradient methods perform approximate gradient descent in the cost function $J(\theta)$ [14, ch. 13],

$$\theta_{t+1} = \theta_t - \beta \nabla J(\hat{\theta}_t), \quad (9)$$

with β the learning rate and $\nabla J(\hat{\theta}_t)$ an estimate of the gradient of $J(\theta)$ with respect to θ . The policy gradient theorem gives a closed-form expression for the gradient $\nabla J(\theta)$ of the cost function $J(\theta)$ with respect to the parameter vector θ . Policy gradient algorithms then come up with strategies to sample the actions, states and rewards of the underlying MDP to approximate that policy gradient expression [14]. First we consider the REINFORCE algorithm, where the estimate will depend on the action a_t taken at time t [17], [18],

$$\theta_{t+1} = \theta_t - \beta \gamma^t R_t \frac{\nabla \pi(a_t|s_t; \theta_t)}{\pi(a_t|s_t; \theta_t)}. \quad (10)$$

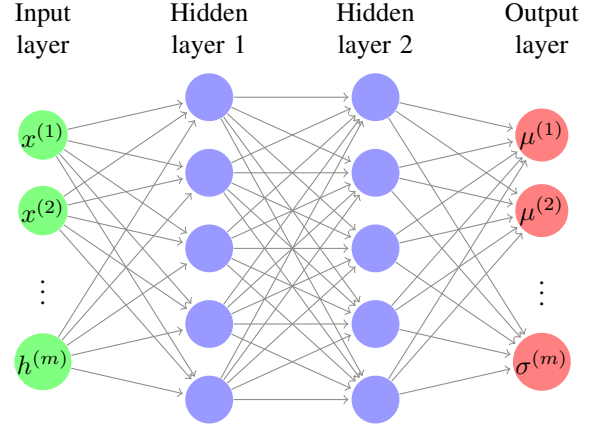


Fig. 2. Basic architecture of a neural network. Each node in the first hidden layer takes a linear combination of the nodes in the input layer and produces a nonlinear transformation of this linear combination. The nonlinear transformation is given by an activation function, usually a sigmoid or rectified linear function. The nodes in the second hidden layer take a linear combination of the outputs of the first hidden layer and once again produce a nonlinear transformation of that linear combination. The process is repeated until the output layer. Here the inputs correspond to the plant states and channel variables, and the output is a set of parameters used to characterize the allocation policy.

The equation shows that each update of the REINFORCE algorithm depends on the return R_t associated to the action a_t taken and the ratio between the gradient of the probability of executing that action and the probability of doing so [14]. Other policy gradient algorithms follow the same basic structure, although with different estimates of the gradient of the cost function. Note that the parameter update on the right-hand side of (10) takes into account only the states, actions and returns sampled from the environment; the algorithm does not need to know or to learn a model of the system.

Remark 1. For resource allocation problems, it is essential that we design policies that not only minimize the control cost in (8), but do so while satisfying the wireless resource constraints defined by \mathcal{P} in (7). In particular, we may select or design the stochastic policy distribution $\pi(p|s; \theta)$ to output resource allocation actions α such that $\sum \alpha_i \leq p_{\max}$ by construction. This amounts to outputting actions that are normalized, or belong to a $m - 1$ simplex, and scaling by p_{\max} . Natural choices for such a distribution include a Dirichlet distribution parameterized by θ or a series of independent Gaussian distributions parameterized by θ_i . The latter case would require some normalization procedure, such as softmax, to properly scale the power allocations.

Resource allocation functions such as the one we consider here do not necessarily have a known form. Thus we leverage neural networks to search for allocation policies within a larger class of nonlinear functions; cf. Figure 2. In this case, the neural network takes the plants states $x_t^{(1)}, \dots, x_t^{(m)}$ and channel variables $h_t^{(1)}, \dots, h_t^{(m)}$ as inputs, and outputs a set of parameters used to characterize a (multivariate) Gaussian policy with means $\mu^{(1)}, \dots, \mu^{(m)}$ and covariance matrices $\sigma^{(1)}, \dots, \sigma^{(m)}$. In a multilayer, fully connected neural net-

work, each element in the first hidden layer constructs a linear combination of the input elements and passes this combination through a nonlinear transformation or activation function $\phi(\cdot)$. Each element in the second hidden layer then performs a similar transformation on the elements of the first hidden layer, and the process is repeated up to the output layer. At each hidden layer l this generates the so-called hidden units [19, ch. 5]

$$z_l = \phi(C_l z_{l-1} + b_l). \quad (11)$$

Here the matrix C_l and the vector b_l represent the linear combination. The outputs $y^{(k)}$ of the neural network will then be given by [19]

$$y^{(k)}(x, h) = \phi(\phi(\dots \phi(C_1 z_0 + b_1)) + b_L) \quad (12)$$

with $z_0 = [x; h]$ and $y^{(L)}(x, h) = [\mu; \sigma]$, while d_l is the number of hidden units in hidden layer $l = 1, \dots, L$ and the parameter vector is given by $\theta = [C_1; b_1; \dots; C_L; b_L]$. We consider here ReLU activation functions, since they handle better issues such as vanishing gradients [20].

Algorithm 1: Deep PG for resource allocation in wireless control systems (based on [14])

Result: Resource allocation policy.

```

1 initialization: load initial training / parameter set  $\Theta$ 
  /* Loop over the episodes */
2 for  $ii = 1, \dots, N$  do
  /*  $T$ -step horizon simulation */
  generates complete episode:
3   $x_0, h_0, p_0, r_1, \dots, x_{T-1}, h_{T-1}, p_{T-1}, r_T$ 
4  while  $t < T$  do
    /* calculates cost-to-go */
5     $R_t \leftarrow \sum_{k=t+1}^T \gamma^{k-t-1} r_k$ 
    /* updates parameters */
6     $\theta \leftarrow \theta - \beta \gamma^t R_t \nabla \ln \pi(p_t | s_t; \theta)$ 
7  end
8 end
9 end

```

IV. NUMERICAL EXPERIMENTS

We now present numerical experiments to illustrate the use of the proposed learning-based approach. We consider the distribution of a power budget among a collection of unstable but controllable plants sharing a wireless communication network. The dynamics of each plant are determined by the matrices (2)

$$A^{(i)} = \begin{bmatrix} -1.1 & 0.5 & 0.2 \\ -0.2 & 1.1 & 0.3 \\ 0.4 & 0.5 & -1 \end{bmatrix}; \quad B^{(i)} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}, \quad (13)$$

unknown to the agent. The probability of successfully receiving an information packet is given by

$$v(h^{(i)}, \alpha^{(i)}) = 1 - \exp(-h^{(i)} \alpha^{(i)}(h, x)) \quad (14)$$

and the control policy $u(x_t)$ is a standard linear quadratic regulator. Here we consider a scenario where m plants share

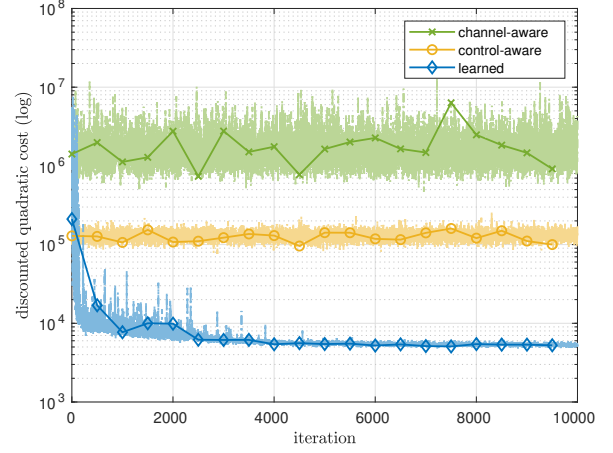


Fig. 3. Discounted cost (for a full episode) during learning phase. This simulation considers $m = 10$ unstable but controllable plants. Here the agent has access only to noisy observations of the plants and channel states.

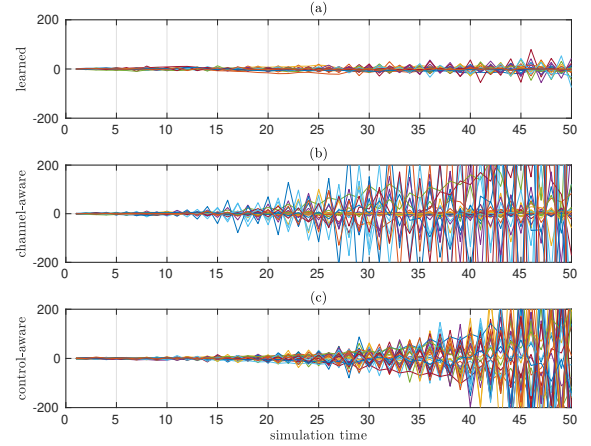


Fig. 4. Test phase example. Figure shows the evolution of the plant states for (a) the learned allocation, (b) channel-aware selection and (c) control-aware selection.

a power budget of $p_{\max} = \frac{m}{2}$. The channels fading states follow an exponential distribution with parameter $\lambda_h = 1$. The AP has access only to noisy estimates of the control and channel states, i.e. the allocation decision is taken based on

$$[\tilde{h}_t; \tilde{x}_t] = [h_t; x_t] + w_t^{(o)} \quad (15)$$

where the observation noise $w_t^{(o)}$ is a Gaussian disturbance with covariance $W^{(\text{obs})}$. Training was performed with a horizon $T = 30$ and test with $T = 50$. We used 60 samples per iteration, a learning rate $\beta = 5e-6$ and a neural network with two hidden layers with 400 and 300 units. The neural network was initialized with a supervised pre-learning phase to initially fit a heuristic that gives more power to plants further away from the equilibrium point. Figure 3 shows the training cost for this simulation, where we considered $W^{(\text{obs})} = 1$.

After training, the learned allocation policy was compared

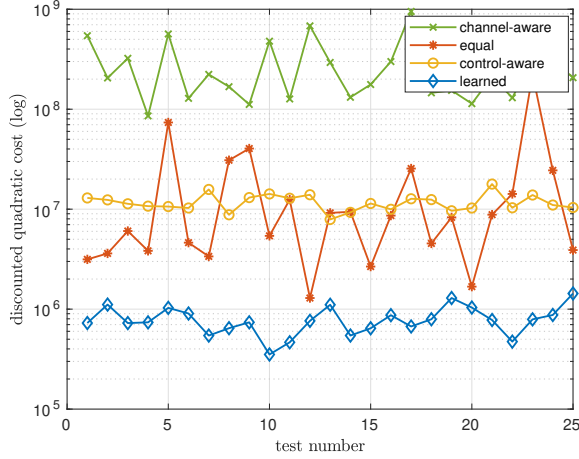


Fig. 5. Test comparison between the learned allocation (blue), a control-aware selection heuristic (yellow), a channel-aware selection heuristic (green) and random selection (purple). Here, $T = 50$ and $m = 10$. Each test point is an average over 60 simulations with initial states $x_0 \sim \mathcal{N}(0, 1)$.

against some baseline solutions:

- 1) dividing power equally among all plants.
- 2) channel-aware selection: choose $m/3$ plants with best channel conditions to transmit with power $\alpha_0 = \frac{p_{\max}}{m/3}$.
- 3) control-aware selection: choose $\frac{m}{3}$ plants furthest away from equilibrium to transmit with power $\alpha_0 = \frac{p_{\max}}{m/3}$.

For this comparison, we considered a larger horizon $T = 50$. Results are summarized in Figure 5. Each point in that plot amounts to an average of 60 simulations with same plant dynamics (unknown to the agent) and with initial states x_0 sampled from a standard normal distribution. Note that in this setting the learned policy outperforms all the baseline solutions mentioned above. Moreover, simulation results summarized in Figures 3 and 5 show that taking both plant states and channel conditions into account improves performance of the allocation policy.

V. CONCLUSION

This paper discusses a deep reinforcement learning approach for power allocation in wireless control systems. On the one hand, resource allocation problems are usually hard to solve, so it is natural to leverage heuristics to find an approximate allocation policy. Deep reinforcement learning algorithms, on the other, have achieved good results in traditional AI benchmarks, which, combined with their model-free learning capabilities and successful application in wireless systems, make it an attractive framework to handle resource allocation problems in wireless control systems where explicit system information is often unknown. Here, in particular, we made use of policy-based deep RL algorithms that allow us to learn continuous allocation policies based on current control and channel state information. Numerical results show that the proposed approach outperforms baseline solutions.

REFERENCES

- [1] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A Survey of Recent Results in Networked Control Systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 138–162, Jan. 2007.
- [2] P. Park, S. C. Ergen, C. Fischione, C. Lu, and K. H. Johansson, "Wireless Network Design for Control Systems: A Survey," *IEEE Communications Surveys Tutorials*, vol. 20, no. 2, pp. 978–1013, 2018.
- [3] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. S. Sastry, "Foundations of Control and Estimation Over Lossy Networks," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 163–187, Jan. 2007.
- [4] H. Fattah and C. Leung, "An overview of scheduling algorithms in wireless multimedia networks," *IEEE Wireless Communications*, vol. 9, no. 5, pp. 76–83, Oct. 2002.
- [5] A. Ribeiro, "Optimal resource allocation in wireless communication and networking," *EURASIP Journal on Wireless Communications and Networking*, vol. 2012, no. 1, p. 272, Aug. 2012.
- [6] M. Eisen, C. Zhang, L. F. O. Chamon, D. D. Lee, and A. Ribeiro, "Learning optimal resource allocations in wireless systems," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2775–2790, May 2019.
- [7] F. Liang, C. Shen, W. Yu, and F. Wu, "Towards Optimal Power Control via Ensembling Deep Neural Networks," *arXiv e-prints*, arXiv:1807.10025, arXiv:1807.10025, Jul. 2018. arXiv: 1807.10025 [eess.SP].
- [8] H. Rehder and M. Sanfridson, "Scheduling of a limited communication channel for optimal control," *Automatica*, vol. 40, no. 3, pp. 491–500, Mar. 2004.
- [9] K. Gatsis, M. Pajic, A. Ribeiro, and G. J. Pappas, "Opportunistic Control Over Shared Wireless Channels," *IEEE Transactions on Automatic Control*, vol. 60, no. 12, pp. 3140–3155, Dec. 2015.
- [10] T. Charalambous, A. Ozcelikkale, M. Zanon, P. Falcone, and H. Wymeersch, "On the resource allocation problem in wireless networked control systems," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, Dec. 2017, pp. 4147–4154.
- [11] B. Demirel, A. Ramaswamy, D. E. Quevedo, and H. Karl, "Deep-CAS: A Deep Reinforcement Learning Algorithm for Control-Aware Scheduling," *IEEE Control Systems Letters*, vol. 2, no. 4, pp. 737–742, Oct. 2018.
- [12] A. S. Leong, A. Ramaswamy, D. E. Quevedo, H. Karl, and L. Shi, "Deep Reinforcement Learning for Wireless Sensor Scheduling in Cyber-Physical Systems," *ArXiv e-prints*, Sep. 2018. arXiv: 1809.05149 [cs].
- [13] D. Baumann, J. Zhu, G. Martius, and S. Trimpe, "Deep reinforcement learning for event-triggered control," in *2018 IEEE Conference on Decision and Control (CDC)*, Dec. 2018, pp. 943–950.
- [14] R. S. Sutton and A. G. Barto, *Reinforcement Learning*, Second edition. The MIT Press, Nov. 2018.
- [15] O. Hernandez-Lerma and J. B. Lasserre, *Discrete-Time Markov Control Processes: Basic Optimality Criteria*, en, ser. Stochastic Modelling and Applied Probability. New York: Springer-Verlag, 1996.
- [16] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.
- [17] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3, pp. 229–256, May 1992.
- [18] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *In Advances in Neural Information Processing Systems 12*, MIT Press, 2000, pp. 1057–1063.
- [19] C. M. Bishop, "Chapter 5 - neural networks," in *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006.
- [20] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, G. Gordon, D. Dunson, and M. Dudík, Eds., ser. Proceedings of Machine Learning Research, vol. 15, Fort Lauderdale, FL, USA: PMLR, Nov. 2011, pp. 315–323.