Optimal Selective Transmission Policy for Energy-Harvesting Wireless Sensors via Monotone Neural Networks

Keyu Wu, Fudong Li, Chintha Tellambura, *Fellow, IEEE*, and Hai Jiang, *Senior Member, IEEE*

***Abstract*—We investigate the optimal transmission policy for an energy-harvesting wireless sensor node. The node must decide whether an arrived packet should be transmitted or dropped, based on the packet's priority, wireless channel gain, and the energy status of the node. The problem is formulated under the Markov decision process (MDP) framework. For such a problem, the conventional method to get the optimal policy is by using a state value function, which is three-dimensional in the considered problem, leading to high complexity. Fortunately, to reduce complexity, we derive an equivalent solution for the optimal policy via a one-dimensional after-state value function. We show that the after-state value function is differentiable and nondecreasing. We also discover a threshold structure of the optimal policy that is derived by the after-state value function. Furthermore, to approximate the after-state value function, we propose a learning algorithm to train a three-layer monotone neural network. The trained network thus finds a near-optimal selective transmission policy of the node. Finally, through simulation, we demonstrate the learning efficiency of the algorithm and the performance of the learned policy.**

***Index Terms*—After-state, data priority, energy harvesting, neural network, reinforcement learning (RL), wireless sensor networks (WSNs).**

I. Introduction

WIRELESS sensor networks (WSNs) have the ability to monitor environments, monitor/control industrial processes, and enable machine learning and data gathering, and many more [2], [3]. Ubiquitous deployment of WSNs is thus a key enabler of the Internet of Things [4]. The sensors must be autonomous with limited energy and computational resources. To achieve this, they may be powered by harvested energy [5] from ambient sources (wind, solar, and, others [6]). This enhances the energy self-sustainability of the nodes and consequently, the lifetime of the network is constrained by hardware limits, not by battery capacity [7]. Hence, energy harvesting improves network lifetime and enables a more sustainable evolution of WSNs.

Manuscript received December 30, 2018; revised May 31, 2019 and July 30, 2019; accepted August 3, 2019. Date of publication August 6, 2019; date of current version December 11, 2019. This work was supported by the Natural Sciences and Engineering Research Council of Canada. This paper was presented in part at the 2017 IEEE ICC (Paris, France) and was published in its Proceedings [1]. *(Corresponding author: Keyu Wu.)*

The authors are with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 1H9, Canada (e-mail: keyu2@ualberta.ca; fudong1@ualberta.ca; ct4@ualberta.ca; hai1@ualberta.ca).

Digital Object Identifier 10.1109/JIOT.2019.2933579

However, due to finite battery capacity and the randomness of harvested energy, energy depletion may happen. To avoid this, energy-saving approaches are needed, including transmission strategy design, sleep mode scheduling, transmission cooperation, energy-efficiency routing, and selective transmission (see Section II) [8]–[34]. In this article, we focus on selective transmission, in which the sender decides whether to send or to drop a packet. For example, selective transmission of data packets depending on their *priority* may be implemented. For instance, data packets of enemy attacks [35] or fire alarms [36] may have higher priority. So low-priority packets may be dropped when available energy is limited, which will allow the sensor to transmit more important packets in a long term. Such *selective transmission* strategies were studied in [8]–[13].

The works in [8] and [9] investigated selective transmission problems in conventional WSNs, i.e., sensors do not have energy harvesting ability. In [8], with the objective of maximizing the expected total priority of transmitted packets, a transmitting node, considering its available energy and the priority of a packet, decides whether or not to send the packet. The work in [9] extended the result of [8] by introducing a success index for each packet, which improves overall performance.

The works in [10]–[13] considered energy-harvesting enhanced WSNs. The work in [10] developed an optimal selective transmission policy by considering that the harvested energy amount is either zero or one, and the energy expense for each packet transmission is always one. Using a similar energy model, the work in [11] also developed an optimal transmission policy. Further, it proposed a low-complexity policy, which transmits packets whose priority are above a threshold. Work [12] extended the result of [11] to the case where there exists temporal correlation in the energy-harvesting process. The work in [13] used a learning approach to derive the optimal transmission policy, by modeling the harvested energy amount as a general random variable (r.v.).

A. Motivation, Problem Statement, and Contributions

The selective transmission decisions in [8]–[13] are made based on energy status and/or packet priority, but the effect of fading has not been considered in the decision making.

Since wireless data transmission is affected by channel fading, incorporating channel state information (CSI) into

decision making is necessary. Thus, a node can use CSI to adjust its transmission power to achieve reliable communication; or it may skip transmission to save energy if CSI indicates deep fading. Conversely, it may reduce transmit power if CSI indicates a good channel. These facts suggest that CSI inclusion in selective transmission could further improve energy efficiency, compared to that of the policies in [8]–[13].

Recall that a node obtains CSI via the feedback from the receiver. To realize this, the node first sends pilot signals to the receiver with the aim of performing channel estimation (see [37]–[39] and references therein for pilot designs). Because the pilot length is much shorter than that of the data packets [40], the energy required for channel estimation is much smaller compared to that required for data transmission. Moreover, for slow-fading quasi-static channels, the channel coherence time is rather long and hence channel estimation can be done less frequently.

With these motivations, we consider selective transmissions of a wireless sensor node, to optimize its energy usage by exploiting CSI. Transmission selection is based upon battery status, data priority, and fading status, whereas only the first two factors are considered in [8]–[13]. In addition, we assume that the node is unaware of the statistical distributions of the battery status, data priority, and fading status. This lack of the distribution information will affect the optimal transmission policy. Considering all the aforementioned challenges, we make the following contributions.

1) We model the selective transmission problem as a continuous-state Markov decision process (MDP) [41]. The optimal policy is derived from an after-state value function. This approach transforms the conventional three-dimensional control problem (i.e., on battery status, priority, and fading status) to a one-dimensional control problem (i.e., on the "after-state" battery status). As a result, control of transmission is greatly simplified.
2) The structural properties of the after-state value function and the optimal policy are analyzed. We prove that the after-state value function is differentiable and nondecreasing, and that the optimal policy is threshold-based with respect to data priority and channel status.
3) To compute the continuous after-state value function, we find a parameterized representation. The parameters are learned from a batch of data samples. To build this representation of after-state value function, we propose to train a monotone neural network (MNN) [42], and we prove that MNN preserves the necessary properties of the after-state value function, which is differentiable and nondecreasing.
4) We develop a learning algorithm to train the proposed MNN. The learning process exploits data samples (but not the distributional information, which is unknown to the node). The trained MNN can construct a near-optimal transmission policy. With simulations, we demonstrate the learning efficiency of the proposed algorithm, and also the performance achieved by the learned policy.

This article is the extended, comprehensive journal version of our previous conference paper [1]. The conference paper omits the proofs of all theorems, which are provided here. Further, this journal paper adds explanations, insights, and numerical results. Other differences of the conference and journal papers are as follows.

1) A more practical model is adopted herein. In [1], we assumed that the node expends energy for data transmissions only. In contrast, this journal paper includes more practical concerns of energy expenses from idling, data reception, and channel probing operations of the node. This makes the development and analysis of optimal policy considerably more challenging than those in [1].
2) A more efficient learning method is used in this journal paper. For both [1] and this article, we need to learn a differentiable nondecreasing function, which yields the optimal policy. In [1], this function was represented by a polynomial approximation. However, the polynomial approximation does not possess the nondecreasing property, and hence the developed learning algorithm is inefficient. In this journal paper, we propose the use of an MNN, and prove that it is a well-designed approximation for differentiable nondecreasing functions. Moreover, we also develop a learning algorithm for training the MNN, which is not available in [1].

The rest of this article is organized as follows. Section II discusses existing energy-saving approaches in WSNs. Section III presents the considered system model and formulates the selective transmission control problem. Section IV derives and analyzes the optimal transmission policy based on an after-state value function. Section V proposes an MNN to approximate the after-state value function, and develops a learning algorithm to train the proposed MNN. Section VI provides simulation results of the proposed algorithm and learned policy. Section VII concludes this article.

II. Related Works

As wireless sensors are generally energy constrained, energy management is crucial for WSNs. A broad range of energy-saving approaches has been considered in the literature to ensure WSNs' proper and efficient operations. These approaches can be classified into five categories: 1) transmission strategy design; 2) sleep mode scheduling; 3) transmission cooperation; 4) energy-efficient routing; and 5) selective transmission. In what follows, we briefly discuss problem setups and representative results of the first four categories, and then we discuss in details the category of selective transmission, which is also the topic of this article.

A. Transmission Strategy Design

The transmission strategy, which decides on transmission power, transmission rate, and/or modulation scheme, strongly affects communication reliability and efficiency. Therefore, designing a transmission strategy from an energy-efficient perspective is crucial for WSNs, which is studied extensively in the literature. For example, the work in [14] investigated a WSN-WiFi hybrid network, and showed that longer packets lead to larger transmission delay, while shorter packets cause more overhead and energy consumption. Therefore, in

order to minimize consumed energy while satisfying certain delay constraint, transmission packet size is optimized in [14]. Moreover, the work in [15] extended the result of [14] and studied a data rate adaption problem. Specifically, it is shown in [16] that transmitting with lower rate is generally more energy efficient (given the same amount of data), but requires longer transmission time. Hence, the work in [15] proposed a transmission strategy with adaptive rate, by which the tradeoff between energy efficiency and transmission delay is balanced. In [17], a similar energy management problem is studied for an energy-harvesting wireless link, where data rate of a single node is maximized under the energy causality constraint, i.e., at each instant, the consumed energy cannot exceed total harvested energy. The works in [18]–[20] studied an uplink transmission control problem in an energy-harvesting wireless network, where, based on channel condition and energy status, the network needs to decide the transmission power of each node in order to maximize overall throughput. To solve this problem, the work in [18] proposed a centralized scheme, which learns the optimal control strategy via a deep learning technique. The works in [19] and [20] considered a distributed scheme, where each node learns its transmission strategy with multi-agent reinforcement learning (RL) algorithms. The distributed scheme has similar performance to that of the centralized scheme. The work in [21] considered an uplink optimization scenario based on user selection. Taking energy status and fading profile into account, total data transmission rate is maximized by selecting a subset of the users. The works in [22]–[24] studied joint control of data queue and energy battery with specific constraints, where model predictive control methods are used, which have the ability to predict and exploit future harvested energy in control processes.

B. Sleep Mode Scheduling

In many scenarios of WSNs, packets arrive infrequently. Sleep mode scheduling exploits this observation, and switches some sensors into sleep mode (i.e., turns off the sensors' transceivers and other energy-consuming modules) for energy saving. However, when sensors are not scheduled carefully, transmission delay or packet loss probability may increase significantly. For example, the work in [25] found that different scheduling algorithms vary significantly in terms of worst-case transmission delay. Furthermore, a "multi-parent" scheduling scheme is proposed in [25] to reduce the transmission delay of the algorithm. The key idea is that rather than using a single-path topology for packet routing, where a sensor can only deliver packets to a fixed next-hop node, the proposed scheme ensures that an active node has multiple next-hop neighbors. And whenever the active node needs to deliver a packet, it delivers the packet to the next-hop neighbor with the earliest awaken time.

Sleep mode scheduling may also result in frequent state transitions between active and sleep modes. Due to hardware limitations, these transitions can cause non-negligible energy consumption [26]. A scheduling algorithm was proposed in [26], which assigns a sensor consecutive time slots, and ensures that neighboring sensors' time slots are overlapped properly, thus minimizing the number of state transitions and improving energy efficiency.

In [25] and [26], time is divided to cycles, each consisting of a number of time slots. Each node has its own scheduling pattern in each cycle, i.e., what time slots in the cycle are in active mode and what time slots are in sleep mode? The node repeats its scheduling pattern over different cycles. The works in [25] and [26] require time slot synchronization and cycle synchronization, referred to as *synchronized setting*. On the other hand, a nonsynchronized and cycling-free setting is investigated in [27], which eliminates time synchronization. Each sensor autonomously decides its working mode at each time slot with the goal of balancing between saving energy and transmitting packets. To achieve this, the sensor uses an RL algorithm to learn its decision policy distributively, from its own state dynamics and its interactions with other sensors.

C. Transmission Cooperation

In WSNs, spatially separated sensors generally experience independent wireless channels, i.e., spatial diversity, which can be exploited to improve transmission reliability and energy efficiency. For example, the work in [28] considered a cooperative transmission setup, where a sensor can either directly send data to its destination, or through a intermediate relay node that is selected from a set of candidates. Due to spatial diversity, it is possible to select a relay such that the two-hop cooperative link (from the source to the relay and finally to the destination) has better quality than that of the direct link. The work in [29] studied a relay communication scheme in energy-harvesting WSNs, where the power allocation of the first hop (from the source to the relay) and second hop (from the relay to the destination) are optimized under the energy availability constraint.

The work in [30] considered a different cooperation setup, where a set of sensors sense a certain event and send measurements to a fusion center. Based on sensors' observations, the fusion center decides whether or not the targeted event has occurred. Rather than transmitting all measurements to the fusion center, an ordered transmission scheme is proposed in [30], where sensors sequentially send data (the sending order is based on the quality of the sensors' sensed data). The transmission process is stopped once the data fusion center has received enough data for decision making. This scheme can reduce the number of needed transmissions and improve the network's energy efficiency.

D. Energy-Efficient Routing

Routing selects a path for delivering a packet from the source to the destination. Conventionally, this is done by selecting a path with minimum topology distance or transmission delay. However, since WSNs need to prolong the network lifetime, energy efficiency in routing design is critical.

For example, the work in [31] proposed a routing algorithm for energy-harvesting WSNs. Specifically, the routing metric combines the residual energy of a node, energy-harvesting rate, and energy cost for transmission. Therefore, this algorithm

tends to select a path that has the least energy "reduction" of the network, thus improving energy efficiency. More comprehensive construction of routing metrics has also been considered [32], [33]. Specifically, the work in [32] selects a routing path considering energy-harvesting wastage due to the overcharge of finite-capacity batteries, while the work in [33] considers the required time to recover from an energy deficient status. The work in [34] jointly considers packet routing and transmission power control in a multihop energy-harvesting WSN. The total network utility is maximized by deciding on the traffic amount for each source-destination pair, routing and transmission power allocation strategies in the network, under the data queue stability and energy availability constraints.

E. Selective Transmission

Selective transmission, which is the topic of this article, considers the case where a sensor's arriving packets are associated with different priorities. When a packet arrives, the sensor must decide either to send or drop the packet.

The works in [8] and [9] investigated selective transmission problems in conventional WSNs, i.e., sensors do not have energy harvesting ability. In [8], with the objective of maximizing the expected total priority of transmitted packets, a transmitting node, considering its available energy and the priority of a packet, decides whether or not to send the packet when the packet arrives. The work in [9] extended the result of [8] via further considering a success index, which is a measure of the likelihood that a transmitted packet reaches its destination, e.g., a sink node. Therefore, when making its transmission decision, each sensor takes decisions of other sensors into consideration through the use of the success indices, which may improve overall performance. Nevertheless, the use of success indices introduces communication overhead and additional energy consumption, as the success index for each packet has to be passed from the sink node to all sensors along the packet's routing path.

Different from [8] and [9], the works in [10]–[13] considered energy-harvesting enhanced WSNs, which poses additional challenges of modeling and dealing with random replenishment of energy. In [10], the harvested energy of a sensor is modeled as a Bernoulli r.v. taking values of 0 or 1, and the energy expense for each packet transmission is always defined as one. Accordingly, the sensor's battery dynamic is analyzed by the Markov chain theory, which is then used to develop an optimal transmission policy. Using a similar energy model to that in [10], the work in [11] also develops an optimal transmission policy. In addition, a low-complexity balanced policy (BP) is proposed: if the priority of a packet exceeds a predefined threshold, it is transmitted. BP is designed to ensure that the expected energy consumption equals the expected energy-harvesting gain, leading to energy neutrality. This ensures energy use efficiency while reducing energy outage risks. The work in [12] extended the result of [11] to the case where there exists temporal correlation in the energy-harvesting process.

However, in order to find optimal and/or heuristic policies, the statistical distributions of data priority and/or energy-harvesting process are needed in [8]–[12]. In addition, works [10]–[12] assumed one unit of energy for both energy replenishment and energy consumption, which may not be practical. These two limitations are resolved in [13]. Specifically, harvested energy is modeled as a general r.v. in [13]. In addition, based on the Robbins–Monro algorithms [43], the sender learns the optimal transmission policy from the observed data, without their statistical distributions.

Different from [8]–[13], where the selective transmission decision is made based on energy status and packet priority, this article further considers the exploitation of CSI to improve performance. In addition, a neural network and corresponding training algorithm are designed to efficiently learn the optimal transmission policy, given the data samples from the underlying stochastic environment (with no need for distribution information).

III. System Model and Problem Formulation

In this section, we will consider a selective transmission problem for an energy harvesting (EH) wireless link, which is briefly introduced in Section III-A. Then, the system model is elaborated from the aspects of operation cycles, states and actions, state dynamics, and rewards in Sections III-B–III-E, respectively. Finally, given the system model, the problem of finding the optimal transmission policy is formulated in Section III-F.

A. System Overview

We consider a single link with one wireless sensor node (transmitter) and its receiver. Throughout this article, this sensor node is simply called "the node." The node works in a cyclical manner. At each cycle, it receives a data packet and selects to transmit or discard it depending on the current system state, including the priority of the packet, energy status of the node, and CSI. Via intelligently dropping low-priority packets, the node aims to use its energy efficiently and transmit more high-priority packets in the long term. The details of operation cycle, system state, and the selective transmission scheme are given next.

B. Operation Cycles

Time is partitioned into cycles of random durations (Fig. 1). A cycle (say cycle t) begins with a silent period, in which the node waits until a data packet arrives. When that occurs, the silent period ends and an active period starts. During this active period, the node receives and decodes the data packet, estimates the channel CSI, and then it decides whether to transmit the received packet or discard it. After the packet is transmitted or discarded, cycle t ends and the silent period of cycle $t+1$ starts. At the same time, the node obtains an energy replenishment of e_t, which was harvested during cycle t. Note that the duration of a cycle can be random, but is assumed to be long enough to perform necessary tasks in an active period, i.e., data reception, channel estimation (which includes sending pilot signal to the receiver and receiving estimated CSI

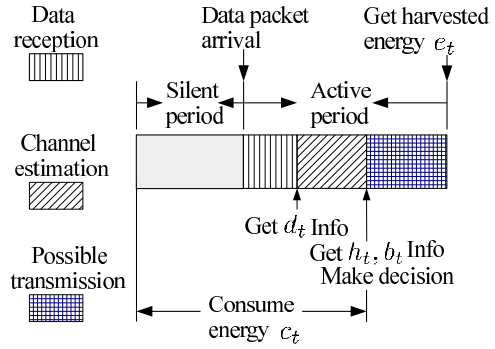


Fig. 1. Cycle structure.

from the receiver's feedback), and possible transmission. This assumption is reasonable due to the following reason. In our system model, the duration of a cycle is actually determined by the interval of two successive packet arrivals. As a WSN usually has light traffic load, it is reasonable to assume that the duration of a cycle is long enough to perform necessary tasks in an active period.

C. States and Actions

In the active period of cycle t, the node makes a transmission decision based on state $s_t = [b_t, h_t, d_t]$, where b_t is the remaining energy at the moment of decision making, h_t is the energy needed for transmission, and d_t is the packet priority. These quantities are detailed below.

1) The node receives and decodes a data packet ("data reception" in Fig. 1). It is assumed that the node is able to evaluate the priority d_t of the packet via, for example, reading the packet contents. Here, a higher priority value d_t means more importance.
2) The node sends a pilot signal to the receiver, and obtains information of the channel power gain z_t (CSI) from the receiver's feedback ("channel estimation" in Fig. 1). Based on z_t, the node estimates the required transmit energy h_t according to the full channel inversion power control scheme [44], which ensures a certain target signal power at the receiver. Without loss of generality, we assume that the targeted receiving power is one unit, and the transmission duration is also one unit. Thus, the required energy for transmission can be given as $h_t = 1/z_t$.
3) $b_t \in [0, 1]$ represents the remaining energy in the node's battery after the energy expenditure (denoted as c_t) in cycle t for standing by (in the silent period of cycle t), data reception and channel estimation (in the active period of cycle t). In other words, b_t is the remaining energy in the battery at the end of channel estimation in cycle t. Note that the battery's energy capacity is set to be one unit.

At the end of channel estimation in cycle t, the node has information of $s_t = [b_t, h_t, d_t]$, and at this moment, it needs to decide whether to transmit or discard the packet. The decision variable $a_t = 1$ represents "transmit" and $a_t = 0$ represents "discard." If $a_t = 0$, the packet is dropped with zero energy consumption for transmission. On the other hand, if $a_t = 1$ is chosen, we have the following.

1) If energy is sufficient ($b_t \geq h_t$), the node consumes energy h_t to transmit the packet, and consequently the packet will be delivered successfully.
2) If the energy is not sufficient, packet delivery fails, and the remaining energy is exhausted, i.e., the energy consumption for transmission is b_t.

D. State Dynamics

Here we model the relationship between s_{t+1} and (s_t, a_t). We assume that $\{h_t\}_t$ are independent and identically distributed (i.i.d.) continuous r.v.s with a probability density function (PDF) $f_H(x)$. Similarly, $\{d_t\}_t$ are i.i.d. continuous r.v.s with PDF $f_D(x)$. Therefore, h_{t+1} and d_{t+1} are independent of (s_t, a_t).

However, b_{t+1} is affected by (s_t, a_t), since different combinations of (b_t, h_t, a_t) cause different energy consumptions (Section III-C). Moreover, b_{t+1} also depends on e_t. Further, during cycle $t + 1$, the waiting in the silent period, and the data reception and channel estimation in the active period all consume energy, whose total amount is denoted as c_{t+1}. Therefore, b_{t+1} is affected by c_{t+1}. In summary, we have

$$b_{t+1} = \left((\varrho(s_t, a_t) + e_t)^- - c_{t+1}\right)^+ \tag{1}$$

where $(x)^- \triangleq \min\{x, 1\}$, $(x)^+ \triangleq \max\{x, 0\}$, and

$$\varrho(s_t, a_t) = (b_t - h_t \cdot a_t)^+. \tag{2}$$

We assume that $\{e_t\}_t$ and $\{c_t\}_t$ are, respectively, i.i.d. continuous r.v.s with PDF $f_E(x)$ and PDF $f_C(x)$. In Lemma 1 of Section IV-C, we will show that, given the value of $\varrho(s_t, a_t)$, b_{t+1} is a time-independent continuous r.v., i.e., its conditional PDF can be written as $f_B(\cdot|\varrho(s_t, a_t))$.

Therefore, given state s and action a at current cycle, the state $s' = [b', h', d']$ at next cycle can be characterized by the following conditional PDF, named as state transition kernel:

$$f(s'|s, a) = f_H(h') \cdot f_D(d') \cdot f_B(b'|\varrho(s, a)). \tag{3}$$

In the sequel, we use $(\cdot)'$ to denote a variable in the next cycle.

E. Rewards

At cycle t, a packet is successfully transmitted, if and only if $a_t = 1$ and $b_t \geq h_t$. Also considering that the packet's priority is quantified by d_t, the immediate reward of deciding on action a_t in presence of state s_t is defined as

$$r(s_t, a_t) \triangleq \mathbb{1}(a_t = 1) \cdot \mathbb{1}(b_t \geq h_t) \cdot d_t \tag{4}$$

where $\mathbb{1}(\cdot)$ is an indicator function.

F. Problem Formulation

A policy is designed to maximize the expected total rewards within infinite duration. We consider only the set of all deterministic stationary policies, denoted as Π. A deterministic stationary policy $\pi \in \Pi$ is a time-independent mapping from states to actions, i.e., $\pi : \mathbb{S} \mapsto \mathbb{A}$, where $\mathbb{S} = \{s =$

$[b, h, d]|b \in [0, 1], h \in \mathbb{R}_+, d \in \mathbb{R}_+\}$ denotes the state space, and $\mathbb{A} = \{0, 1\}$ denotes the action space.

Since the node continuously harvests energy from the environment, potentially over an infinite number of cycles, the total rewards can be infinite. To avoid this, discounting is perhaps the most analytically tractable and most widely studied approach. A discounting factor $\gamma \in [0, 1]$ ensures that the infinite summation is bounded. If $\gamma \to 0$, we pay more attention to immediate reward. If $\gamma \approx 1$, a long-term reward is taken into consideration.

For each policy π, the objective value obtained following policy π is defined as:

$$V^{\pi} = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, \pi(s_t))\right] \tag{5}$$

where expectation $\mathbb{E}[\cdot]$ is defined over the distribution of initial state s_0 and state trajectory $\{s_t\}_{t=1}^{\infty}$ induced by the sequence of actions $\{\pi(s_t)\}_{t=0}^{\infty}$. Note that if $\gamma \approx 1$, in our problem, V^{π} can be (approximately) interpreted as the expected sum of the priority of sent packets by policy π.

Our target is to solve an optimal policy π^* such that

$$\pi^* = \arg\sup_{\pi \in \Pi} \{V^{\pi}\}. \tag{6}$$

The optimal policy π^* tells the node about optimal transmission decision at each cycle t. In addition, we assume that the node does not know the PDFs of the battery status ($f_B(\cdot)$), data priority ($f_D(\cdot)$), or fading status ($f_H(\cdot)$). Thus, the solution of π^* must involve samples of the corresponding r.v.s.

IV. Optimal Selective Transmission Policy

A. Standard Results From MDP Theory

The 4-tuple $<\mathbb{S}, \mathbb{A}, r, f>$, namely, the state space, action space, reward function, and state transition kernel, defines an MDP. From basic MDP theory [41, Ch. 6], policy π^* (6) can be constructed from state-value function $V^* : \mathbb{S} \mapsto \mathbb{R}$ as

$$\pi^*(s) = \arg\max_a \{r(s, a) + \gamma \cdot \mathbb{E}[V^*(s')|s, a]\} \tag{7}$$

where the expectation is taken over the next state s' given current s and a. In addition, V^* is a solution to the Bellman equation

$$V(s) = \max_a \{r(s, a) + \gamma \cdot \mathbb{E}[V(s')|s, a]\}. \tag{8}$$

Finally, V^* can be computed recursively[1] by using (8).

Remark: Although V^* can be solved via (8), it is hard to compute π^* via (7). Specifically, (7) requires a conditional expectation over a random next state s', which is a computationally expensive task. We thus address this difficulty through a reformulation based on the after-state value function.

[1]This is known as the value iteration algorithm. Section IV-B provides an example of using it to compute the after-state value function. V^* can be similarly computed.

B. Reformulation Based on After-State Value Function

An after-state (also known as post-decision state), which is an intermediate variable between two successive states, can be used to simplify the optimal control of certain MDPs [17], [45]–[50]. The physical interpretation of after-state is problem dependent.

We next define the after-state variable for our problem. We also show that π^* can be defined over an after-state value function, which can be solved by a value iteration algorithm.

Physically, an after-state p_t of cycle t is the remaining energy after action a_t is performed but before harvested energy e_t is stored in the battery. Therefore, given state s_t and action a_t, the after-state is $p_t = \varrho(s_t, a_t)$. Recall that $\varrho(s_t, a_t) = (b_t - h_t \cdot a_t)^+$ [as defined in (2)]. Hence, deriving from (3), the conditional PDF of state $s' = [b', h', d']$ of next cycle given after-state p at current cycle is

$$q(s'|p) \triangleq f_H(h') \cdot f_D(d') \cdot f_B(b'|p). \tag{9}$$

Hence, the term $\mathbb{E}[V^*(s')|s, a]$ inside (7) and (8), where the conditional expectation is defined with PDF (3), can be written as $\mathbb{E}[V^*(s')|\varrho(s, a)]$ whose expectation is defined with PDF (9) with $p = \varrho(s, a)$. Keeping this observation in mind, π^* is redefined as follows.

We define the after-state value function $J^* : [0, 1] \mapsto \mathbb{R}$ as

$$J^*(p) = \gamma \mathbb{E}[V^*(s')|p]. \tag{10}$$

Plugging (10) into (7), we have

$$\pi^*(s) = \arg\max_a \{r(s, a) + J^*(\varrho(s, a))\}. \tag{11}$$

Therefore, (11) provides an alternative formulation of the optimal policy. We next present a value iteration algorithm to solve J^*.

Plugging (10) into (8), we have $V^*(s) = \max_a\{r(s, a) + J^*(\varrho(s, a))\}$. By replacing a with a', replacing s with s' and taking (γ-weighted) conditional expectation $\gamma \cdot \mathbb{E}[\cdot|p]$ on both sides, we further have $\gamma \cdot \mathbb{E}[V^*(s')|p] = \gamma \cdot \mathbb{E}[\max_{a'}\{r(s', a') + J^*(\varrho(s', a'))\}|p]$. Noticing that $\gamma \cdot \mathbb{E}[V^*(s')|p]$ on the left hand side is exactly the definition of $J^*(p)$, we have that J^* satisfies the following equation:

$$J^*(p) = \gamma \cdot \mathbb{E}\left[\max_{a'} \{r(s', a') + J^*(\varrho(s', a'))\}|p\right]. \tag{12}$$

Finally, following a similar procedure to [50, Th. 1], J^* can be solved by a value iteration algorithm (with the technical assumption that r.v. d_t has finite mean). Specifically, initially with a bounded function J_0, the sequence of functions $\{J_k\}_{k=1}^{K}$ computed via, $\forall p \in [0, 1]$

$$J_{k+1}(p) \leftarrow \gamma \cdot \mathbb{E}\left[\max_{a'} \{r(s', a') + J_k(\varrho(s', a'))\}\middle| p\right] \tag{13}$$

converges to J^* when $K \to \infty$.

Remark: Unlike (7), which requires conditional expectation for optimal decisions, (11) shows that the optimal decisions can be directly made with J^* (without taking expectation).

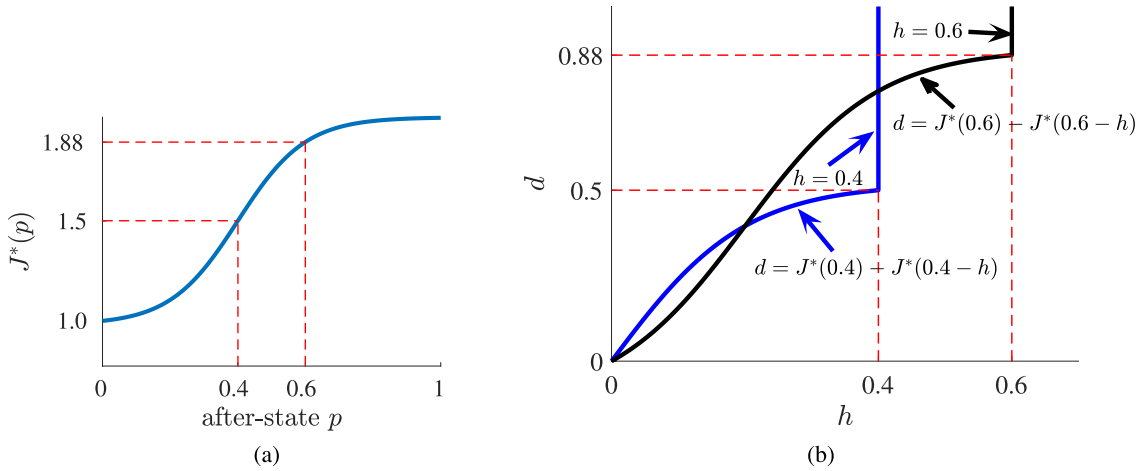


Fig. 2. After-state value function and optimal policy. (a) J^* example. (b) π^* boundaries for $b = 0.4$ and $b = 0.6$.

C. Properties of J^* and π^*

This section shows the properties of J^* and π^*. We begin with Lemma 1, whose proof is provided in Appendix A.

Lemma 1: Given that $p_t = p$, b_{t+1} is a continuous r.v. whose distribution does not depend on t. In addition, denoting its conditional cumulative distribution function as $F_B(b|p)$, we have $F_B(b|p_1) \leq F_B(b|p_2)$, if $p_1 \geq p_2$. Finally, $F_B(b|p)$ is differentiable with respect to p.

The after-state value function J^* can be theoretically derived from the value iteration algorithm (13). The results of Lemma 1 provide us a tool to analyze the conditional expectation operation $\mathbb{E}[\cdot|p]$ in (13). Via exploiting Lemma 1, Theorem 1 analyzes the structure of J^* with (13). The proof is provided in Appendix B,

Theorem 1: The after-state value function J^* is a differentiable and nondecreasing function with respect to after-state p.

Note that π^* can be defined via J^* as shown in (11). Therefore, Theorem 1 can be used to analyze the structures of π^*, as shown in Theorem 2.

Theorem 2: The optimal policy π^* has the following structure:

$$\pi^*([b,h,d]) = \begin{cases} 1 & \text{if } b \geq h \text{ and } d \geq J^*(b) - J^*(b-h) \\ 0 & \text{otherwise.} \end{cases} \tag{14}$$

Proof: From (11), we know that, $\pi^*([b,h,d]) = 1$ is equivalent to

$$\mathbb{1}(b \geq h) \cdot d + J^*\big((b-h)^+\big) \geq J^*(b). \tag{15}$$

Furthermore, (15) requires $b \geq h$, since otherwise we have $J^*(0) > J^*(b)$, which cannot hold as J^* is nondecreasing. Therefore, (15) is equivalent to $b \geq h$ and $d \geq J^*(b) - J^*(b-h)$. ■

Corollary 1: The optimal policy π^* is threshold-based nondecreasing with respect to d and $-h$. To be specific: 1) given any b and h, if $\pi^*([b,h,d_1]) = 1$, then $\pi^*([b,h,d_2]) = 1$, for any $d_2 \geq d_1$; and 2) given any b and d, if $\pi^*([b,h_1,d]) = 1$, then $\pi^*([b,h_2,d]) = 1$, for any $h_2 \leq h_1$.

Proof: From Theorem 2, $\pi^*([b,h,d_1]) = 1$ implies $h \leq b$ and $d_1 \geq J^*(b) - J^*(b-h)$. Therefore, we have $d_2 > J^*(b) - J^*(b-h)$ for any $d_2 \geq d_1$, which implies $\pi^*([b,h,d_2]) = 1$.

Similarly, $\pi([b,h_1,d]) = 1$ implies $h_1 \leq b$ and $d > J^*(b) - J^*(b-h_1)$. And because J^* is nondecreasing, we have $h_2 \leq b$ and $d > J^*(b) - J^*(b-h_2)$ for any $h_2 \leq h_1$, which implies $\pi^*([b,h_2,d]) = 1$. ■

Remark: Corollary 1 states that, for a given battery level b, the optimal policy is to send, if the data priority and channel quality exceed certain thresholds.

D. Example of J^* and π^*

We now present an example of J^* and π^* in Fig. 2.

$J^*(p)$ is the function shown in Fig. 2(a), which is nondecreasing and differentiable (Theorem 1). Based on $J^*(p)$, the optimal policy $\pi^*([b,h,d])$ is then determined based on (11). From Theorem 2, we know that, in the (h,d) space given battery level b, a decision boundary consisting of curve "$d = J^*(b) - J^*(b-h)$" and line "$h = b$" partitions the (h,d) space into two subspaces: in the subspace on the upper-left side of the boundary, the decision is $\pi^*([b,h,d]) = 1$; in the subspace on the bottom-right side of the boundary, the decision is $\pi^*([b,h,d]) = 0$. In Fig. 2(b), we show two examples for decision boundaries with $b = 0.4$ and $b = 0.6$, respectively. It is easily seen that $\pi^*([0.4,h,d])$ and $\pi^*([0.6,h,d])$ are threshold-based nondecreasing with respect to d and $-h$, as proved in Corollary 1. However, the threshold structure does not hold in dimension b. As one can see in Fig. 2(b), there is an area of (h,d) that $a = 1$ is chosen with $\pi^*([0.4,h,d])$, but $a = 0$ is chosen with $\pi^*([0.6,h,d])$.

V. Neural Network for Optimal Control

Section IV-B shows that π^* can be effectively constructed by J^*, which, in turn, can be solved by the value iteration algorithm (13). However, the implementation of (13) has two difficulties. First, as the PDFs $f_H(\cdot)$, $f_D(\cdot)$ and $f_B(\cdot|p)$ are not available, we cannot compute $\mathbb{E}[\cdot|p]$. Second, because after-state p is a continuous variable over $[0,1]$, each iteration of (13) has to be computed over infinitely many p values.

RL provides a useful solution to address both difficulties. Specifically, instead of exactly solving J^*, RL targets an approximation of J^* via learning a parameter vector (a set of real values), while the learning process exploits data samples (rather than underlying distributions). In other words, the design of an RL algorithm involves the following.

1) *Parameterization:* This decides how a parametric function $\hat{J}(p|\boldsymbol{\theta})$ is determined from a given parameter vector $\boldsymbol{\theta}$.
2) *Parameter Learning:* Parameter vector $\boldsymbol{\theta}^*$ is learned from a batch of data samples, and $\hat{J}(p|\boldsymbol{\theta}^*)$ is used to approximate J^*.

Learned $\boldsymbol{\theta}^*$ enables the construction of the transmission policy as

$$\hat{\pi}\left(s|\boldsymbol{\theta}^*\right) = \arg\max_a \left\{ r(s,a) + \hat{J}\left(\varrho(s,a)|\boldsymbol{\theta}^*\right)\right\}. \tag{16}$$

Comparing (16) with (11), we see that, if $\hat{J}(p|\boldsymbol{\theta}^*)$ approximates $J^*(p)$ well, the performance of $\hat{\pi}(s|\boldsymbol{\theta}^*)$ is close to that of $\pi^*(s)$ ([51, Ch. 6] provides rigorous statements).

In this section, we propose an RL algorithm, which exploits MNN [42] for parameterization (Section V-A) and learns the associated parameter vector via iteratively executing least square regression (Section V-B). The learned parameter vector is applied for transmission control in Section V-C. The time complexity of the proposed algorithm is analyzed in Section V-D.

A. Monotone Neural Network Approximation

The function parameterization should provide sufficient representation ability [parameter vector $\boldsymbol{\theta}$ is found such that $\hat{J}(p|\boldsymbol{\theta})$ is close to $J^*(p)$]. Artificial neural network (ANN) [52, Ch. 4] seems to be a good option, as the universal approximation theorem [53] states that a three-layer ANN is able to approximate a continuous function to arbitrary accuracy.

However, we know that J^* is nondecreasing from Theorem 1, whereas (classical) ANNs include all types of continuous functions (not necessarily nondecreasing). This would make the learning of parameters inefficient, as a learning algorithm needs to search over a not-necessarily large function space.

With this motivation, we propose the use of an MNN [42] for parameterization. Mathematically, the parameterized function $\hat{J}(p|\boldsymbol{\theta})$ with the MNN is expressed as

$$\hat{J}(p|\boldsymbol{\theta}) = \left(\sum_{i=1}^{N} u_i^2 \sigma_H\left(w_i^2 p + \alpha_i\right)\right) + \beta \tag{17}$$

with parameter vector

$$\boldsymbol{\theta} = [w_1, \ldots, w_N, \alpha_1, \ldots, \alpha_N, u_1, \ldots, u_N, \beta]$$

and function $\sigma_H(x) = 1/(1+e^{-x})$.

Function $\hat{J}(p|\boldsymbol{\theta})$ (17) is depicted in Fig. 3. It is actually a three-layered single-input–single-output ANN (with small modifications). Specifically, there is an input-layer with one single node, whose output represents the value of after-state p. In addition, there is a hidden layer with N nodes. The input of the ith node is the sum of weighted after-state value $w_i^2 \cdot p$ and hidden-layer bias α_i. And the input–output relationship of each hidden-layer node is defined by $\sigma_H(\cdot)$ (known as the sigmoid activation function [52, Ch. 4]). Finally, there is an output layer with one node, whose output represents the ultimate approximated function value $\hat{J}(p|\boldsymbol{\theta})$. Its input is the summation of weighted outputs from the hidden-layer and the output-layer bias β. And the output of the output-layer node is equal to its input.

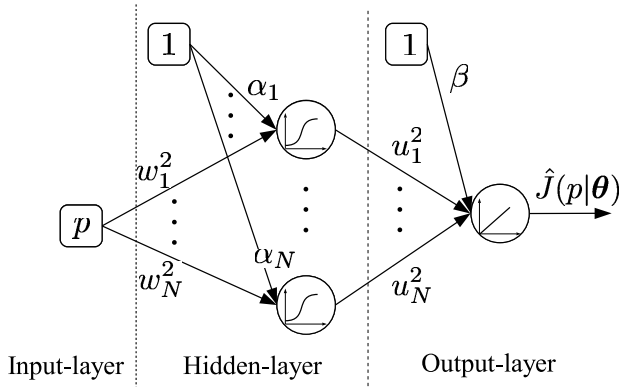


Fig. 3. MNN.

Note that the key difference between MNN and classical ANN is the sign of weights. The former has non-negative weights, which is not necessary for classical ANN. But MNN guarantees a nondecreasing function, which is what $\hat{J}(p|\boldsymbol{\theta})$ is as shown in Theorem 3.

Theorem 3: For any parameter θ, $\hat{J}(p|\boldsymbol{\theta})$ is a differentiable nondecreasing function.

Proof: We have $(d/dp)\hat{J}(p|\boldsymbol{\theta}) = \sum_{i=1}^{N} u_i^2 \cdot w_i^2 \times \sigma_H(w_i^2 p + \alpha_i) \times (1 - \sigma_H(w_i^2 p + \alpha_i))$. It is easily verified that $(d/dp)\hat{J}(p|\boldsymbol{\theta}) \geq 0$, which completes the proof. ∎

Moreover, Theorem 4 states that the proposed MNN has sufficient ability to represent any continuous and nondecreasing function.

Theorem 4: For any continuous nondecreasing function $J : [0,1] \mapsto \mathcal{R}$, there exists an MNN with N hidden nodes and parameter vector $\boldsymbol{\theta}$, such that $J(p) - \hat{J}(p|\boldsymbol{\theta}) \leq \epsilon$, for any $p \in [0,1]$ and $\epsilon > 0$.

Proof: It can be proven by [42, Th. 3.1], which considers a general case of representing a multiple-input–single-output nondecreasing function. ∎

Remark: Theorem 4 states that $\hat{J}(p|\boldsymbol{\theta})$ is able to approximate $J^*(p)$ to arbitrary accuracy via optimizing the parameter vector $\boldsymbol{\theta}$.

B. Fitted Value Iteration to Train MNN

Fitted value iteration [54] is a state-of-the-art learning methodology that is especially useful for training neural networks for optimal control. When working with complex neural networks (with multiple hidden layers), it is entitled with the name of deep RL [55], [56]. Here, we develop an algorithm, called fitted value iteration with MNN (FMNN), via tailoring the fitted value iteration method into our problem. FMNN trains an MNN to approximate $J^*(p)$ via exploiting a

batch of data samples. In the following, we first specify the required data. Then, the training process is presented.

1) Collecting Training Data: The required training data for FMNN is a batch of samples $\mathcal{F} = \{(p_m, s_m = [b_m, h_m, d_m])\}_{m=0}^{M-1}$, where $b_m \sim f_B(\cdot|p_m)$, $h_m \sim f_H(\cdot)$ and $d_m \sim f_D(\cdot)$. In the following, we provide two possible methods for collecting $\mathcal{F}$.

When PDFs $f_C(\cdot)$, $f_E(\cdot)$, $f_H(\cdot)$, and $f_D(\cdot)$ are known in advance, then we can use a simulator that is able to generate data samples of r.v.s c_t, e_t, h_t, and d_t following PDFs $f_C(\cdot)$, $f_E(\cdot)$, $f_H(\cdot)$, and $f_D(\cdot)$, respectively. In this case, we can first obtain $\{p_m\}_m$ by uniformly sampling over [0, 1]. Then, for a given p_m, we sample those r.v.s and get realizations (c, e, h, d). With these realizations, a valid data sample (p_m, s_m) can be obtained by setting $s_m = [((p_m + e)^- - c)^+, h, d]$. Finally, $\mathcal{F}$ is constructed by repeating the procedure for all p_m.

When PDFs $f_C(\cdot)$, $f_E(\cdot)$, $f_H(\cdot)$, and $f_D(\cdot)$ are unknown in advance, we can construct $\mathcal{F}$ via physically interacting with environments. We can run a certain sampling policy $\pi_S(s)$ (such as the greedy policy, i.e., always choose to send if energy is sufficient). And during the execution of $\pi_S(s)$, we can observe a sample path of r.v.s $(\ldots, s_{t-1} = [b_{t-1}, h_{t-1}, d_{t-1}], p_{t-1} = \varrho(s_{t-1}, \pi_S(s_{t-1})), s_t = [b_t, h_t, d_t], p_t = \varrho(s_t, \pi_S(s_t)), \ldots)$. Then, by setting $p_m = p_{t-1}$ and $s_m = [b_t, h_t, d_t]$, we are able to collect a valid sample (p_m, s_m). Finally, $\mathcal{F}$ is constructed by sweeping from $t = 0$ to M.[2]

2) Fitting MNN Iteratively: Here, we present FMNN, which trains an MNN to approximate $J^*(p)$ with $\mathcal{F}$ via imitating the iterative computing scheme of (13).

Specifically, similar to (13), FMNN works iteratively. At the kth iteration, suppose that the current MNN parameter vector is $\boldsymbol{\theta}_k$, which defines a function $\hat{J}(p|\boldsymbol{\theta}_k)$ with (17). As suggested by (13), given current value function $J_k(p) = \hat{J}(p|\boldsymbol{\theta}_k)$, the updated value function should be

$$J_{k+1}(p) = \gamma \cdot \mathbb{E}\left[\max_a \left\{r\big(s', a\big) + \hat{J}\big(\varrho\big(s', a\big)|\boldsymbol{\theta}_k\big)\right\}\middle| p\right]. \quad (18)$$

Therefore, we wish to update the MNN's parameters to obtain a new function $J(p|\boldsymbol{\theta}_{k+1})$ that is close to $J_{k+1}(p)$.

To do so, from $\mathcal{F}$ and $\boldsymbol{\theta}_k$, we construct a batch of data

$$\mathcal{T}_k = \{(p_m, o_m)\}_{m=0}^{M-1} \quad (19)$$

where

$$o_m = \gamma \cdot \max_a \left\{r(s_m, a) + \hat{J}(\varrho(s_m, a)|\boldsymbol{\theta}_k)\right\}. \quad (20)$$

Note that by comparing (20) with (18), o_m can be seen as a noisy realization of $J_{k+1}(p_m)$. In other words, given p_m as an input value, o_m defines the corresponding output of J_{k+1} (plus certain noise). Therefore, we may get a function that is close to $J_{k+1}(p)$ by training the MNN to fit the (noisy) input–output patterns contained in $\mathcal{T}_k$.

[2]After the initial data samples are collected, the node can apply the learned policy for transmission control, and during the transmission control, the node can collect new fresh data samples. The fresh data samples can be used to update the learned policy [57].

Algorithm 1 FMNN: Approximate $J^*(p)$

Input: Data samples $\mathcal{F} = \{(p_m, s_m)\}_{m=0}^{M-1}$
Output: Learned MNN $\hat{J}(p|\boldsymbol{\theta}_K)$
procedure
  Randomly initialize parameter $\boldsymbol{\theta}_0$
  for k from 0 to $K-1$ **do**
    for m from 0 to $M-1$ **do**
      Get (p_m, s_m) as the mth element of $\mathcal{F}$
      Compute o_m from $\boldsymbol{\theta}_k$ and s_m via (20)
      Collect $\mathcal{T}_k(m) = (p_m, o_m)$
    end for
    Regression: $\boldsymbol{\theta}_{k+1} = \text{Fit}(\mathcal{T}_k, \boldsymbol{\theta}_k)$ (see Algorithm 2)
  end for
  Learned MNN is determined from (17) with $\boldsymbol{\theta} = \boldsymbol{\theta}_K$
end procedure

Specifically, given $\mathcal{T}_k$, the MNN parameter is updated as

$$\boldsymbol{\theta}_{k+1} = \arg\min_{\boldsymbol{\theta}} \{\mathcal{L}(\boldsymbol{\theta}|\mathcal{T}_k)\} \quad (21)$$

where

$$\mathcal{L}(\boldsymbol{\theta}|\mathcal{T}_k) = \frac{1}{2M}\sum_{m=0}^{M-1}\Big(\hat{J}(p_m|\boldsymbol{\theta}) - o_m\Big)^2 \quad (22)$$

[the solving of (21) is discussed in Section V-B3]. That is, the output of updated function $\hat{J}(p|\boldsymbol{\theta}_{k+1})$ minimizes the square error with respect to data set $\mathcal{T}_k$, i.e., least square regression. Given sufficiently large M, this regression process can efficiently average out data's randomness. Hence, the approximation error between $J_{k+1}(p)$ and $\hat{J}(p|\boldsymbol{\theta}_{k+1})$ should be small.

With $\boldsymbol{\theta}_{k+1}$, we can generate $\mathcal{T}_{k+1}$ from $\boldsymbol{\theta}_{k+1}$ and $\mathcal{F}$ [similar to (19)], and then solve $\boldsymbol{\theta}_{k+2}$ by fitting the MNN to $\mathcal{T}_{k+1}$ (similar to (21)). Iterations continue by repeating the procedure. The ultimate learned function $\hat{J}(p|\boldsymbol{\theta}_K)$ after K iterations should be close to $J^*(p)$ with sufficiently large K. For illustration, Fig. 4 shows the 1st, 2nd, and 20th iterations during FMNN's execution. Summarizing above concepts, FMNN algorithm is presented in Algorithm 1.

3) Train MNN With Gradient Descent: Here, we apply gradient descent to address (21) for solving MNN parameter $\boldsymbol{\theta}_{k+1}$ such that the represented function $\hat{J}(p|\boldsymbol{\theta}_{k+1})$ fits an input–output pattern $\mathcal{T}_k$ in least square error sense.

Gradient descent works by iteratively searching over the parameter space. Denote $\boldsymbol{\theta}^{(0)}$ as the initial search point, which can be intuitively set as current MNN parameter $\boldsymbol{\theta}_k$. By gradient descent, the parameter searching is conducted as follows:

$$\boldsymbol{\theta}^{(l+1)} = \boldsymbol{\theta}^{(l)} - \xi^{(l)} \cdot \nabla\mathcal{L}\Big(\boldsymbol{\theta}^{(l)}\Big) \quad (23)$$

where $\xi^{(l)}$ is the updating step size and $\nabla\mathcal{L}(\boldsymbol{\theta}^{(l)})$ is the gradient of $\mathcal{L}$ (22) at $\boldsymbol{\theta}^{(l)}$. Given properly decreasing $\xi^{(l)}$ and sufficient number of iterations L, we set $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}^{(L)}$, which is considered as an approximated solution of (21).

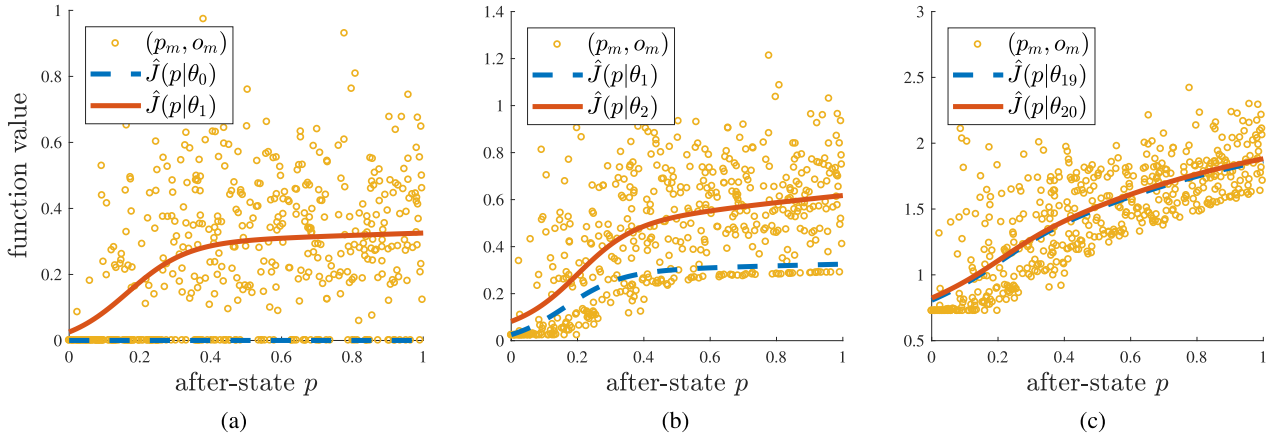


Fig. 4. Illustration of FMNN with $|\mathcal{F}| = 500$. (a) 1st iteration. (b) 2nd iteration. (c) 20th iteration.

Lastly, we derive $\nabla\mathcal{L}(\boldsymbol{\theta})$. With $\mathcal{T}_k = \{(p_m, o_m)\}_{m=0}^{M-1}$, the partial derivatives of $\mathcal{L}$ can be obtained as follows:

$$\frac{\partial \mathcal{L}}{\partial \beta} = \frac{1}{M}\sum_{m=0}^{M-1} \epsilon_m \tag{24}$$

$$\frac{\partial \mathcal{L}}{\partial u_i} = \frac{1}{M}\sum_{m=0}^{M-1} \left(\epsilon_m \cdot 2 \cdot u_i \cdot \sigma_H\left(w_i^2 \cdot p_m + \alpha_i\right)\right) \tag{25}$$

$$\frac{\partial \mathcal{L}}{\partial \alpha_i} = \frac{1}{M}\sum_{m=0}^{M-1} \left(\epsilon_m \cdot u_i^2 \cdot \sigma_H\left(w_i^2 \cdot p_m + \alpha_i\right) \times \left(1 - \sigma_H\left(w_i^2 \cdot p_m + \alpha_i\right)\right)\right) \tag{26}$$

$$\frac{\partial \mathcal{L}}{\partial w_i} = \frac{1}{M}\sum_{m=0}^{M-1} \left(\epsilon_m \cdot u_i^2 \cdot \sigma_H\left(w_i^2 \cdot p_m + \alpha_i\right) \times \left(1 - \sigma_H\left(w_i^2 \cdot p_m + \alpha_i\right)\right) \cdot 2 \cdot w_i \cdot p_m\right) \tag{27}$$

where

$$\epsilon_m = \hat{J}(p_m|\boldsymbol{\theta}) - o_m. \tag{28}$$

Therefore, the gradient of $\mathcal{L}$ is

$$\nabla\mathcal{L}(\boldsymbol{\theta}) = \left[\frac{\partial \mathcal{L}}{\partial w_1}, \ldots, \frac{\partial \mathcal{L}}{\partial w_N}, \frac{\partial \mathcal{L}}{\partial \alpha_1}, \ldots, \frac{\partial \mathcal{L}}{\partial \alpha_N}, \frac{\partial \mathcal{L}}{\partial u_1}, \ldots, \frac{\partial \mathcal{L}}{\partial u_N}, \frac{\partial \mathcal{L}}{\partial \beta}\right]. \tag{29}$$

Summarizing above results, we provide Algorithm 2, which works as an inner loop of FMNN for training an MNN to fit data set $\mathcal{T}_k$.

Algorithm 2 Inner Loop of FMNN: Fit Input–Output Pattern

Input: Input-output pattern $\mathcal{T}_k$, initial search point $\boldsymbol{\theta}_k$
Output: Trained parameter $\boldsymbol{\theta}_{k+1}$

1: **procedure**
2: $\quad\boldsymbol{\theta}^{(0)} = \boldsymbol{\theta}_k$
3: $\quad$**for** l from 0 to $L-1$ **do**
4: $\quad\quad$Compute $\nabla\mathcal{L}(\boldsymbol{\theta}^{(l)})$ with $\mathcal{T}_k$ via (24)-(27)
5: $\quad\quad$Obtain $\boldsymbol{\theta}^{(l+1)}$ with $\boldsymbol{\theta}^{(l)}$ and $\nabla\mathcal{L}(\boldsymbol{\theta}^{(l)})$ via (23)
6: $\quad$**end for**
7: $\quad\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}^{(L)}$
8: **end procedure**

C. Apply Learned MNN for Transmission Control

After collecting training data $\mathcal{F}$ and executing Algorithm 1, MNN parameter $\boldsymbol{\theta}_K$ is learned, and from it, a policy $\hat{\pi}(\cdot|\boldsymbol{\theta}_K)$ can be further constructed via (16) by setting $\boldsymbol{\theta}^* = \boldsymbol{\theta}_K$. As shown in [58] that, for large N, M and K, $\hat{\pi}(s|\boldsymbol{\theta}_K)$ (16) should be close to $\pi^*(s)$. Given the learned MNN/policy, we can then apply it for online selective transmission control, which is presented in Algorithm 3.

D. Computational Complexity

As we have shown, applying the proposed FMNN for transmission control involves a training phase (Algorithm 1) and a control phase (Algorithm 3). In the following, we analyze the time complexity of executing FMNN for training and control, respectively.

The main computational burden of the training phase is to execute the inner loop (Algorithm 2) K times. Furthermore, Algorithm 2 requires computing the partial derivative $\nabla\mathcal{L}(\boldsymbol{\theta}^{(l)})$ for L times. Moreover, from (29) and (24)–(27), it can be see that the complexity of computing the partial derivative is of $\mathcal{O}(M \times N)$. In summary, the time complexity of FMNN in training phase is of $\mathcal{O}(K \times L \times M \times N)$, i.e., linearly proportional to the number of value iterations (K), gradient descent steps (L), the size of MNN's hidden-layer (N), and the size of training data (M).

During the control phase, the major computational burden for determining whether or not to transmit a packet comes from evaluating the values $\hat{J}(p_0|\boldsymbol{\theta}_K)$ and $\hat{J}(p_1|\boldsymbol{\theta}_K)$ (see line 9 of Algorithm 3) at after-states p_0 and p_1. Therefore, from (17), it can be seen that the time complexity of each transmission decision making is of $\mathcal{O}(N)$, which is only relevant to the size of MNN's hidden layer. It can be seen that, once MNN is trained, the complexity for applying it for transmission control is negligible.

Algorithm 3 Online Transmission Control With Learned MNN

Input: Learned MNN $\hat{J}(\cdot|\boldsymbol{\theta}_K)$

procedure
  for t from 0 to ∞ **do**
    A packet arrives, and the node ends the silent period of cycle t
    Decode the packet and evaluate its priority d_t
    Probe CSI and estimate required transmit energy h_t
    Determine current remaining energy in battery b_t
    Construct state $s_t = [b_t, h_t, d_t]$
    Compute after-states $p_0 = \varrho(s_t, 0)$ and $p_1 = \varrho(s_t, 1)$
    Evaluate after-state values with trained MNN as $J_0 = \hat{J}(p_0|\boldsymbol{\theta}_K)$ and $J_1 = \hat{J}(p_1|\boldsymbol{\theta}_K)$
    if $J_0 > J_1 + \mathbb{1}(b_t \geq h_t) \cdot d_t$ **then** ▷ see (16)
      Discard the packet
    else
      Send the packet with energy h_t
    end if
    Battery is replenished with harvested energy e_t
    Enter silent period of cycle $t+1$
  end for
end procedure

VI. Numerical Simulation

We will next numerically study the learning characteristics of the proposed FMNN and the performance of the learned policy. In Section VI-B, we investigate the learning efficiency of FMNN. Section VI-C demonstrates the structure and performance of the learned policy. In Section VI-D, we further compare the performance of the proposed algorithm with existing algorithms under real measured data.

A. Simulation Setup

We model the wireless channels as Rayleigh fading, the most common model in wireless research. It is especially accurate for signal propagation in heavily built-up urban environments. The PDF of channel power gain z_t is then $f(x) = (1/\mu_Z)e^{-x/\mu_Z},\ x \geq 0$, where $\mu_Z = 1$ is the mean of z_t (other values of μ_Z are investigated in Section VI-C).

We assume that energy is harvested from wind power, which is well characterized by the Weibull distribution [59]. Hence, we model e_t with Weibull PDF $f_E(x) = (k_E/\lambda_E)(x/\lambda_E)^{k_E-1}e^{-(x/\lambda_E)^{k_E-1}},\ x \geq 0$, with shape parameter $k_E = 1.2$ and scale parameter $\lambda_E = 0.15/\,\Gamma(1+1/k_E)$, where $\Gamma(\cdot)$ is the gamma function. The shape and scale parameters imply that the mean of e_t equals 0.15.

Moreover, the total energy consumption c_t during the silent period, data reception, and channel estimation is modeled as a Gamma PDF $f_C(x) = (\Gamma(k_C)\theta_C^{k_C})^{-1}x^{k_C-1}e^{-(x/\theta_C)},\ x > 0$, with shape parameter $k_C = 10$, and scale parameter $\theta_C = 0.02/k_C$. The shape and scale parameters imply that the mean of c_t equals 0.02.

Furthermore, the model of data priority d_t depends on the specific practical application. We assume that d_t is exponentially distributed, i.e., $f_D(d) = e^{-d}, d \geq 0$. This assumption is also used in [8] and [13]. In Section VI-D, we further study the priority model for data packets obtained from a real experiment.

Finally, the number of hidden nodes N of the MNN is set to 3. The FMNN is executed with the data sample size of $|\mathcal{F}| = M = 500$, and the number of iterations is $K = 20$.

B. Sample Efficiency for Learning π^*

To evaluate learning efficiency, we use the *sample efficiency*, which is the number of data samples needed to be processed before an algorithm can learn a (near) optimal policy. Sample efficiency is a good proxy of an algorithm's training and adaptive abilities. We next assess the sample efficiency of FMNN, FNN (fitted value iteration with a classical neural network), and Online-DIS (online learning with after-state space discretization), which are constructed as follows.

1) FNN and Online-DIS: FNN is the same as FMNN except replacing the MNN with a three-layer classical ANN (without non-negative weights constraint) and modifying the gradient descent method for ANN in Algorithm 2. Thus, FNN does not exploit the monotonicity of J^*, and the learning efficiency is expected to be inferior to that of FMNN.

Online-DIS algorithm is developed via applying the well-known Q-learning algorithm [45, Ch. 6.5] into our problem (Q-learning is also chosen for comparison in [13]). Online-DIS applies discretization for parameterization and an online learning scheme for learning associated parameters. Specifically, Online-DIS discretizes the after-state space into $\bar{N}$ bins, which are, respectively, associated with $\bar{N}$ parameters. The nth parameter presents the "aggregated" function values of $J^*(p)$ for all after-states p that fall into the nth bin. These parameters are learned via continuously updating parameters with each available sample. In order to properly average out data samples' randomness, the updating step size needs to be sufficiently small (see [45, p. 39]). Therefore, the learning generally progresses fairly slowly, and requires a large amount of data samples.

2) Sample Efficiency Comparison: The setting of FNN is the same as that of FMNN. For Online-DIS, the number of bins $\bar{N}$ is set as 20. To investigate the learning efficiency, we evaluate the performance of learned policies when each iteration of FMNN and FNN consumes 500 data samples. Fig. 5 shows the results on a logarithmic scale and the learning progress.

First, we observe that FMNN and FNN are about 100 times more efficient than Online-DIS. This significant disparity occurs because FMNN and FNN directly train an MNN/ANN to fit data samples with regression, whereas Online-DIS must gradually average out randomness with a small step size.

Second, FMNN learns considerably faster than FNN. This is because FMNN exploits the nondecreasing property of J^* to learn a reasonably good policy with fewer iterations.

Finally, after processing enough data samples, the three learning curves converge. Both FMNN and FNN converge to

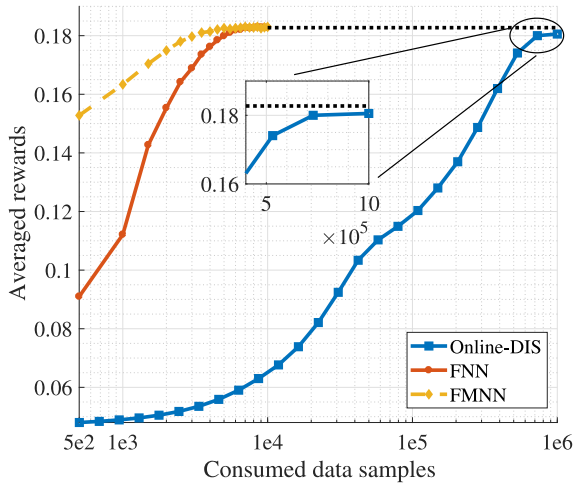


Fig. 5. Learning curve.

the same value, because both can represent a continuous nondecreasing function. Therefore, their learned policies achieve the same performance. In contrast, Online-DIS converges to a slightly inferior level.

Fig. 6 shows the learn value functions by the three algorithms. In addition, we also present the after-state value function $J^*(p)$, which is computed via (13) by exploiting the knowledge of PDFs $f_H(\cdot)$, $f_D(\cdot)$, and $f_B(\cdot|p)$ and taking sufficiently many p values. From Fig. 6(a), it can be seen that, after processing 1500 samples (3 iterations), FMNN learns a good value function, which is fairly close to J^*. The function learned by FNN after processing 1500 samples does not capture the nondecreasing structure of J^*. This fact suggests that FNN has inferior sample efficiency compared with FMNN. Nevertheless, both functions learned via FNN and FMNN eventually converge to J^*. Finally, as one can notice, even with 1.5×10^5 data samples, the function learned by Online-DIS fluctuates because the randomness of data samples has not been averaged out. Given 10^6 data samples and gradually decreasing step size, Online-DIS properly averages out noise and learns a nondecreasing function to fit J^*. However, due to after-state space discretization, the learned value function is piece-wise constant function, whose discontinuities cause a slight performance loss of the resulted policy (i.e., as shown in Fig. 5, Online-DIS converges to a slightly inferior level).

C. Structure and Performance of Learned Policy

Given FMNN's learned parameter $\boldsymbol{\theta}_K$, a policy $\hat{\pi}(s|\boldsymbol{\theta}_K)$ can be constructed via (16) with $\boldsymbol{\theta}^* = \boldsymbol{\theta}_K$. The constructed policy is shown in Fig. 7, in which decision "to send" is made if the current state is above the curved surface, and decision "to drop" is made otherwise. This is consistent with the optimal policy's structured results proved in Theorem 2 and Corollary 1.

With the constructed policy, Algorithm 3 can then be applied for selective transmission control.

The major difference between $\hat{\pi}(s|\boldsymbol{\theta}_K)$ and those of [8]–[13] is that: polices of [8]–[13] rely on available energy b_t and packet priority d_t only, whereas $\hat{\pi}(s|\boldsymbol{\theta}_K)$ further exploits CSI h_t. To investigate the performance gain of exploiting CSI, we compare $\hat{\pi}(s|\boldsymbol{\theta}_K)$, named as DecBDH (i.e., decision based on b_t, d_t, and h_t), with the policy of work [13], named as DecBD (i.e., decision based on b_t and d_t). Note that works [8]–[12] do not fit energy-harvesting and/or wireless fading settings. DecBD works as follows. It first uses the scheme in [13] to send or to discard a packet based on available energy b_t and packet priority d_t. If the decision is to send, the node transmits only if there is enough energy ($b_t \geq h_t$); otherwise the node does not transmit and there is no energy consumption for transmission.

TABLE I
SENT PACKETS' TOTAL PRIORITY VALUE

	Priority value	Relative improvement over AdaTx
AdaTx	850.8	N/A
DecBD	977.4	13%
DecBDH	1128.7	30%

We also compare with an adaptive transmission (AdaTx) scheme, which always tries to send if a successful transmission can be achieved, i.e., the node transmits with energy h_t if $b_t \geq h_t$, or drops the packet without energy consumption for transmission otherwise.

Fig. 8 shows the performance of DecBDH, DecBD, and AdaTx under different channel conditions. Note that DecBD outperforms AdaTx. The reason is that DecBD considers both data priority and energy status, and ensures the transmission of high priority packets and avoids transmissions when the available energy level is low. In contrast, DecBDH exploits instantaneous CSI, which translates more transmission opportunities at good channel conditions, and therefore, outperforms DecBD.

D. Compared Under Real Measured Data

Next, we further compare the performance of the algorithms with some real measured data. Specifically, the work in [60] reported indoor temperature data that were measured via a temperature sensor every 10 min for about 4.5 months, which yield 2×10^4 data packets. We define the priority of a data packet as the absolute difference between the temperature indicated by the current packet and that of the previous packet. Therefore, a high-priority packet indicates steep indoor temperature change, which may be critical for some applications, e.g., room occupancy detection. Fig. 9 shows the priority values of those measured packets in chronological order. In Table I, we present the total priority of packets sent by DecBDH, DecBD, and AdaTx. It can be seen that, DecBDH (learned via our proposed algorithm) performs the best; while DecBD follows. These two algorithm, respectively, achieve 30% and 13% performance gain compared to the least favorable policy AdaTx.

Lastly, we investigate packets' priority distribution before/after processing via a transmission policy. In Fig. 10, curve "Measured" represents the priority PDF of all arrived packets. The curves "AdaTx," "DecBD," and "DecBDH" are the priority PDF of packets transmitted by AdaTx, DecBD, and DecBDH, respectively. It can be seen that curve AdaTx

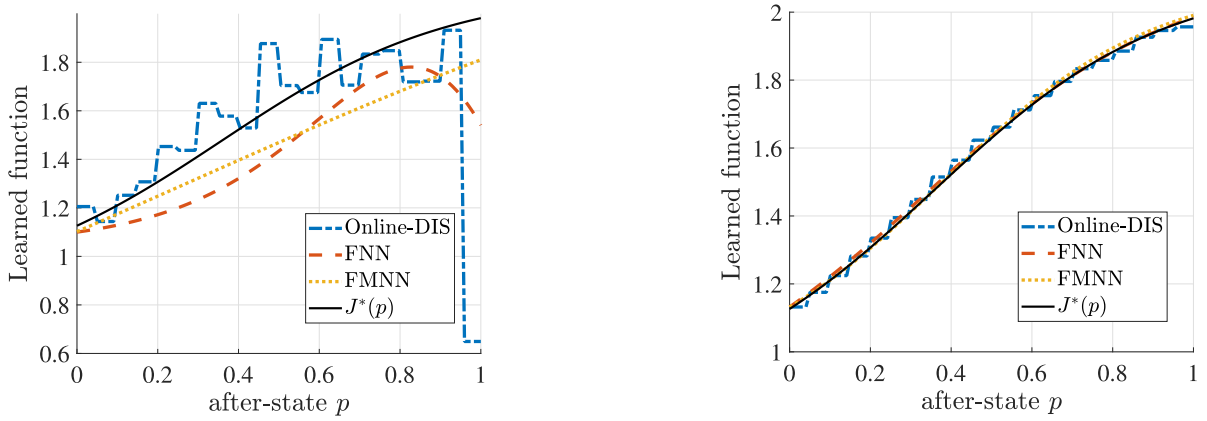


Fig. 6. Learned value functions. (a) FMNN and FNN after 1500 samples, and Online-DIS after 1.5×10^5 samples. (b) FMNN and FNN after 10^4 samples, and Online-DIS after 10^6 samples.

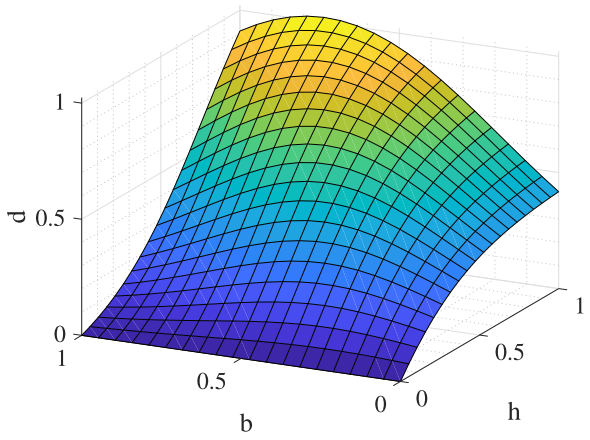


Fig. 7. Constructed policy $\hat{\pi}(s = [b, h, d]|\boldsymbol{\theta}_K)$ via learned MNN.

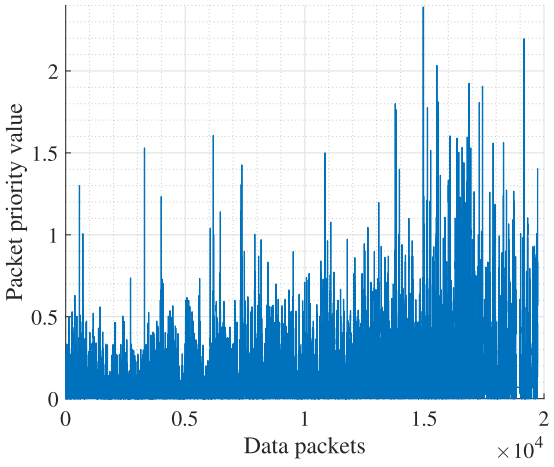


Fig. 9. Measured data from a temperature sensor.

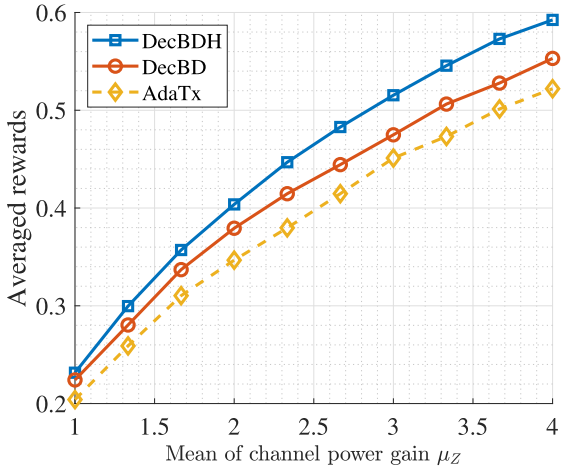


Fig. 8. Achieved performance under different channel conditions.

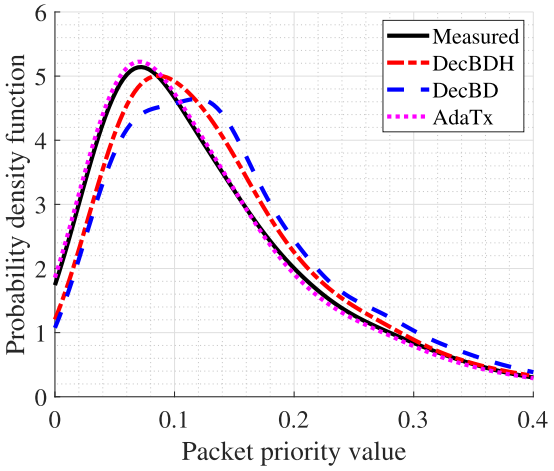


Fig. 10. PDF of packets' priority values.

is close to curve Measured, since AdaTx's transmission decision is not based on d_t. In contrast, policies DecBDH and DecBD selectively transmit high-priority packets, and therefore, curves DecBDH and DecBD shift toward high priority region (compared to curve AdaTx). Note that, compared to DecBDH, curve DecBD shifts more, due to the following reason. DecBD selects to transmit only when the arrived packet's priority is sufficiently high. On the other hand, when an arrived packet's priority is not sufficiently

high, DecBDH may still select to transmit if the channel gain is good. In other words, DecBDH transmits more packets, some of which are not with sufficiently high priority.

VII. Conclusion

We have considered the selective transmission problem for an energy-harvesting wireless sensor node, which is modeled under the framework of an MDP. The optimal policy is constructed by the after-state value function J^*, which greatly simplifies the problem. We showed that J^* is a differentiable and nondecreasing function. As an efficient solution of J^*, we proposed an FMNN learning algorithm, which approximates J^* with an MNN, and learns the associated parameters by iterative least-square regression. Our simulations demonstrated the learning efficiency of FMNN and also the performance gain achieved by the learned policy.

Appendix A
Proof of Lemma 1

Since $b_{t+1} = ((p_t + e_t)^- - c_{t+1})^+$, we have, for $x \in [0, 1]$

$$\begin{aligned}
&\text{Prob}(b_{t+1} > x | p_t = p) \\
&= \text{Prob}\Big(\big((p + e_t)^- - c_{t+1}\big)^+ > x\Big) \\
&= \text{Prob}\big((p + e_t)^- > x + c_{t+1}\big) \\
&= \text{Prob}(1 > x + c_{t+1} \text{ and } p + e_t > x + c_{t+1}) \\
&= \int_0^{1-x} \int_{x-p}^{1} f_E(e) \cdot f_C(c)\, de\, dc
\end{aligned} \tag{30}$$

where Prob$(\cdot)$ means probability, and the last equality holds as e_t has PDF f_E and c_{t+1} has PDF f_C.

It is easy to check that Prob$(b_{t+1} > x | p_t = p)$ does not depend on t, and

$$F_B(b|p) = 1 - \text{Prob}(b_{t+1} > b | p_t = p). \tag{31}$$

In addition, since f_E and f_C are both non-negative, Prob$(b_{t+1} > b|p_t = p)$ (30) is nondecreasing with respect to p, which implies $F_B(t|p)$ is nonincreasing with respect to p from (31). Finally, it is obvious that $(d/dp)F_B(t|p)$ and $(d/dt)F_B(t|p)$ exist (and can be computed via the Leibniz integral rule), which completes the proof. ∎

Appendix B
Proof of Theorem 1

With value iteration algorithm (13), we prove Theorem 1 by induction. Specifically, we set $J_0(p) = 0$ for all $p \in [0, 1]$, which is differential and nondecreasing. With this choice of J_0, we can prove Theorem 1 by showing that, if J_k is nondecreasing and differentiable, J_{k+1} is nondecreasing and differentiable.

First, from (13), we have

$$J_{k+1}(p) = \gamma \mathbb{E}\big[r\big(s', A'\big) + J_k\big(\varrho\big(s', A'\big)\big)\big|p\big] \tag{32}$$

where the expectation is taken over both s' and A', and A' is a function of r.v. s'

$$A' = \begin{cases} 1 & \text{if } r\big(s', 1\big) + J_k\big(\varrho\big(s', 1\big)\big) \geq J_k\big(\varrho\big(s', 0\big)\big) \\ 0 & \text{otherwise.} \end{cases}$$

In addition, due to the induction assumption and the fact $\varrho(s', 1) \leq \varrho(s', 0)$, we have $J_k(\varrho(s', 1)) \leq J_k(\varrho(s', 0))$. Therefore, if $A' = 1$, we must have $b' \geq h'$, since $b' < h'$ implies $r(s', 1) = 0$, and further implies $r(s', 1) + J_k(\varrho(s', 1)) < J_k(\varrho(s', 0))$, and therefore, $A' = 0$ (a contradiction of $A' = 1$).

Denoting $S_i(b)$ as the region of (h, d) with action $A' = i \in \{0, 1\}$ given b, we have $S_1(b) = \{(h, d)|b \geq h, d + J_k(b - h) > J_k(b)\}$, and $S_0(b) = \{(h, d)|b \geq h, d + J_k(b - h) \leq J_k(b)\} \cup \{(h, d)|b < h\}$. Given $S_0(b)$ and $S_1(b)$, $J_{k+1}(p)$ (32) can be rewritten as

$$J_{k+1}(p) = \gamma \int_0^1 f_B(\eta|p) \cdot M(\eta)\, d\eta \tag{33}$$

where

$$\begin{aligned}
M(\eta) \triangleq &\iint_{(x,y)\in S_1(\eta)} \big[y + J_k(\eta - x)\big] \cdot f_D(y) \cdot f_H(x)\, dy\, dx \\
&+ \iint_{(x,y)\in S_0(\eta)} J_k(\eta) \cdot f_D(y) \cdot f_H(x)\, dy\, dx \\
= &\int_0^{\eta} \int_{J_k(\eta-x)}^{+\infty} (y + J_k(\eta - x)) \cdot f_H(x) \cdot f_D(y)\, dy\, dx \\
&+ \int_0^{\eta} \int_0^{J_k(\eta-x)} J_k(\eta) \cdot f_H(x) \cdot f_D(y)\, dy\, dx \\
&+ \int_t^{+\infty} \int_0^{+\infty} J_k(\eta) \cdot f_H(x) \cdot f_D(y)\, dy\, dx.
\end{aligned} \tag{34}$$

In the following, we present Lemma 2, whose proof is given in Appendix C. With Lemma 2 and (33), we thereby can complete the proof by showing that $M(\eta)$ is differentiable and nondecreasing.

Lemma 2: Given that function $\Omega(\eta)$ is nondecreasing and differentiable for $\eta \in [0, 1]$, function $J(p)$ generated with $J(p) = \int_{\eta=0}^{1} f_B(\eta|p)\Omega(\eta)\, d\eta$ is nondecreasing and differentiable for $p \in [0, 1]$.

With the assumption that $J_k(p)$ is differentiable, the function $M(\eta)$ can be shown to be differentiable by repeatedly using the Leibniz's rule. Therefore, we are remaining to show that $M(\eta)$ is nondecreasing.

Given any $\eta_1, \eta_2 \in [0, 1]$ and $\eta_1 \geq \eta_2$, we define $S_{ij} \triangleq S_i(\eta_1) \cap S_j(\eta_2)$ with $i, j \in \{0, 1\}$. Then, from (34), it is easy to verify that

$$\begin{aligned}
&M(\eta_1) - M(\eta_2) \\
&= \underbrace{\mathbb{E}[\mathbb{1}((h, d) \in S_{11}) \cdot (J_k(\eta_1 - h) - J_k(\eta_2 - h))]}_{\triangleq \Delta_{11}} \\
&\quad + \underbrace{\mathbb{E}[\mathbb{1}((h, d) \in S_{10}) \cdot (d + J_k(\eta_1 - h) - J_k(\eta_2))]}_{\triangleq \Delta_{10}} \\
&\quad + \underbrace{\mathbb{E}[\mathbb{1}((h, d) \in S_{01}) \cdot (J_k(\eta_1) - d - J_k(\eta_2 - h))]}_{\triangleq \Delta_{01}} \\
&\quad + \underbrace{\mathbb{E}[\mathbb{1}((h, d) \in S_{00}) \cdot (J_k(\eta_1) - J_k(\eta_2))]}_{\triangleq \Delta_{00}}
\end{aligned}$$

where the expectations are taken over h and d. With the assumption that $J_k(p)$ is nondecreasing, the following results hold. Since $\eta_1 \geq \eta_2$, we have $J_k(\eta_1 - H) \geq J_k(\eta_2 - H)$ and

$J_k(\eta_1) \geq J_k(\eta_2)$, and therefore, $\Delta_{11} \geq 0$ and $\Delta_{00} \geq 0$. And according to the definition of S_{10}, we have $d_t + J_k(\eta_1 - H) \geq J_k(\eta_1) \geq J_k(\eta_2)$, which means $\Delta_{10} \geq 0$. Finally, for S_{01}, we have $J_k(\eta_1) \geq d_t + J_k(\eta_1 - H) \geq d_t + J_k(\eta_2 - H)$, which means $\Delta_{01} \geq 0$. In summary, we have $M(\eta_1) - M(\eta_2) \geq 0$ given $\eta_1 \geq \eta_2$. Therefore, $M(\eta)$ is nondecreasing. ■

APPENDIX C
PROOF OF LEMMA 2

Suppose the derivative of $\Omega(\eta)$ in $\eta \in [0, 1]$ is $\omega(\eta)$. Since $\Omega(\eta)$ is nondecreasing, we have $\omega(\eta) \geq 0$. Then, using integration by parts, we have

$$J(p) = \Omega(1) - \Omega(0)F_B(0|p) - \int_0^1 \omega(\eta) \cdot F_B(\eta|p)\, d\eta \quad (35)$$

where the fact $F_B(1|p) = 1$ is exploited. From Lemma 1, $F_B(\eta|p)$ is nonincreasing respected to p for any η. This means that $-\Omega(0)F_B(0|p)$ is nondecreasing. Furthermore, because $\omega(\eta) \geq 0$, we have $-\int_0^1 \omega(\eta) \cdot F_B(\eta|p)\, d\eta$ is nondecreasing (respected to p). In summary, $J(p)$ is nondecreasing.

In addition, since $F_B(\eta|p)$ is differentiable over p, it is easy to see from (35) that $J(p)$ is differentiable over p.

REFERENCES

[1] K. Wu, C. Tellambura, and H. Jiang, "Optimal transmission policy in energy harvesting wireless communications: A learning approach," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, May 2017, pp. 1–6.

[2] K. T. Phan, H. Jiang, C. Tellambura, S. A. Vorobyov, and R. Fan, "Joint medium access control, routing and energy distribution in multi-hop wireless networks," *IEEE Trans. Wireless Commun.*, vol. 7, no. 12, pp. 5244–5249, Dec. 2008.

[3] Z. Shen, H. Jiang, and Z. Yan, "Fast data collection in linear duty-cycled wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 63, no. 4, pp. 1951–1957, May 2014.

[4] F. K. Shaikh, S. Zeadally, and E. Exposito, "Enabling technologies for green Internet of Things," *IEEE Syst. J.*, vol. 11, no. 2, pp. 983–994, Jun. 2017.

[5] S. Sudevalayam and P. Kulkarni, "Energy harvesting sensor nodes: Survey and implications," *IEEE Commun. Surveys Tuts.*, vol. 13, no. 3, pp. 443–461, 3rd Quart., 2011.

[6] S. Ulukus *et al.*, "Energy harvesting wireless communications: A review of recent advances," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 3, pp. 360–381, Mar. 2015.

[7] K. W. Choi, P. A. Rosyady, L. Ginting, A. A. Aziz, D. Setiawan, and D. I. Kim, "Theory and experiment for wireless-powered sensor networks: How to keep sensors alive," *IEEE Trans. Wireless Commun.*, vol. 17, no. 1, pp. 430–444, Jan. 2018.

[8] R. Arroyo-Valles, A. G. Marques, and J. Cid-Sueiro, "Optimal selective transmission under energy constraints in sensor networks," *IEEE Trans. Mobile Comput.*, vol. 8, no. 11, pp. 1524–1538, Nov. 2009.

[9] R. Arroyo-Valles, A. G. Marques, and J. Cid-Sueiro, "Optimal selective forwarding for energy saving in wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 10, no. 1, pp. 164–175, Jan. 2011.

[10] J. Lei, R. Yates, and L. Greenstein, "A generic model for optimizing single-hop transmission policy of replenishable sensors," *IEEE Trans. Wireless Commun.*, vol. 8, no. 2, pp. 547–551, Feb. 2009.

[11] N. Michelusi, K. Stamatiou, and M. Zorzi, "On optimal transmission policies for energy harvesting devices," in *Proc. Inf. Theory Appl. (AIT) Workshop*, San Diego, CA, USA, Feb. 2012, pp. 249–254.

[12] N. Michelusi, K. Stamatiou, and M. Zorzi, "Transmission policies for energy harvesting sensors with time-correlated energy supply," *IEEE Trans. Commun.*, vol. 61, no. 7, pp. 2988–3001, Jul. 2013.

[13] J. Fernandez-Bes, J. Cid-Sueiro, and A. G. Marques, "An MDP model for censoring in harvesting sensors: Optimal and approximated solutions," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 8, pp. 1717–1729, Aug. 2015.

[14] Y. Li *et al.*, "Communication energy modeling and optimization through joint packet size analysis of BSN and WiFi networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 9, pp. 1741–1751, Sep. 2013.

[15] Y. Li, G. Zhou, and G. Peng, "Energy modeling and optimization for BSN and WiFi networks using joint data rate adaptation," *Ad Hoc Sensor Wireless Netw.*, vol. 32, nos. 1–2, pp. 149–173, 2016.

[16] M. A. Zafer and E. Modiano, "A calculus approach to energy-efficient data transmission with quality-of-service constraints," *IEEE/ACM Trans. Netw.*, vol. 17, no. 3, pp. 898–911, Jun. 2009.

[17] I. Ahmed, K. T. Phan, and T. Le-Ngoc, "Optimal stochastic power control for energy harvesting systems with delay constraints," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3512–3527, Dec. 2016.

[18] M. K. Sharma, A. Zappone, M. Debbah, and M. Assaad, "Deep learning based online power control for large energy harvesting networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Brighton, U.K., May 2019, pp. 8429–8433.

[19] M. K. Sharma, A. Zappone, M. Debbah, and M. Assaad, "Multi-agent deep reinforcement learning based power control for large energy harvesting networks," in *Proc. 17th Int. Symp. Model. Optim. Mobile Ad Hoc Wireless Netw. (WiOpt)*, Avignon, France, Jun. 2019, pp. 1–7.

[20] M. K. Sharma, A. Zappone, M. Assaad, M. Debbah, and S. Vassilaras, "Distributed power control for large energy harvesting networks: A multi-agent deep reinforcement learning approach," *arXiv preprint*, Apr. 2019. [Online]. Available: https://arxiv.org/abs/1904.00601

[21] M. Chu, H. Li, X. Liao, and S. Cui, "Reinforcement learning-based multiaccess control and battery prediction With energy harvesting in IoT systems," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2009–2020, Apr. 2019.

[22] N. Ashraf, A. Hasan, H. K. Qureshi, and M. Lestas, "Combined data rate and energy management in harvesting enabled tactile IoT sensing devices," *IEEE Trans. Ind. Informat.*, vol. 15, no. 5, pp. 3006–3015, May 2019.

[23] N. Ashraf, M. Faizan, W. Asif, H. K. Qureshi, A. Iqbal, and M. Lestas, "Energy management in harvesting enabled sensing nodes: Prediction and control," *J. Netw. Comput. Appl.*, vol. 132, pp. 104–117, Apr. 2019.

[24] K. Suto, H. Nishiyama, N. Kato, and T. Kuri, "Model predictive joint transmit power control for improving system availability in energy-harvesting wireless mesh networks," *IEEE Commun. Lett.*, vol. 22, no. 10, pp. 2112–2115, Oct. 2018.

[25] A. Keshavarzian, H. Lee, and L. Venkatraman, "Wakeup scheduling in wireless sensor networks," in *Proc. ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, Florence, Italy, 2006, pp. 322–333.

[26] J. Ma, W. Lou, Y. Wu, X.-Y. Li, and G. Chen, "Energy efficient TDMA sleep scheduling in wireless sensor networks," in *Proc. IEEE INFOCOM*, Rio de Janeiro, Brazil, Apr. 2009, pp. 630–638.

[27] D. Ye and M. Zhang, "A self-adaptive sleep/wake-up scheduling approach for wireless sensor networks," *IEEE Trans. Cybern.*, vol. 48, no. 3, pp. 979–992, Mar. 2018.

[28] Z. Zhou, S. Zhou, J.-H. Cui, and S. Cui, "Energy-efficient cooperative communication based on power control and selective single-relay in wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 7, no. 8, pp. 3066–3078, Aug. 2008.

[29] M. Dong, W. Li, and F. Amirnavaei, "Online joint power control for two-hop wireless relay networks with energy harvesting," *IEEE Trans. Signal Process.*, vol. 66, no. 2, pp. 463–478, Jan. 2018.

[30] S. K. Pallapothu and N. B. Mehta, "Energy-efficient detection using ordered transmissions in energy harvesting WSNs," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, May 2018, pp. 1–6.

[31] L. Lin, N. B. Shroff, and R. Srikant, "Asymptotically optimal energy-aware routing for multihop wireless networks with renewable energy sources," *IEEE/ACM Trans. Netw.*, vol. 15, no. 5, pp. 1021–1034, Oct. 2007.

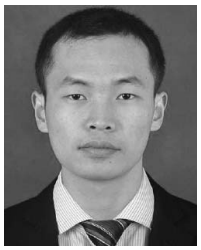
[32] G. Martinez, S. Li, and C. Zhou, "Wastage-aware routing in energy-harvesting wireless sensor networks," *IEEE Sensors J.*, vol. 14, no. 9, pp. 2967–2974, Sep. 2014.

[33] T. D. Nguyen, J. Y. Khan, and D. T. Ngo, "A distributed energy-harvesting-aware routing algorithm for heterogeneous IoT networks," *IEEE Trans. Green Commun. Netw.*, vol. 2, no. 4, pp. 1115–1127, Dec. 2018.

[34] L. Huang, "Fast-convergent learning-aided control in energy harvesting networks," in *Proc. 54th IEEE Conf. Decis. Control (CDC)*, Osaka, Japan, 2015, pp. 5518–5525.

[35] J. Carle and D. Simplot-Ryl, "Energy-efficient area monitoring for sensor networks," *IEEE Comput.*, vol. 37, no. 2, pp. 40–46, Feb. 2004.

[36] J. Zhang, W. Li, Z. Yin, S. Liu, and X. Guo, "Forest fire detection system based on wireless sensor network," in *Proc. IEEE Conf. Ind. Electron. Appl. (ICIEA)*, Xi'an, China, May 2009, pp. 520–523.
[37] T. Cui and C. Tellambura, "Joint data detection and channel estimation for OFDM systems," *IEEE Trans. Commun.*, vol. 54, no. 4, pp. 670–679, Apr. 2006.
[38] G. Wang, F. Gao, Y.-C. Wu, and C. Tellambura, "Joint CFO and channel estimation for OFDM-based two-way relay networks," *IEEE Trans. Wireless Commun.*, vol. 10, no. 2, pp. 456–465, Feb. 2011.
[39] G. Wang, F. Gao, W. Chen, and C. Tellambura, "Channel estimation and training design for two-way relay networks in time-selective fading environments," *IEEE Trans. Wireless Commun.*, vol. 10, no. 8, pp. 2681–2691, Aug. 2011.
[40] H. Arslan and G. E. Bottomley, "Channel estimation in narrowband wireless communication systems," *Wireless Commun. Mobile Comput.*, vol. 1, no. 2, pp. 201–219, Mar. 2001.
[41] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York, NY, USA: Wiley, 1994.
[42] H. Daniels and M. Velikova, "Monotone and partially monotone neural networks," *IEEE Trans. Neural Netw.*, vol. 21, no. 6, pp. 906–917, Jun. 2010.
[43] H. Kushner and G. G. Yin, *Stochastic Approximation and Recursive Algorithms and Applications*. New York, NY, USA: Springer, 2003.
[44] S. Weber, J. G. Andrews, and N. Jindal, "The effect of fading, channel inversion, and threshold scheduling on ad hoc networks," *IEEE Trans. Inf. Theory*, vol. 53, no. 11, pp. 4127–4149, Nov. 2007.
[45] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.
[46] N. Salodkar, "Online algorithms for delay constrained scheduling over a fading channel," Ph.D. dissertation, Dept. Comput. Sci. Eng., Indian Inst. Technol. Bombay, Bombay, India, 2008.
[47] N. Mastronarde and M. van der Schaar, "Fast reinforcement learning for energy-efficient wireless communication," *IEEE Trans. Signal Process.*, vol. 59, no. 12, pp. 6262–6266, Dec. 2011.
[48] Y. Cui, V. K. N. Lau, and Y. Wu, "Delay-aware BS discontinuous transmission control and user scheduling for energy harvesting downlink coordinated MIMO systems," *IEEE Trans. Signal Process.*, vol. 60, no. 7, pp. 3786–3795, Apr. 2012.
[49] F. Fu and M. van der Schaar, "Structure-aware stochastic control for transmission scheduling," *IEEE Trans. Veh. Technol.*, vol. 61, no. 9, pp. 3931–3945, Aug. 2012.
[50] K. Wu, H. Jiang, and C. Tellambura, "Sensing, probing, and transmitting strategy for energy harvesting cognitive radio," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, May 2017, pp. 1–6.
[51] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*. Belmont, MA, USA: Athena Sci., 1996.
[52] T. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, 1997.
[53] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Control Signals Syst.*, vol. 2, no. 4, pp. 303–314, Feb. 1989.
[54] M. Riedmiller, "Neural fitted Q iteration—First experiences with a data efficient neural reinforcement learning method," in *Proc. Eur. Conf. Mach. Learn. (ECML)*, Porto, Portugal, Oct. 2005, pp. 317–328.
[55] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[56] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access in wireless networks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 2, pp. 257–265, Jun. 2018.
[57] S. Lange, T. Gabel, and M. Riedmiller, "Batch reinforcement learning," in *Reinforcement Learning: State-of-the-Art*, M. Wiering and M. van Otterlo, Eds. Heidelberg, Germany: Springer, 2012, pp. 45–73.
[58] R. Munos and C. Szepesvári, "Finite-time bounds for fitted value iteration," *J. Mach. Learn. Res.*, vol. 9, no. 5, pp. 815–857, May 2008.
[59] J. Seguro and T. Lambert, "Modern estimation of the parameters of the Weibull wind speed distribution for wind energy analysis," *J. Wind Eng. Ind. Aerodyn.*, vol. 85, no. 1, pp. 75–84, Mar. 2000.
[60] L. M. Candanedo, V. Feldheim, and D. Deramaix, "Data driven prediction models of energy use of appliances in a low-energy house," *Energy Build.*, vol. 140, no. 4, pp. 81–97, Apr. 2017.



Keyu Wu received the Ph.D. degree in electrical engineering from the University of Alberta, Edmonton, AB, Canada, in 2018.

His current research interests include detection and estimation, dynamic stochastic control, machine learning, and their applications in wireless communications.

Fudong Li is currently pursuing the Ph.D. degree with the University of Alberta, Edmonton, AB, Canada.

His current research interests include resource allocation in wireless systems, nonorthogonal multiple access, and energy harvesting.

Chintha Tellambura (F'11) received the Ph.D. degree in electrical engineering from the University of Victoria, Victoria, BC, Canada.

He was with Monash University, Clayton, VIC, Australia, from 1997 to 2002. He is currently a Professor with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada. His current research interests include design, modeling, and analysis of cognitive radio, heterogeneous cellular networks, 5G wireless networks, and machine learning algorithms.

Hai Jiang (SM'15) received the Ph.D. degree in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 2006.

Since July 2007, he has been a Faculty Member with the University of Alberta, Edmonton, AB, Canada, where he is currently a Professor with the Department of Electrical and Computer Engineering. His current research interests include radio resource management, cognitive radio networking, and cooperative communications.