Learning to Transmit Fresh Information in Energy Harvesting Networks

Shiyang Leng and Aylin Yener, *Fellow, IEEE,*

***Abstract*—We study age of information (AoI) minimization in an ad hoc network consisting of energy harvesting transmitters that are scheduled to send status updates to their intended receivers. The transmission scheduling with power allocation problem over a communication session is first studied assuming apriori knowledge of channel state information, harvested energy, and update packet arrivals, i.e., the offline setting. The global optimal scheduling policy in this case is the solution of a mixed integer linear program which is known to be computationally hard. We propose a supervised-learning-based algorithm to mitigate the high computational complexity. A bidirectional recurrent neural network that interprets user scheduling as a time-series classification problem is trained and tested to achieve near-optimal AoI. Next, we consider online scheduling and power allocation with causal knowledge of the system state, which is an infinite-state Markov decision problem. In this case, the related reinforcement learning problem is solved by a model-free on-policy deep reinforcement learning, where the actor-critic algorithm with deep neural network function approximation is implemented. Comparable AoI to the optimal is demonstrated and faster runtime of learning solvers is observed, verifying the efficacy of learning in terms of both optimality and computational energy efficiency for AoI-focused scheduling and resource allocation problems in wireless networks.**

***Index Terms*—Age of information, information freshness, energy harvesting, user scheduling, mixed integer programming, machine learning, BiLSTM, actor-critic deep reinforcement learning.**

I. Introduction

Timely information exchange is crucial for many of forthcoming wireless networking applications, including vehicular networks, unmanned aerial vehicle networks, and IoT networks [3]–[5]. Maintaining freshness of information in such networks brings about the need for a new network design metric, i.e., *age of information* (AoI) [6], [7]. AoI quantifies the time elapsed since the generation of the latest successfully received update. Distinct from metrics of delay or latency, AoI thus captures the timeliness of information from a receiver's perspective.

AoI has been studied widely in various system setups since its introduction, see for example [8]–[16]. Reference [7] has analyzed AoI from a queuing theoretic perspective, and characterized AoI for single-source M/M/1, M/D/1, and D/M/1 queues with first-come-first-served (FCFS) service, revealing that AoI minimization offers different insights than delay minimization. The authors have next considered last-come-first-served (LCFS) service for M/M/1 with and without preemption in [8], and concluded that AoI can be potentially reduced by LCFS and preemption. Extensive studies on the analysis of AoI has subsequently been carried out for multiple sources [9], multiple servers [10], with packet deadlines [11], for multi-hop networks [12]. General arrival and service time distributions have been considered in reference [13], where the stationary distributions of AoI for FCFS and LCFS service disciplines are characterized, considering M/G/1, G/M/1, and G/G/1 queuing models. The exact age for G/G/1/1 systems with and without service preemption are derived in [14]. Packet generation at-will has been considered in [15], and general nonnegative, nondecreasing age penalty functions are proposed. For arbitrary packet generation times and arrival times, the preemptive Last-Generated-First-Serve (LGFS) policy is shown to be age-optimal in [16] in a stochastic ordering sense for multi-server systems with exponential service times. This paper focuses on wireless ad hoc networks through which timely information needs to be sent.

User scheduling, i.e., scheduling transmissions in a (wireless) ad hoc network, is a classical resource allocation problem with a decades-long history, often with throughput as the metric, see for example [17], [18]. Recent references have considered user scheduling for minimum AoI. In [19], the multi-source scheduling problem is identified as NP-hard as an integer linear program, and a suboptimal algorithm is proposed to reduce complexity. User scheduling in broadcast wireless networks is considered in [20], where the base station transmits updates to users scheduling one user at a time. Interference constraints are considered in [21] when a subset of links can be active simultaneously. Extensions on time-varying channels with perfect CSI and CSI statistics are studied in [22] and [23], respectively. Decentralized scheduling in multi-access uplink channels is investigated in [24], where round-robin scheduling with newest-packet buffer is proved to be asymptotically optimal with a massive number of terminals.

In energy harvesting communication networks, where communication is powered by intermittently acquired energy, energy availability has to be explicitly taken into account to ensure the information freshness. Consequently, AoI-optimal update policies under energy harvesting constraints have generated significant recent interest. AoI minimization for an energy harvesting source has been considered in references [25], [26]. In [27], asymptotically optimal update policies based on system statistics are derived. In particular, for a unit battery,

Manuscript received February 15, 2022; revised May 29, 2022; accepted June 20, 2022. This paper was presented in part at IEEE ICASSP 2021 [1] and at IEEE Asilomar Conference on Signals, Systems, and Computers 2021 [2].

Shiyang Leng was with The Pennsylvania State University, University Park, PA 16801, USA (e-mail: sfl5154@psu.edu).

Aylin Yener is with the Department of Electrical and Computer Engineering, The Ohio State University, Columbus, OH 43210, USA (e-mail: yener@ece.osu.edu).

a threshold policy is shown to be optimal. Reference [28] has further considered the full battery recharge and incremental battery recharge models for a finite battery. In the class of renewal policies, the optimal policy is proved to have an energy-dependent multi-threshold structure. AoI minimization for energy harvesting nodes is further investigated in [29]–[31] considering energy-controlled delays, two-hop systems, and erasure channels, respectively. Update failure due to channel noise is also considered in [32]. In our earlier work [33], online AoI minimization in cognitive radio networks is studied by formulating Markov decision processes (MDPs) taking into account opportunistic spectrum access by an energy harvesting AoI-focused secondary user. In [34], a non-linear function of AoI is analyzed considering randomness in update generation, transmission, and energy arrival. AoI in wireless powered networks, where nodes obtain energy from dedicated wireless energy signals, has been studied in [35]–[38]. For a recent comprehensive survey in AoI, see [39].

Most works on AoI-focused energy harvesting communications study optimal update policies and their properties for simple network structures, e.g., point-to-point transmission, which are amenable to model-based analytic approaches. In this paper, we take a different view point and consider a network formation model that is unlikely to admit a simple solution, but could benefit from learning-based approaches. Specifically, we consider a wireless network consisting of nodes capable of energy harvesting to send status updates, each wanting to update their intended receivers. The goal is to enable *all users* to reliably update their status information in a timely and energy efficient manner over a common channel, by selecting their transmission times and their transmit powers. We consider time division as in [19], [20], while taking into account energy constraints, channels, as well as the packet arrival process. User scheduling in both offline and online settings are investigated with different levels of system state information. The problem is solved in a centralized manner by the proposed approaches, where the scheduling decision is signaled to the wireless nodes in control messages. The signaling overhead for control signaling and node status report and the corresponding energy consumption are considered as normal system operation cost.

In the offline setting with full apriori knowledge of the system state, user scheduling is formulated as a mixed integer program with the objective of AoI minimization, whose solution complexity is a critical issue due to the binary variables of transmitter activation. The typical approach to address this challenge has been to either solve a computationally easier optimization problem related to the original one, i.e., a relaxation, or identify a near-optimal solution by some other means. Recent advent of machine learning in wireless communications [40], [41] provides a new avenue. For the offline problem, we utilize machine learning with the express purpose of solving a computationally hard optimization problem, and assess the potential of utilizing such an approach for problem sizes where computation cost of the optimal solution maybe prohibitive. We propose a learning-based algorithm that solves the offline problem as a time series classification problem. Recurrent neural networks are particularly successful in processing time series data for sequence-to-sequence learning [42], and that is the approach we take. In particular, we implement a multi-layered neural network with bidirectional long short-term memory (BiLSTM) to map the system state input vector to the user scheduling output vector, which performs near-optimally. This is meant to be a benchmark for any system with causal knowledge and is typically a first step in scheduling problems.

We next turn our attention to the more realistic online setting where only causal information of the system state is available. We aim to develop an online scheduling policy based on the current and the past observations. Unlike classical model-based approaches, here, we aim to do so without the knowledge of *any statistics* of the system state on energy, channel, or packet arrivals. In this case, the associated MDP formulation lacks the transition model rendering a model-free reinforcement learning problem. Among diverse learning approaches, deep reinforcement learning (DRL) [43]–[45], is a powerful technique for dynamic control design and has been applied in solving challenging problems in wireless networks due to its capability of dealing with model-free and large dimensional systems. In [46], update policy for AoI minimization is learned via reinforcement learning algorithms when the successful update probability is unknown. In [47], a multi-agent deep reinforcement learning algorithm based on the deep recurrent Q-network is proposed to solve the AoI-oriented online user scheduling in wireless (non-energy harvesting) ad hoc networks. Recent reference [48] implements a deep Q-network (DQN) learning to jointly optimize wireless energy transfer and scheduling of update packets in an MDP formulation with a large dimensional state space. For our model-free system with continuous-valued states, we address the online user scheduling leveraging DRL. An actor-critic algorithm with neural network function approximations is utilized, which is an on-policy algorithm and does not require large memory for experience replay in contrast to DQN.

We highlight our main contributions in this paper as follows.

- We formulate the user scheduling and power allocation for AoI minimization in energy harvesting ad hoc networks as a mixed integer linear program.
- We propose a learning-based near-optimal algorithm that transforms the offline problem to a time series classification problem and design a recurrent neural network with BiLSTM layers to compute the user scheduling.
- We formulate the online user scheduling and power allocation as a reinforcement learning problem and solve it using the actor-critic DRL algorithm.
- The optimality and energy efficiency of the proposed learning solvers are verified by experimental results which demonstrates near-optimal AoI performance and significant reduction in runtime as compared to the optimization solver.

The remainder of the paper is organized as follows. Section II presents the system model. In Section III, we focus on offline AoI minimization. The online AoI minimization is studied in Section IV. In Section V, we show the experimental results. Section VI concludes the paper.

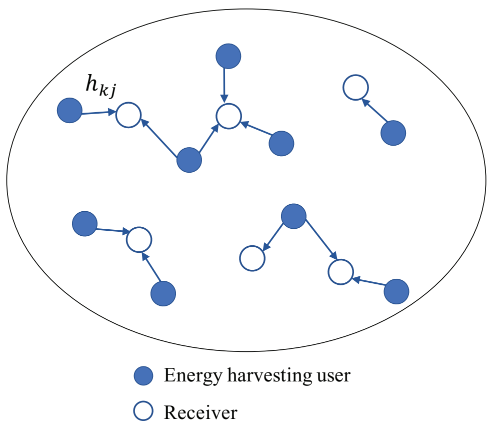

Fig. 1. System model. h_{kj} denotes the channel gain of user k in time slot j.

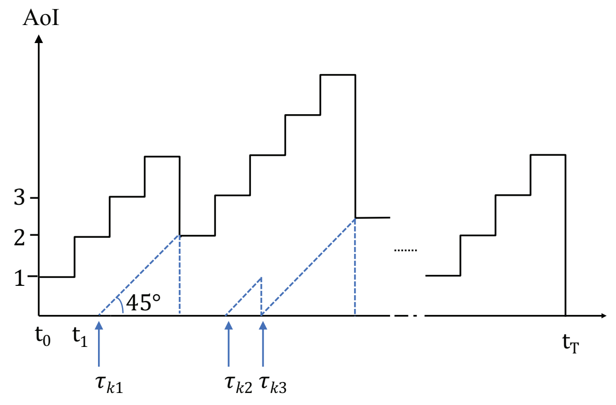

Fig. 2. A sample path of AoI. Dashed lines indicate the packet waiting time at the user and the solid stair-shaped lines indicate the AoI counted discretely at the end of each slot by the receiver. The first and the third packet are delivered. The second update is replaced by the third one.

II. System Model

We consider a system consisting of K users and their intended receivers. Each user wants to send status updates, e.g., of a physical process, and would like to keep the information fresh at its intended receiver, as shown in Fig. 1. The transmitter-receiver pairs are fixed throughout the session[1]. The user index is denoted by $k \in \{1, 2, \ldots, K\}$. The users harvest energy from ambient energy sources and transmit update packets consuming the harvested energy. The battery capacity at each user is assumed to be sufficiently large so that energy overflow need not be considered; this simplifies the energy state formulation. Each node is assumed to harvest energy independently, i.e., correlation of harvested energy at different nodes is not considered in our formulation. Time is slotted and the duration of each slot is normalized 1 second for simplicity. The jth slot indicates the time interval $[t_{j-1}, t_j)$, where $j = 1, 2, \ldots$ and $t_0 = 0$. The channel between each user and its receiver is assumed to be flat-fading. The path loss and Rayleigh multipath fading are taken into account for the channel gain, which is denoted by h_{kj} for user k in slot j. At most one user is scheduled to transmit in each slot. Each user either transmits to its receiver *or* harvests energy in each slot so that idle users harvest and accumulate energy. Let p_{kj} denote the transmission power and e_{kj} denote the energy that user k can harvest in slot j.

Each user sends update packets that are generated by itself or received from an external source. Either scenario is referred to as packet arrivals in the paper. The timestamp for the arrival of the uth update at user k is denoted by τ_{ku}, for $u = 1, 2, \ldots, U_k$, where U_k is the total number of updates in T slots. The new update packet replaces the old one that has not been sent out. Thus, only the newest packet is buffered at each user for the sake of information freshness. We assume that the size of the update packet is uniform and small, for which the transmission takes one slot. An update is delivered successfully by user k if the received signal-to-noise ratio (SNR) is larger than a target SNR γ_k^*, that is,

$$\frac{p_{kj} h_{kj}}{\sigma^2} \geq \gamma_k^*, \tag{1}$$

[1]Multiple transmitters may have the same intended receiver.

where σ^2 is the noise power.

We adopt a linear AoI model [6], [7]. AoI is defined as the time elapsed since the most recently received update is generated. Let a_{kj} denote the AoI for the user k at t_j, which indicates the age of the received packets at the end of slot j. At each slot, the scheduled user is enabled to transmit an update packet. If the delivery is successful, i.e., the received SNR is above the target, the age drops to $t_j - \tau_{ku_j}$ for the delivery of packet u_j, where u_j is the newest packet by t_{j-1}. Otherwise, the age grows by 1, as shown in Fig. 2. AoI evolves as follows for all k, j.

$$a_{kj} = \begin{cases} t_j - \tau_{ku_j}, & \text{if user } k \text{ delivers packet } u_j \text{ at } t_j \\ & \text{successfully,} \\ a_{k,j-1} + 1, & \text{otherwise,} \end{cases} \tag{2}$$

where a_{k0} is the AoI at t_0 for user k. the The objective is to minimize the average AoI of the system. Next, we consider the AoI minimization problem in offline and online settings.

III. Offline AoI Minimization

In the offline setting, we consider the AoI minimization problem with channel states, energy states, and packet arrival timestamps for T slots known apriori to the scheduler. Optimal transmission policy of user scheduling and power allocation in this case is the solution to a mixed integer linear program (MILP).

A. Mixed Integer Linear Program

We first define binary scheduling variables. Let $y_{kj} \in \{0, 1\}$ denote the update scheduling variable, where $y_{kj} = 1$ indicates user k is scheduled to send an update in slot j and $y_{kj} = 0$ indicates it is idle and harvesting energy. Let $x_{kuj} \in \{0, 1\}$ denote the packet scheduling variable, that $x_{kuj} = 1$ indicates the uth packet of user k is sent in slot j and successfully delivered at t_j, and $x_{kuj} = 0$ otherwise. That is,

$$y_{kj} = \begin{cases} 1, & \text{if user } k \text{ is scheduled at } j\text{th slot}, \\ 0, & \text{otherwise}. \end{cases} \tag{3}$$

$$x_{kuj} = \begin{cases} 1, & \text{if user } k \text{ delivers packet } u \text{ at } t_j, \\ 0, & \text{otherwise}. \end{cases} \tag{4}$$

The energy state at t_j is expressed in terms of the update scheduling variable y_{kj} and the transmission power p_{kj}, which is given by

$$E_{kj} = e_{k0} + \sum_{i=1}^{j} e_{ki}(1 - y_{ki}) - \sum_{i=1}^{j} p_{ki}, \tag{5}$$

$$= E_{k,j-1} + e_{kj}(1 - y_{kj}) - p_{kj}. \tag{6}$$

where e_{k0} is the initial energy of user k.

The average AoI over all users and T slots is minimized over the scheduling variables, as well as the transmission powers. The MILP formulation is given by

$$\min_{x_{kuj},\, y_{kj} \in \{0,1\},\, a_{kj},\, p_{kj}} \Delta = \frac{1}{TK} \sum_{j=1}^{T} \sum_{k=1}^{K} a_{kj} \tag{7a}$$

$$\text{s.t.} \sum_{k=1}^{K} y_{kj} \leq 1, \ \forall j, \tag{7b}$$

$$0 \leq p_{kj} \leq p_{\max} y_{kj}, \ \forall k, j, \tag{7c}$$

$$E_{kj} \geq 0, \ \forall k, j, \tag{7d}$$

$$\frac{p_{kj} h_{kj}}{\sigma^2} \geq \gamma_k^* y_{kj}, \ \forall k, j, \tag{7e}$$

$$a_{kj} \geq a_{k,j-1} + 1 - y_{kj}(a_{k0} + T), \ \forall k, j, \tag{7f}$$

$$a_{kj} \geq t_j - \sum_{u=1}^{U_k} \tau_{ku} x_{kuj} - (1 - y_{kj}) t_j, \ \forall k, j, \tag{7g}$$

$$x_{kuj}(t_{j-1} - \lceil \tau_{ku} \rceil)(\lfloor \tau_{k,u+1} \rfloor - t_{j-1}) \geq 0, \ \forall k, u, j, \tag{7h}$$

$$\sum_{j=1}^{T} x_{kuj} \leq 1, \ \forall k, u, \tag{7i}$$

$$y_{kj} = \sum_{u=1}^{U_k} x_{kuj}, \ \forall k, j, \tag{7j}$$

The optimization objective of the average AoI over time and users is given by (7a). Constraint (7b) indicates that at most one user is scheduled to update in each slot. Constraint (7c) imposes that the power is no larger than $p_{\max}$ if the user is scheduled, and $p_{kj} = 0$ if not. In (7d), we specify the energy causality constraint that the consumed energy by t_j is no larger than the harvested energy. The constraint in (7e) dictates that the update by the user k is successful if the received SNR is at least γ_k^* when the user is scheduled. The inequalities in (7f) and (7g) specify the AoI of user k, which are linear variations of (2) [19]. Specifically, if user k is idle, i.e., $y_{kj} = 0$, (7f) becomes $a_{kj} \geq a_{k,j-1} + 1$ and (7g) becomes $a_{kj} \geq -\sum_{u=1}^{U_k} \tau_{ku} x_{kuj}$. Thus, (7f) must be satisfied with equality due to the minimization of the objective and the nonnegative age. For $y_{kj} = 1$, the RHS of (7f) is negative since the age cannot be larger than $a_{k0} + T$, where $a_{k0} > 0$ is the initial AoI of user k. Then, only (7g) is active, which implies that $a_{kj} \geq t_j - \sum_{u=1}^{U_k} \tau_{ku} x_{kuj}$. Constraint (7h) ensures that the uth packet can only be scheduled after its arrival and before the replacement of the next packet. That is, t_{j-1}, the beginning of the scheduled slot for packet u has to be within the interval $[\lceil \tau_{ku} \rceil, \lfloor \tau_{k,u+1} \rfloor]$ to satisfy the nonnegativity expressed in (7h). By (7i), each packet is transmitted only once or dropped. Constraint (7j) implies the relationship between y_{kj} and x_{kuj}, that an user must be scheduled if one of its packet is to be sent.

The solution of the MILP in (7) can be found by using the off-the-shelf optimization solver CPLEX [49], which implements algorithms like branch-and-bound and cutting planes [50]. As MILP is known to be computationally hard to solve [50], we next utilize a learning-based algorithm to obtain scheduling policies.

B. Learning-based Algorithm

We propose a learning-based near-optimal algorithm in order to alleviate the computational burden of the MILP for large K. In particular, the original problem in (7) is decomposed into two steps: determining the update scheduling variables and calculating the transmission power. We utilize supervised learning and design a neural network (NN) to solve the scheduling subproblem as a time series classification problem, where each time step of the sequence input is classified into one of the output classes [42], [51]. The training input for the sequence to sequence learning is the system state information over T steps, and the training output is the sequence of the optimal update scheduling variables obtained by solving the MILP in (7). For our system with K users and only one update in each slot, the output of the NN at each time step is one of the $K+1$ classes, where class k indicates user k is scheduled to update, i.e., $y_{kj} = 1, y_{mj} = 0, \forall m \neq k$, and class $K+1$ denotes that no one is scheduled, i.e., $y_{kj} = 0, \forall k$.

We construct a multi-layered bidirectional recurrent neural network mapping the input system state sequence to a vector of update scheduling variables. Since user scheduling is determined by exploiting the dependencies of the system states over time slots, the bidirectional long short-term memory (BiLSTM) layer is deployed as the core of the NN to learn these dependencies in both forward and backward directions [52], [53]. An LSTM block consists of a cell, a forget gate, an input gate, and an output gate. The cell tracks the information of the time series input. At each block (time step), based on the output of the last block and the current time step of the input sequence, the three gates control the level of information to forget, update, and output for the cell state. The unidirectional LSTM structure helps maintain an internal state and aggregate the history of observations [54], [55], while BiLSTM preserves information from both the past and the future by letting cell states flow both forwards and backward, as shown in Fig. 3.

Based on the MILP in (7), we extract the state information of the system as input features, which are the energy harvesting state e_{kj}, the required energy $\frac{\sigma^2 \gamma_k^*}{h_{kj}}$, the maximum available energy $\min\{p_{\max}, \sum_{i=0}^{j} e_{ki}\}$, and the age at a successful

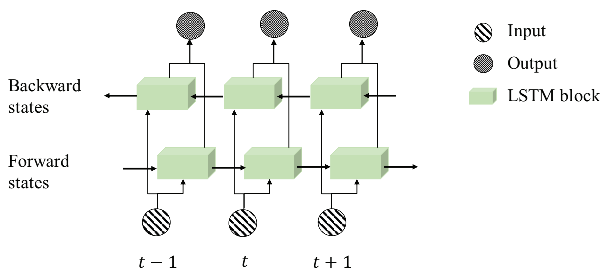

Fig. 3. A general structure of a BiLSTM layer shown unfolded in time for three time steps [52].

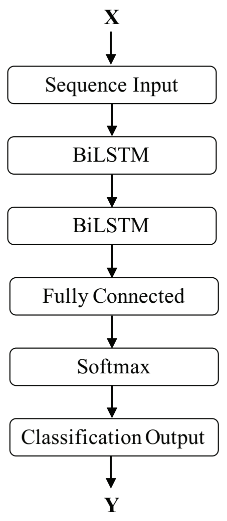

Fig. 4. The structure of the BiLSTM NN.

update instant, i.e., $t_j - \tau_{ku_j}$. Hence, one input sequence sample is represented by a K-by-T matrix given by

$$\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_K \end{bmatrix}, \tag{8}$$

where X_k is the 4-by-T block matrix for user k:

$$X_k = \begin{bmatrix} \text{——} & \log_{c_1}(e_{kj}) & \text{——} \\ \text{——} & \log_{c_2}\left(\frac{\sigma^2\gamma_k^*}{h_{kj}}\right) & \text{——} \\ \text{——} & \log_{c_3}\left(\min\left\{p_{\max}, \sum_{i=0}^{j} e_{ki}\right\}\right) & \text{——} \\ \text{——} & t_j - \tau_{ku_j} & \text{——} \end{bmatrix}. \tag{9}$$

We take the logarithmic representation to normalize the values of the states that could vary by orders of magnitude. The logarithmic bases, c_1, c_2, c_3, are chosen to normalize the numerical values in a proper range. The output sequence, denoted by $\mathbf{Y}$ is a T-dimensional vector with categorical elements from $K+1$ classes. The structure of the NN is illustrated in Fig. 4, where the input sequence feeds into the two BiLSTM layers, and a fully connected layer with a softmax activation function before the classification output.

With a well-trained NN, the user scheduling output can be easily obtained by inputting the system state sequence. The classification output $\mathbf{Y}$ gives the *prescheduling* variable $\hat{y}_{kj}$, $\forall k, j$. For $\hat{y}_{kj} = 1$, a packet availability variable v_{kj} can be determined by checking if an update packet is available at t_{j-1}, based on the constraints that the packet arrival timestamp needs to be smaller than t_{j-1} and each packet can be either sent only once or dropped. $v_{kj} = 1$ indicates there is an update to send at slot j and $v_{kj} = 0$ otherwise. Then, the actual scheduling variable at each time slot is obtained by further checking the energy causality constraint (7d), and the power allocation p_{kj} can be calculated by the SNR constraint (7e). Let q_{kj} denote the required energy for a successful update, i.e., $q_{kj} = \frac{\gamma_k^*\sigma^2}{h_{kj}}$. Therefore, we have

$$y_{kj} = \begin{cases} v_{kj}\mathbf{1}_{\hat{E}_{kj}\geq 0}, & \text{if } \hat{y}_{kj} = 1 \text{ and } q_{kj} \leq p_{\max}, \\ 0, & \text{otherwise,} \end{cases} \tag{10}$$

$$p_{kj} = y_{kj}q_{kj}, \tag{11}$$

where

$$\hat{E}_{kj} = E_{k,j-1} - q_{kj} \tag{12}$$

$$= e_{k0} + \sum_{i=1}^{j-1}\Big(e_{ki}(1-y_{ki}) - p_{ki}\Big) - q_{kj}, \tag{13}$$

and $\mathbf{1}_\alpha$ denotes the indicator function of α, that $\mathbf{1}_\alpha = 1$ if α is true and $\mathbf{1}_\alpha = 0$ otherwise. Hence, the AoI can be obtained by

$$a_{kj} = y_{kj}(t_j - \tau_{ku_j}) + (1-y_{kj})(a_{k,j-1}+1). \tag{14}$$

By this learning-based algorithm, we are able to determine near-optimal schedules and achieve computational energy efficiency by avoiding the branch-and-bound search implemented by the optimization solver, as will be demonstrated in Section V.

IV. Online AoI Minimization

In the online setting, we consider user scheduling for AoI minimization based on the causal knowledge of the system state information. In contrast with knowing the channel gain, the harvested energy, and the packet time stamps for T slots apriori as in offline optimization, the user scheduling decision is made at each slot with the past and the current states available. Assuming the well-defined transition probabilities, we first formulate the sequential decision making problem as a Markov decision process (MDP). The MDP formulation serves as a benchmark and helps motivate the proposed approach in the sequel that is based on the actor-critic method.

A. Markov Decision Process

We aim to derive an online policy that sequentially schedules status updates over time to minimize the long-term average AoI of the system. MDP is defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathbb{P}, r)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ denotes the set of actions, $\mathbb{P}$ is the transition model that specifies the probability from the one state to another given an action is taken, and r is the immediate reward for taking an action at a certain state. These are specified for our system as follows.

State: The system state at the beginning of each time slot, denoted by $S_j \in \mathcal{S}$, consists of the AoI, the packet waiting time, the required energy, and the available energy for all users, i.e., $S_j = (\mathbf{a}_{j-1}, \mathbf{w}_j, \mathbf{q}_j, \mathbf{E}_{j-1})$, where $\mathbf{a}_{j-1}$ and $\mathbf{E}_{j-1}$ are the vectors with entries $a_{k,j-1}$ and $E_{k,j-1}$ for $k = 1, 2, \dots, K$ given in (2) and (5), respectively. $\mathbf{w}_j$ defines the vector of packet waiting times at the beginning of slot j, whose entries are

$$w_{kj} = \begin{cases} t_{j-1} - \tau_{ku_j}, & \text{if an update packet is buffered} \\ & \text{at user } k \text{ at } t_{j-1}, \\ -1, & \text{if no update packet is available} \\ & \text{at user } k \text{ at } t_{j-1}, \end{cases} \tag{15}$$

for $k = 1, 2, \dots, K$. Vector $\mathbf{q}_j$ denotes the required energy by all users for successful updates at slot j, i.e., $q_{kj} = \frac{\gamma_k^* \sigma^2}{h_{kj}}$. Note that the state space $\mathcal{S}$ is infinite (the states are continuous-valued).

Action: A_j in each slot is the index of the scheduled user, i.e., $A_j \in \mathcal{A} = \{0, 1, 2, \dots, K\}$; $A_j = 0$ implies no one is scheduled as we consider at most one user is scheduled per slot.

Reward: The immediate reward is the negative of the user-averaged AoI of the system, since we aim to minimize AoI. More specifically, given state $S = (\mathbf{a}, \mathbf{w}, \mathbf{q}, \mathbf{E}) \in \mathcal{S}$, if action $A \in \mathcal{A}$ is taken and the next state is $S' = (\mathbf{a}', \mathbf{w}', \mathbf{q}', \mathbf{E}') \in \mathcal{S}$, the immediate reward is defined as

$$r(S, A) = -\frac{1}{K} \sum_{k=1}^{K} a'_k. \tag{16}$$

Transition Probability: The transition probability of reaching state S' from state S by taking action A, denoted by $\mathbb{P}(S'|S, A)$, defines the dynamics of the system, where the transition depends only on S but not the history of the earlier states. Note that action A_j at slot j gives the pre-scheduling variables $\hat{y}_{kj}$ for $k = 1, 2, \dots, K$, as defined in Section III-B. Taking action A_j on the system results in going through the steps in (10)-(14) so that the next state is determined.

The goal is to maximize the cumulative reward in the long run. Here, we consider the sum of the discounted rewards from a starting slot onward, i.e., $G_j = \sum_{k=j}^{\infty} \beta^{k-j} r(S_k, A_k)$, where $\beta \in (0, 1)$ is the discount rate. As β approaches 1, the future rewards are more relevant and the discounted return becomes more farsighted [44]. For any given state, the policy specifies the action, i.e., the mapping from the state space to the action space, denoted by $\pi : \mathcal{S} \to \mathcal{A}$. The value function is the measure of "how good" to be in a state or to perform an action in a state under a given policy. Mathematically, the state-value function is the expected return given an initial state, and the action-value function is the expected return taking an action at a given state, which are defined as $V_\pi(S) = \mathbb{E}_\pi[G_j | S_j = S]$ and $Q_\pi(S, A) = \mathbb{E}_\pi[G_j | S_j = S, A_j = A]$, respectively. The objective is to find the optimal policy π^* that enables the system to act in the way that maximizes the expected discounted return, i.e.,

$$\pi^* = \operatorname*{argmax}_{\pi} V_\pi(S), \ \forall S \in \mathcal{S}. \tag{17}$$

The optimal value functions, $V_{\pi^*}(S)$ and $Q_{\pi^*}(S, A)$, are the values achieved by the optimal policy, which satisfy the Bellman equations [44].

$$V_{\pi^*}(S) = \max_A Q_{\pi^*}(S, A) \tag{18}$$

$$= \max_A \sum_{S'} \mathbb{P}(S'|S, A) \big[r(S, A) + \beta V_{\pi^*}(S') \big], \tag{19}$$

$$Q_{\pi^*}(S, A) = \sum_{S'} \mathbb{P}(S'|S, A) \big[r(S, A) + \beta \max_{A'} Q_{\pi^*}(S', A') \big]. \tag{20}$$

The optimal action-value function, namely Q-function, determines the optimal action at each state directly by

$$\pi^*(S) = \operatorname*{argmax}_{A \in \mathcal{A}} Q_{\pi^*}(S, A), \ \forall S \in \mathcal{S}. \tag{21}$$

Aside from the curse of dimensionality that MDP suffers from, we note that this approach relies on the statistics of the state variables. We wish to consider the general case that *does not assume any statistics* of the random processes of the energy harvesting nor the packet arrivals. Without an explicit model of the system dynamics, the agent, which is the controller of the system, can learn from the interaction with the environment, i.e., the system, over a sequence of time slots. Hence, a reinforcement learning problem with the need for model-free methods naturally arises. Additionally, the infinite states force us to approximate the value function using the more compact parameterized function representations, especially the NN function approximation. The NN with layers of neurons and connections ensures the approximate input-output mapping for the value functions due to its universal nature. Reinforcement learning with the NN function approximation approach leads to the framework of deep reinforcement learning (DRL). Next, we address the online scheduling problem using DRL.

B. Actor-Critic Deep Reinforcement Learning Algorithm

We use actor-critic deep reinforcement learning [44], which is a model-free on-policy method combining the learning techniques of both the value-based and the policy-based methods. In value-based reinforcement learning methods, for instance, SARSA [44], the Q-function is learned as an approximate solution to the Bellman equation and the policy is derived from the Q-function as given in (21). On the other hand, in policy-based learning algorithms, a policy function that maps the state observations to actions is directly learned to identify the probability of taking action A at state S, denoted $\pi(A|S; \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is the parameter of the policy function. The method seeks to maximize the expected return $\mathbb{E}[G]$ of the policy, and thus updates the parameter in the direction that increases the probabilities of actions that achieve higher rewards.

The value function can be viewed as a critic to evaluate the quality of the policy, while the policy function is the actor that indicates how to act in a certain state. The actor-critic methods merge the two classes in the way that the policy function and the value function are learned simultaneously, and the policy is improved from the feedback given by the value-function

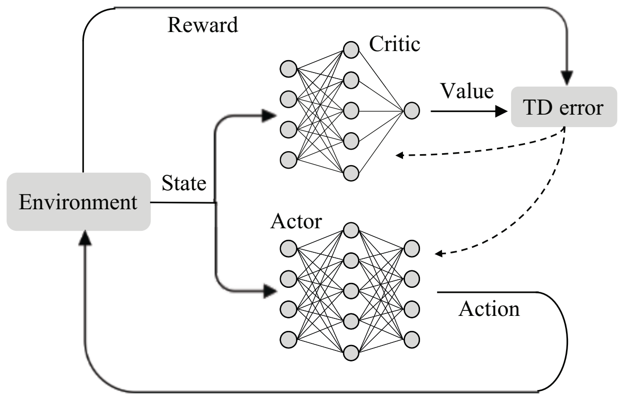

Fig. 5. The schematics of the actor-critic deep reinforcement learning algorithm. The dashed lines indicate the parameter update of the actor and the critic network.

instead of the rewards directly. Thus, it inherits the advantage of low-variance gradients and accelerates learning [56].

Here, we focus on the advantage actor-critic (A2C) algorithm [44], [57]. As shown in Fig. 5, the actor is a network parameterized by $\boldsymbol{\theta}_a$, which consists of an input layer of $4K$ neurons and a softmax output layer with hidden layers in between. The actor network maps the system state observation S_j to the action probability distribution $\pi(A_j|S_j;\boldsymbol{\theta}_a)$, and schedules an user based on the estimated probabilities at each slot. The critic is a second network that approximates the state-value function with parameter $\boldsymbol{\theta}_c$. Its input layer has the same number of neurons for the $4K$-dimensional state input, and the estimated state value is given by the output layer with one neuron. In the training stage, at each step, the agent first interacts with the environment for $j_{\max}$ slots following the current policy given by the actor, which generates $j_{\max}$ states, actions, and rewards from the current slot looking ahead. Then, the critic evaluates the policy by the $j_{\max}$-step temporary-difference (TD) error of the value function:

$$\delta_j = G_{j:j+j_{\max}} - V(S_j;\boldsymbol{\theta}_c). \tag{22}$$

Specifically, $G_{j:j+j_{\max}}$ is the sum of the discounted rewards for $j_{\max}$ steps and the estimated value for the future steps, which is given by

$$G_{j:j+j_{\max}} = \sum_{k=j}^{j+j_{\max}-1} \beta^{k-j} r(S_k,A_k) + \beta^{j_{\max}} V(S_{j+j_{\max}};\boldsymbol{\theta}_c). \tag{23}$$

The parameters of the actor and the critic networks, $\boldsymbol{\theta}_a$ and $\boldsymbol{\theta}_c$, are updated in the direction of maximizing the expected return and minimizing the TD error, respectively, where the gradients are given by

$$d\boldsymbol{\theta}_a = \sum_{j=1}^{j_{\max}} \delta_j \nabla_{\boldsymbol{\theta}_a} \ln \pi(A_j|S_j;\boldsymbol{\theta}_a), \tag{24}$$

$$d\boldsymbol{\theta}_c = \sum_{j=1}^{j_{\max}} \delta_j \nabla_{\boldsymbol{\theta}_c} V(S_j;\boldsymbol{\theta}_c), \tag{25}$$

The pseudocode for the training procedure is summarized in Algorithm 1. With a well-trained actor-critic agent, the user scheduling action A_j is determined by the actor alone based on the input state S_j, and performing the action on the system gives the next state. The favorable performance of the proposed approach is demonstrated in Section V.

Algorithm 1 The Training Procedure of the Actor-Critic Deep Reinforcement Learning Agent for User scheduling

```
Set the episode number $N$, the episode length $T$, the looking ahead step number $j_{\max}$, and the learning rate $\eta_a$, $\eta_c$.
Initialize the parameters for the actor network and the critic network, $\boldsymbol{\theta}_a$, $\boldsymbol{\theta}_c$.
for episode $n = 1, \ldots, N$ do
  Reset the environment and initialize state $S_1$
  for step $j = 1, \ldots, T$ do
    for step $i = j, \ldots, j + j_{\max} - 1$ do
      Generate an user pre-scheduling action $A_i$ by following the current policy $\pi(\cdot|S_i;\boldsymbol{\theta}_a)$;
      Take the action $A_i$, observe the next state $S_{i+1}$ according to (10)-(14), and obtain the reward $r(A_i, S_i)$ by (16);
      Compute the estimated value $V(S_i;\boldsymbol{\theta}_c)$ of the critic network.
    end for
    Based on $\{S_i, A_i, r(A_i,S_i), S_{i+1}\}_{i=j}^{j+j_{\max}-1}$, calculate the the return $G_{j:j_{\max}}$ given by (23).
    Compute the $j_{\max}$-step TD error: $\delta_j = G_{j:j_{\max}} - V(S_j;\boldsymbol{\theta}_c)$.
    Calculate the accumulated gradients $d\boldsymbol{\theta}_a$ and $d\boldsymbol{\theta}_c$ by (24) and (25).
    Update the parameter of the actor network: $\boldsymbol{\theta}_a \leftarrow \boldsymbol{\theta}_a + \eta_a\, d\boldsymbol{\theta}_a$.
    Update the parameter of the critic network: $\boldsymbol{\theta}_c \leftarrow \boldsymbol{\theta}_c + \eta_c\, d\boldsymbol{\theta}_c$.
  end for
end for
```

V. Experimental results

In this section, we demonstrate the AoI performance of the proposed algorithms for different system parameters. Each user and its associated receiver are located randomly and separated by a uniformly distributed distance in $[10, 200]$ meters. The path loss exponent is assumed to be 4. The channel gain is considered to be the product of the path loss and the Rayleigh fading, where the fading coefficients are complex Gaussian variables with zero mean and unit variance, i.e., $\mathcal{CN}(0,1)$. The noise power is -71 dBm. The maximum power is set to be $p_{\max} = 0.1$ watt. The initial age and the initial energy are the same for all users, which are $a_{k0} = 1$ and $e_{k0} = 0.01$ joule for all k. The harvested energy for each user k in slot j e_{kj} is exponential with mean 1 mJ, and is i.i.d. for all users and slots. Each user switches between the energy harvesting period and the energy unavailable period, where the durations are exponential with mean 5 and 2 slots, respectively. The update packet arrival at each user is a Poisson process and the inter-arrival time between successive packets is exponential. We vary the mean of the inter-arrival time, μ, and the SNR threshold for successful updates, γ^*, to investigate their impact on the performance.

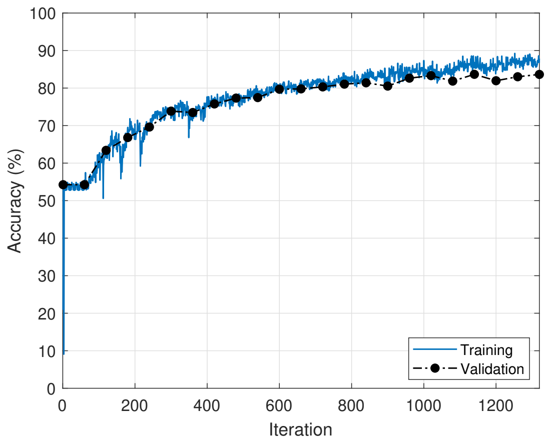

Fig. 6. The training progress of the BiLSTM network for $K = 5$, $\mu = 4$ slots, and $\gamma^* = 15$ dB.

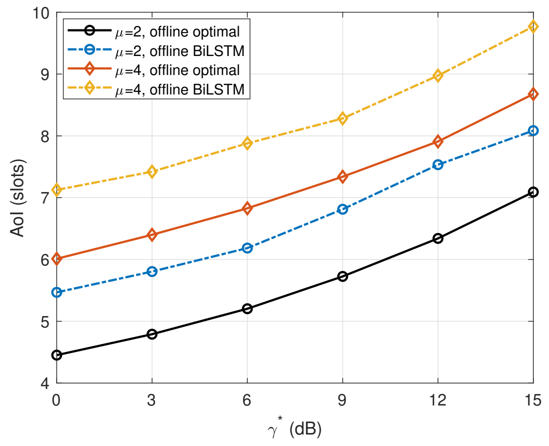

Fig. 7. The offline average AoI versus the update SNR threshold for $K = 5$.

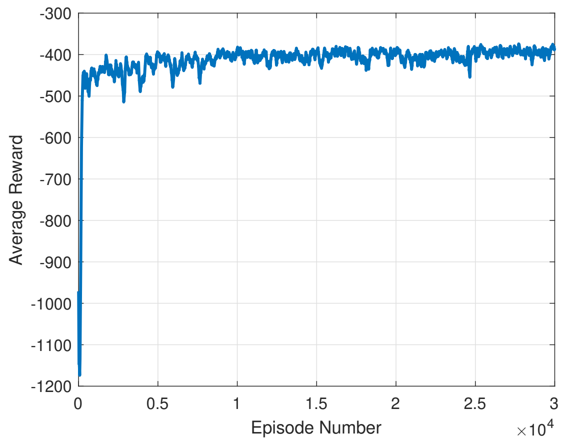

Fig. 8. The training progress of the actor-critic DRL agent for $K = 3$, $\mu = 4$ slots, and $\gamma^* = 3$ dB.

A. Offline Performance

We first present the AoI performance of the offline case for a duration of $T = 50$ slots. The optimal solution is obtained by the CPLEX solver, which provides the performance upper bound for comparison. For the learning-based algorithm, we train the BiLSTM network shown in Fig. 4, where each BiLSTM layer has 60 hidden units and the fully connected layer has $K+1$ neurons. The stochastic gradient descent with momentum (SGDM) optimizer is adopted with momentum 0.9 and the learning rate 0.01. Fig. 6 shows the training process for the offline user scheduling for $K = 5$, $\mu = 4$ slots, and $\gamma^* = 15$ dB. The logarithmic bases are chosen to be $c_1 = c_1 = c_3 = 10$ for data normalization in (9) and the log of zero is set to be a small value, e.g., $1e - 20$. The network is trained on 800 samples with a mini-batch size of 64. The validation is taking on 100 samples for every 60 iteration. The training ends when the validation accuracy cannot be improved anymore, which approaches about 85% in this case. We compare the performance of the learning-based user scheduling with the optimal AoI in Fig. 7 for different values of γ^* and μ. For this case of $K = 5$, the learning-based algorithm achieves the near-optimal performance with a gap of about one slot to the optimum. The AoI increases with the growing SNR threshold since the users wait longer to accumulate sufficient energy for a successful update and thus update less promptly. Similarly, when the packet inter-arrival time becomes larger, the updates are sent less frequently which results in a higher AoI.

B. Online Performance

We next focus on the AoI performance for online user scheduling by the actor-critic DRL algorithm. The actor-critic network is trained by the simulated data as in the offline case. We choose the episode length to be $T = 50$ intended to compare with the offline results, and the number of steps to look ahead for the TD error is $j_{\max} = 50$. The discount factor β is 0.99. The actor network consists of three fully connected layers, the first layer with $4K$ neurons for the input vector of the system state, the second layer with 64 neurons and the ReLU activation function, and the third layer of $K+1$ neurons with the softmax activation function to output the probability distribution of the action space. For the critic network, the input and the hidden layer have the same structure as the actor, followed by the output layer of one neuron to compute the state value. The two networks are trained simultaneously, while the actor evaluates its policy based on the state value learned by the critic. Thus, the learning rate of the actor is set to be $\eta_a = 0.001$, which is smaller than the learning rate of the critic $\eta_c = 0.005$, to allow the actor to converge slower. We use the Adam optimizer. To stabilize the learning process, the gradient is clipped to 1 if it exceeds this threshold. Fig. 8 records the training process for the first 30000 episodes, where the reward is averaged over 100 episodes for illustration. We notice that the agent usually learns fast with a rapidly growing average reward at the beginning stage and then improves the reward gradually and stably with small fluctuations over long training time. We next show the AoI results of the online user scheduling solved by the actor-critic agent trained over 50000 to 80000 episodes for each system setup.

In Fig. 9, the online performance is compared to the

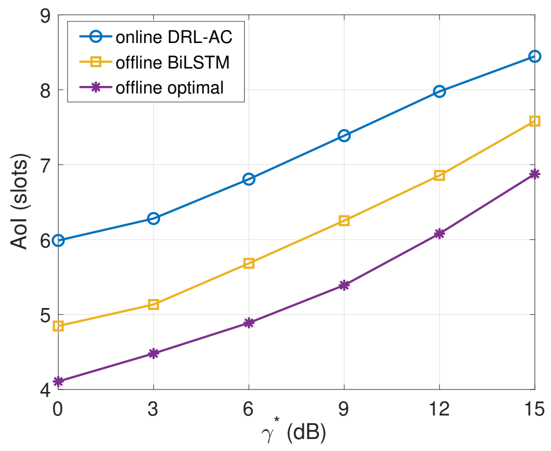

Fig. 9. The average AoI versus the update SNR threshold for $K = 3$ and $\mu = 2$ slots.

offline results. Varying the update SNR threshold, we show the average AoI obtained by the offline MILP optimization, the offline learning-based algorithm, the offline rate-optimal policy, and the online DRL algorithm in Fig. 9 for $K = 3$ and $\mu = 2$. For the online result, the user scheduling action A_j is determined by the trained actor given the current state S_j at each slot, and the agent, i.e., the scheduler of the system, performs the action. We mimic the interaction with the system by going through (10)-(14), where $\hat{y}_{kj}$ is directly given by the action A_j. As shown in Fig. 9, the AoI for the online user scheduling is comparable to the results of the offline optimal and the offline BiLSTM supervised learning even based on the causal system state observations.

In Table. I, the AoI for different number of users, K, are presented given fixed values for μ and γ^*. We observe that the learning algorithms for both the offline and the online scheduling are more likely to achieve the near-optimal performance for the system with a small number of users. This is expected, as the number of users grows, the size of the system state vector is larger, which challenges learning approaches. In particular, for $K = 10$, the number of hidden units of the BiLSTM layers is increased to 90 for the offline learning-based user scheduling. For $K = 1$ and $K = 3$, the hidden layer of the actor network for the online scheduling agent is also omitted to simplify the structure of the network and shorten the training process. However, we also note that for larger networks, the learning based approaches offer feasibility of near-optimal policies.

The average test runtime (seconds) for an episode of 50 slots is also listed in Table. I. The average computation time for the offline MILP optimization by the CPLEX solver increases significantly with the number of users since the complexity increases exponentially with the size of the optimization problem. On the other hand, the average testing time by the BiLSTM solver or the actor-critic DRL solver does not vary much.

Online scheduling by the actor-critic DRL solver takes longer for the interactions over slots as expected, while the offline scheduling by the BiLSTM solver is obtained faster

TABLE I
AVERAGE PERFORMANCE FOR DIFFERENT NUMBERS OF USERS

K	**CPLEX Solver**		**BiLSTM solver**		**AC DRL solver**	
	AoI	Runtime	AoI	Runtime	AoI	Runtime
1	5.89	0.0972	6.39	0.0017	6.90	0.0372
3	6.02	0.9181	6.71	0.0035	7.54	0.0388
5	6.40	2.4279	7.42	0.0054	9.05	0.0407
10	7.26	11.7344	9.55	0.0100	13.62	0.0518

at one shot for the whole episode. Both machine learning methods are orders of magnitude faster compared to CPLEX. Although the computation task of learning solvers is dominated by the training process which could take hundred times of testing time, in contrast to the optimal solver which would need to run at every instance the system state changes, learning-based solvers only need to be trained occasionally to adjust the mapping relationship between the input system state and the output transmission policy. The relative insensitivity of the algorithm performance to precise knowledge in the offline system knowledge suggests coarse estimates may suffice to result in comparable age-based policies to those with perfect knowledge. Overall, the proposed machine learning approaches provide more robust and energy efficient solvers with less sensitivity to system variations and less computation time for comparable system performance.

VI. Conclusion

In this paper, we have considered user scheduling, i.e., transmission times and powers, for AoI minimization in energy harvesting networks. Both problems are computationally hard to solve and make excellent candidates to utilize machine learning approaches. In the offline setting, we have provided the MILP formulation for the optimal scheduling policy and proposed a learning-based near-optimal algorithm by implementing a recurrent neural network. The BiLSTM network that can learn from the time series data bidirectionally is employed to solve the scheduling subproblem. The near-optimal performance of the proposed algorithm verifies the effectiveness of the BiLSTM network on user scheduling. The proposed near-optimal algorithm also shows better scalability in terms of runtime.

For the online setting, we have proposed a model-free reinforcement learning problem for sequential user scheduling based on the MDP formulation. The actor-critic DRL algorithm is adopted to solve the online scheduling, where the actor network and the critic network are trained to approximate the policy function and the value function, respectively. The online scheduling by the DRL algorithm is shown to achieve comparable performance to the offline results with a stable runtime. Furthermore, the runtime is orders of magnitude better than the optimization solver, which suggests energy efficiency of the DRL based solution.

We have shown in this paper that learning can be a viable alternative to optimization relaxation or approximation methods in order to find near-optimal solutions to computationally hard MILP problems, by demonstrating the efficacy of two machine learning approaches in user scheduling for AoI minimization in energy harvesting networks. The proposed learning approach can be applied to moderate-size wireless systems to achieve near-optimality with modest computational time. For very large scale systems (the feature space of system state grows as the number of nodes increases), distributed learning facilitated by wireless edge devices [58] may need to be considered. Future directions in learning and freshness information include, explicitly incorporating the computation times into the problem of learning to transmit fresh information, considering systems where nodes have correlated harvested energy, and investigating decentralized learning for the user scheduling problem with AoI as a metric.

References

[1] S. Leng and A. Yener, "An actor-critic reinforcement learning approach to minimum age of information scheduling in energy harvesting networks," in *ICASSP 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 8128–8132.

[2] ——, "Learning to transmit fresh information in energy harvesting networks using supervised learning," in *Asilomar Conference on Signals, Systems, and Computers*, 2021.

[3] Y. Sun, I. Kadota, R. Talak, and E. Modiano, "Age of information: A new metric for information freshness," *Synthesis Lectures on Communication Networks*, vol. 12, no. 2, pp. 1–224, 2019.

[4] M. A. Abd-Elmagid, N. Pappas, and H. S. Dhillon, "On the role of age of information in the internet of things," *IEEE Communications Magazine*, vol. 57, no. 12, pp. 72–77, 2019.

[5] A. Kosta, N. Pappas, V. Angelakis *et al.*, "Age of information: A new concept, metric, and tool," *Foundations and Trends® in Networking*, vol. 12, no. 3, pp. 162–259, 2017.

[6] S. Kaul, M. Gruteser, V. Rai, and J. Kenney, "Minimizing age of information in vehicular networks," in *Proceedings of 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, 2011, pp. 350–358.

[7] S. Kaul, R. Yates, and M. Gruteser, "Real-time status: How often should one update?" in *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, 2012, pp. 2731–2735.

[8] S. K. Kaul, R. D. Yates, and M. Gruteser, "Status updates through queues," in *Proc. 46th Annual Conference on Information Sciences and Systems (CISS)*, 2012.

[9] R. D. Yates and S. K. Kaul, "The age of information: Real-time status updating by multiple sources," *IEEE Transactions on Information Theory*, vol. 65, no. 3, pp. 1807–1827, 2019.

[10] C. Kam, S. Kompella, G. D. Nguyen, and A. Ephremides, "Effect of message transmission path diversity on status age," *IEEE Transactions on Information Theory*, vol. 62, no. 3, pp. 1360–1374, 2016.

[11] C. Kam, S. Kompella, G. D. Nguyen, J. E. Wieselthier, and A. Ephremides, "On the age of information with packet deadlines," *IEEE Transactions on Information Theory*, vol. 64, no. 9, pp. 6419–6428, 2018.

[12] R. D. Yates, "The age of information in networks: Moments, distributions, and sampling," *arXiv preprint arXiv:1806.03487*, 2018.

[13] Y. Inoue, H. Masuyama, T. Takine, and T. Tanaka, "A general formula for the stationary distribution of the age of information and its application to single-server queues," *IEEE Transactions on Information Theory*, vol. 65, no. 12, pp. 8305–8324, 2019.

[14] A. Soysal and S. Ulukus, "Age of information in G/G/1/1 systems: Age expressions, bounds, special cases, and optimization," *IEEE Transactions on Information Theory*, vol. 67, no. 11, pp. 7477–7489, 2021.

[15] Y. Sun, E. Uysal-Biyikoglu, R. D. Yates, C. E. Koksal, and N. B. Shroff, "Update or wait: How to keep your data fresh," *IEEE Transactions on Information Theory*, vol. 63, no. 11, pp. 7492–7508, 2017.

[16] A. M. Bedewy, Y. Sun, and N. B. Shroff, "Minimizing the age of information through queues," *IEEE Transactions on Information Theory*, vol. 65, no. 8, pp. 5215–5232, 2019.

[17] B. M. Leiner, D. L. Nielson, and F. A. Tobagi, "Issues in packet radio network design," *Proceedings of the IEEE*, vol. 75, no. 1, pp. 6–20, 1987.

[18] L. Kleinrock and J. Silvester, "Spatial reuse in multihop packet radio networks," *Proceedings of the IEEE*, vol. 75, no. 1, pp. 156–167, 1987.

[19] Q. He, D. Yuan, and A. Ephremides, "Optimal link scheduling for age minimization in wireless systems," *IEEE Transactions on Information Theory*, vol. 64, no. 7, pp. 5381–5394, 2018.

[20] I. Kadota, A. Sinha, E. Uysal-Biyikoglu, R. Singh, and E. Modiano, "Scheduling policies for minimizing age of information in broadcast wireless networks," *IEEE/ACM Transactions on Networking*, vol. 26, no. 6, pp. 2637–2650, 2018.

[21] R. Talak, S. Karaman, and E. Modiano, "Optimizing information freshness in wireless networks under general interference constraints," in *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2018, pp. 61–70.

[22] ——, "Optimizing age of information in wireless networks with perfect channel state information," in *Proceedings of 16th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2018.

[23] R. Talak, I. Kadota, S. Karaman, and E. Modiano, "Scheduling policies for age minimization in wireless networks with unknown channel state," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2018.

[24] Z. Jiang, B. Krishnamachari, X. Zheng, S. Zhou, and Z. Niu, "Timely status update in wireless uplinks: Analytical solutions with asymptotic optimality," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3885–3898, 2019.

[25] R. D. Yates, "Lazy is timely: Status updates by an energy harvesting source," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, 2015, pp. 3008–3012.

[26] B. T. Bacinoglu, E. T. Ceran, and E. Uysal-Biyikoglu, "Age of information under energy replenishment constraints," in *Proceedings of IEEE Information Theory Workshop (ITW)*, 2015, pp. 25–31.

[27] X. Wu, J. Yang, and J. Wu, "Optimal status update for age of information minimization with an energy harvesting source," *IEEE Transactions on Green Communications and Networking*, vol. 2, no. 1, pp. 193–204, 2018.

[28] A. Arafa, J. Yang, S. Ulukus, and H. V. Poor, "Age-minimal transmission for energy harvesting sensors with finite batteries: Online policies," *IEEE Transactions on Information Theory*, vol. 66, no. 1, pp. 534–556, 2020.

[29] A. Arafa and S. Ulukus, "Age minimization in energy harvesting communications: Energy-controlled delays," in *2017 51st Asilomar conference on signals, systems, and computers*, 2017, pp. 1801–1805.

[30] ——, "Timely updates in energy harvesting two-hop networks: Offline and online policies," *IEEE Transactions on Wireless Communications*, vol. 18, no. 8, pp. 4017–4030, 2019.

[31] A. Arafa, J. Yang, S. Ulukus, and H. V. Poor, "Timely status updating over erasure channels using an energy harvesting sensor: Single and multiple sources," *IEEE Transactions on Green Communications and Networking*, 2021.

[32] S. Feng and J. Yang, "Age of information minimization for an energy harvesting source with updating erasures: Without and with feedback," *IEEE Transactions on Communications*, vol. 69, no. 8, pp. 5091–5105, 2021.

[33] S. Leng and A. Yener, "Age of information minimization for an energy harvesting cognitive radio," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 2, pp. 427–439, 2019.

[34] X. Zheng, S. Zhou, Z. Jiang, and Z. Niu, "Closed-form analysis of non-linear age of information in status updates with an energy harvesting transmitter," *IEEE Transactions on Wireless Communications*, vol. 18, no. 8, pp. 4129–4142, 2019.

[35] Y. Dong, Z. Chen, and P. Fan, "Uplink age of information of unilaterally powered two-way data exchanging systems," in *Proceedings of IEEE International Conference on Computer Communications (INFOCOM) Workshops*, 2018, pp. 559–564.

[36] I. Krikidis, "Average age of information in wireless powered sensor networks," *IEEE Communications Letters*, vol. 8, no. 2, pp. 628–631, 2019.

[37] Z. Chen, N. Pappas, E. Björnson, and E. G. Larsson, "Age of information in a multiple access channel with heterogeneous traffic and an energy harvesting node," in *Proceedings of IEEE International Conference on Computer Communications (INFOCOM) Workshops*, 2019.

[38] O. M. Sleem, S. Leng, and A. Yener, "Age of information minimization in wireless powered stochastic energy harvesting networks," in *Proceedings of the 54th Annual Conference on Information Sciences and Systems (CISS)*, 2020.

[39] R. D. Yates, Y. Sun, D. R. Brown, S. K. Kaul, E. Modiano, and S. Ulukus, "Age of information: An introduction and survey," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 5, pp. 1183–1210, 2021.

[40] D. Gündüz, P. de Kerret, N. D. Sidiropoulos, D. Gesbert, C. R. Murthy, and M. van der Schaar, "Machine learning in the air," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2184–2199, 2019.

[41] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019.

[42] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of the Advances in neural information processing systems*, 2014, pp. 3104–3112.

[43] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[44] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[45] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[46] E. T. Ceran, D. Gündüz, and A. György, "Average age of information with hybrid ARQ under a resource constraint," *IEEE Transactions on Wireless Communications*, vol. 18, no. 3, pp. 1900–1913, 2019.

[47] S. Leng and A. Yener, "Age of information minimization for wireless ad hoc networks: A deep reinforcement learning approach," in *IEEE Global Communications Conference (GLOBECOM)*, 2019.

[48] M. A. Abd-Elmagid, H. S. Dhillon, and N. Pappas, "A reinforcement learning framework for optimizing age-of-information in RF-powered communication systems," *IEEE Transactions on Communications*, May 2020.

[49] CPLEX optimizer. [Online]. Available: https://www.ibm.com/analytics/cplex-optimizer

[50] J. Lee and S. Leyffer, *Mixed integer nonlinear programming*. Springer Science & Business Media, 2011, vol. 154.

[51] Sequence-to-sequence classification using deep learning. [Online]. Available: https://www.mathworks.com/help/deeplearning/ug/sequence-to-sequence-classification-using-deep-learning.html

[52] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[53] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional LSTM networks for improved phoneme classification and recognition," in *Proceedings of the International Conference on Artificial Neural Networks*. Springer, 2005, pp. 799–804.

[54] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[55] M. Hausknecht and P. Stone, "Deep recurrent Q-learning for partially observable MDPs," in *Proceedings of the AAAI Fall Symposium Series*, 2015.

[56] I. Grondman, L. Busoniu, G. A. Lopes, and R. Babuska, "A survey of actor-critic reinforcement learning: Standard and natural policy gradients," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1291–1307, 2012.

[57] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proceedings of the International conference on machine learning*, 2016, pp. 1928–1937.

[58] J. Mao, H. Yang, P. Qiu, J. Liu, and A. Yener, "CHARLES: Channel-quality-adaptive over-the-air federated learning over wireless networks," in *IEEE 23rd International Workshop on Signal Processing Advances in Wireless Communication (SPAWC)*, 2022.

Aylin Yener (S'91–M'01–SM'14–F'15) received the B.Sc. degree in electrical and electronics engineering and the B.Sc. degree in physics from Bogazici University, Istanbul, Turkey, and the M.S. and Ph.D. degrees in electrical and computer engineering from Rutgers University, New Brunswick, NJ, USA. She is the Roy and Lois Chope Chair of Engineering with The Ohio State University, Columbus, OH, USA, and is a Professor of Electrical and Computer Engineering, Computer Science and Engineering, and Integrated Systems Engineering. Prior to joining Ohio State in 2020, she was a University Distinguished Professor of Electrical Engineering and a Dean's Fellow with The Pennsylvania State University, University Park, PA, USA, where she joined the faculty as an Assistant Professor in 2002. She was a Visiting Professor of Electrical Engineering with Stanford University from 2016 to 2018, where he was a Visiting Associate Professor from 2008 to 2009. She was a Visiting Researcher with Telecom Paris Tech in 2016. Her current research interests are in information security and privacy, green communications, caching, 6G, and more generally in the fields of information theory, communication theory, and networked systems. She received the NSF CAREER Award in 2003, the Best Paper Award in Communication Theory from the IEEE International Conference on Communications in 2010, the Penn State Engineering Alumni Society (PSEAS) Outstanding Research Award in 2010, the IEEE Marconi Prize Paper Award in 2014, the PSEAS Premier Research Award in 2014, the Leonard A. Doggett Award for Outstanding Writing in Electrical Engineering at Penn State in 2014, the IEEE Women in Communications Engineering Outstanding Achievement Award in 2018, the IEEE Communications Society Best Tutorial Paper Award in 2019, and the IEEE Communications Society Communication Theory Technical Achievement Award in 2020. She has been a Distinguished Lecturer for the IEEE Information Theory Society (2019–2021), the IEEE Communications Society (2018–2019), and the IEEE Vehicular Technology Society (2017–2021). She is currently serving as the Junior Past President of the IEEE Information Theory Society. She was the President (2020), the Vice President (2019), the Second Vice President (2018), an Elected Member of the Board of Governors (2015–2018), and the Treasurer (2012–2014) of the IEEE Information Theory Society. She served as the Student Committee Chair for the IEEE Information Theory Society (2007–2011), and was the Co-Founder of the Annual School of Information Theory in North America in 2008. She was a Technical (Co)-Chair for various symposia/tracks at the IEEE ICC, PIMRC, VTC, WCNC, and Asilomar in 2005, 2008–2014, and 2018. She served as an Editor for IEEE TRANSACTIONS ON COMMUNICATIONS (2009–2012) and IEEE TRANSACTIONS ON MOBILE COMPUTING (2017–2018), and an Editor and an Editorial Advisory Board Member for IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS (2001–2012). She also served as a Guest Editor for IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY in 2011, and IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS in 2015. She currently serving as a Senior Editor for IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS. She is on the Senior Editorial Board of IEEE JOURNAL ON SELECTED AREAS IN INFORMATION THEORY and is an Area Editor for Security and Privacy for the IEEE TRANSACTIONS ON INFORMATION THEORY.

Shiyang Leng received the B.E. degree in optoelectronic information engineering from Harbin Institute of Technology, Harbin, China, in 2012, the M.Sc. degree in communications and multimedia engineering from Friedrich-Alexander-University Erlangen-Nuremberg, Erlangen, Germany, in 2014, and the Ph.D. degree in Electrical Engineering from The Pennsylvania State University, in 2020. She was a Research Assistant with the Wireless Communications and Networking Laboratory from 2015 to 2020. She is currently with Samsung Research America.