Scheduling Algorithms for Minimizing Age of Information in Wireless Broadcast Networks with Random Arrivals

Yu-Pin Hsu, Eytan Modiano, and Lingjie Duan

Abstract—*Age of information* is a new network performance metric that captures *the freshness of information at end-users*. This paper studies the age of information from a scheduling perspective. To that end, we consider a wireless broadcast network where a base-station (BS) is updating many users on *random* information arrivals under a transmission capacity constraint. For the offline case when the arrival statistics are known to the BS, we develop a *structural MDP scheduling algorithm* and an *index scheduling algorithm*, leveraging Markov decision process (MDP) techniques and the Whittle's methodology for restless bandits. By exploring optimal structural results, we not only reduce the computational complexity of the MDP-based algorithm, but also simplify deriving a closed form of the Whittle index. Moreover, for the online case, we develop an *MDP-based online scheduling algorithm* and an *index-based online scheduling algorithm*. Both the structural MDP scheduling algorithm and the MDP-based online scheduling algorithm asymptotically minimize the average age, while the index scheduling algorithm minimizes the average age when the information arrival rates for all users are the same. Finally, the algorithms are validated via extensive numerical studies.

Index Terms—Age of information, scheduling algorithms, Markov decision processes

1 Introduction

In recent years there has been a growing research interest in an *age of information* [3]. The age of information is motivated by a variety of network applications requiring *timely* information. Examples range from information updates for *network users*, e.g., live traffic, transportation, air quality, and weather, to status updates for *smart systems*, e.g., smart home systems, smart transportation systems, and smart grid systems.

Fig. 1 shows an example network, where network users $u_1, \ldots, u_N$ are running applications that need some timely information (e.g., user u_1 needs both traffic and transportation information for planning the best route), while at some epochs, snapshots of the information are generated at the sources and sent to the users in the form of packets over wired or wireless networks. The users are being updated and keep the latest information only. Since the information at the end-users is expected to be as timely as possible, the age of information is therefore proposed to capture the *freshness of the information at the end-users*; more precisely, it measures the elapsed time since the generation of the information. In addition to the timely information for the network users, the smart systems also need timely status (e.g., locations and velocities in smart transportation systems) to accomplish some tasks (e.g., collision-free smart transportation systems). As such, the age of information is a good metric to evaluate these networks supporting age-sensitive applications.

Next, we characterize the age-sensitive networks in two aspects. First, while packet delay is usually referred to as the elapsed time from the generation to its delivery, the age of information includes not only the packet delay but also the inter-delivery time, because the age of information keeps increasing until the information at the end-users is updated. We hence need to jointly consider the two parameters so as to design an age-optimal network. Moreover, while traditional relays (i.e., intermediate nodes) need to buffer all packets that are not served yet, the relays in the network of Fig. 1 for timely information store the latest information at most and discard out-of-date packets. The buffers for minimizing the age here are no longer as useful as those in traditional relay networks.

In this paper, we consider a wireless broadcast network, where a base-station (BS) is updating many network users on timely information. The new information is *randomly* generated at its source. We assume that the BS can serve at most one user for each transmission opportunity. Under the transmission constraint, a transmission scheduling algorithm manages how the channel resources are allocated for each time, depending on the packet arrivals at the BS and the ages of the information at the users. For example, consider a simple scenario where the BS is serving two users u_1, u_2, and the BS does not buffer an arriving packet if the packet is not served upon arrival. Suppose that the packet arrival rate (at

- *Y.-P. Hsu is with the Department of Communication Engineering, National Taipei University, New Taipei City 23741, Taiwan. E-mail: yupinhsu@mail.ntpu.edu.tw.*
- *E. Modiano is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139. E-mail: modiano@mit.edu.*
- *L. Duan is with the Engineering Systems and Design Pillar, Singapore University of Technology and Design, Singapore. E-mail: lingjie_duan@sutd.edu.sg.*

Manuscript received 18 Sept. 2018; revised 22 Apr. 2019; accepted 6 Aug. 2019. Date of publication 20 Aug. 2019; date of current version 3 Nov. 2020. (Corresponding author: Yu-Pin Hsu.)
Digital Object Identifier no. 10.1109/TMC.2019.2936199

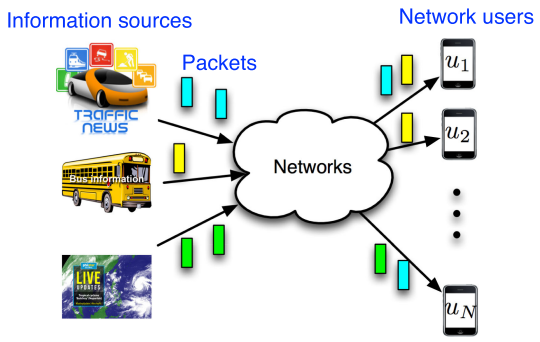

Fig. 1. Timely information updates for network users.

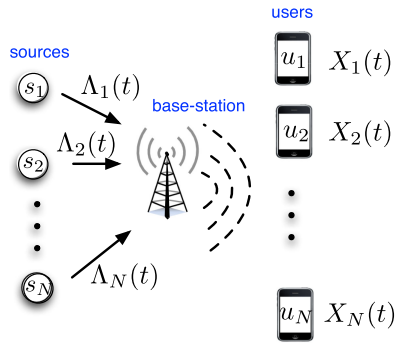

Fig. 2. A BS updates N users $u_1, \cdots, u_N$ on information of sources $s_1, \cdots, s_N$, respectively.

the BS) for user u_1 is higher than that for u_2. When the age of information at u_1 is higher than that at u_2, it is unclear whether updating u_1 (with the largest present age) is an optimal decision (for minimizing the long-run average age), because u_2 (with a lower arrival rate) has to wait for the next arriving packet for a longer time if it misses the chance. This paper aims to develop scheduling algorithms for random arrivals, with the purpose of minimizing the long-run average age.

1.1 Contributions

We study the age-optimal scheduling problem in the wireless broadcast network. Our main contributions lie in designing novel scheduling algorithms and analyzing their age-optimality. For the case when the arrival statistics are available at the BS as prior information, we develop two *offline* scheduling algorithms, leveraging a Markov decision process (MDP) and the Whittle index. However, the MDP and the Whittle index in our problem will be difficult to analyze as they involve *long-run average cost optimization problems* with *infinite state spaces* and *unbounded immediate costs* [4]. Moreover, it is in general very challenging to obtain the Whittle index in closed form. By investigating some *structural results*, we not only successfully resolve the issues but also simplify the calculation of the Whittle index. It turns out that our index scheduling algorithm is very simple. When the arrival statistics are unknown, we develop *online* versions of the two offline algorithms. We show that both offline and online MDP-based scheduling algorithms are asymptotically age-optimal, and the index scheduling algorithm is age-optimal when the information arrival rates for all users are the same. Finally, we compare these algorithms via extensive computer simulations, and further investigate the impact of buffering the latest information. With the numerical results, we can observe that buffers might not be as effective as those in traditional relay networks.

1.2 Related Works

The general idea of *age* was proposed in [5] to study how to refresh a local copy of an autonomous information source to maintain the local copy up-to-date. The age defined in [5] is associated with *discrete* events at the information source, where the age is zero until the source is updated. Differently, the age of information in [3] measures the age of a sample of *continuous* events; therefore, the sample immediately becomes old after generated. Many previous works, e.g., [3], [6], [7], [8], [9], [10], studied the age of information for a single link. The papers [3], [6] considered buffers storing all unserved packets (i.e., out-of-date packets are also stored) and analyzed the long-run average age, based on various queueing models. They showed that neither the throughput-optimal sampling rate nor the delay-optimal sampling rate can minimize the average age. The paper [7] considered a *smart* update and showed that the *always update* scheme might not minimize the average age. Moreover, [8], [9] developed power-efficient updating algorithms for minimizing the average age. The model in [10] considered no buffer or a buffer storing the latest information.

Of the most relevant works on scheduling multiple users are [11], [12], [13], [14], [15]. The works [11], [12], [13] considered buffers storing all out-of-date packets. The paper [14] considered buffers storing the latest information with *periodic* arrivals, while information updates in [15] can be generated *at will*. In contrast, our work is the first to develop both offline and online scheduling algorithms for more challenging *random* information arrivals, with the purpose of minimizing the long-run average age.

We remark that some other recent works [14], [16], [17] leveraged the Whittle index for exploring the age of information. The work in [14] applied the Whittle index for scheduling periodic arrivals; in contrast, our work applied the Whittle index for scheduling random arrivals. The works in [16], [17] followed the conference version [2] of the present work. They investigated distributed scheduling design and joint sampling/scheduling design, respectively, by exploiting the Whittle index.

2 System Overview

2.1 Network Model

We consider a wireless broadcast network in Fig. 2 consisting of a base-station and N wireless users $u_1, \ldots, u_N$. Each user u_i is interested in timely information generated by a source s_i, while the information is transmitted through the BS in the form of packets. We consider a discrete-time system with slots $t = 0, 1, \ldots$ Packets from the sources (if any) arrive at the BS at the *beginning* of each slot. The packet arrivals at the BS for different users are independent of each other and also independent and identically distributed (i.i.d.) over slots, following a Bernoulli distribution. Precisely, by $\Lambda_i(t)$, we indicate if a packet from source s_i arrives at the BS in slot t, where $\Lambda_i(t) = 1$ if there is a packet; otherwise, $\Lambda_i(t) = 0$. We denote the probability $P[\Lambda_i(t) = 1]$ by p_i.

TABLE 1
Notation Table

N	Total number of users
u_i	A user in the network, where $i = 1, \cdots, N$
s_i	Information source for u_i
$\Lambda_i(t)$	Arrival indicator for u_i in slot t
p_i	Arrival rate for u_i
$X_i(t)$	Age of information for u_i in slot t
Δ	The MDP defined in Section 4 for identifying a scheduling algorithm
$\Delta^{(m)}$	The approximate MDP defined in Section 4.2 under the boundary size m
Ω	The MDP defined in Section 5.1 for deriving the Whittle index
$\mathbf{S}(t)$	State for the MDP Δ or for the MDP Ω in slot t
$D(t)$	Scheduling decision or action for the MDP Δ in slot t
π	Scheduling algorithm or policy for the MDP Δ
$C(\mathbf{S}(t), D(t))$	Immediate cost under state $\mathbf{S}(t)$ and action $D(t)$ for the MDP Δ in slot t
$V(\pi)$	Average cost under policy π for the MDP Δ
$V_\alpha(\mathbf{s}; \pi)$	Expected total α-discounted cost under policy π for the MDP Δ when the initial state is $\mathbf{s}$
$V_\alpha(\mathbf{s})$	Minimum expected total α-discounted cost for the MDP Δ when the initial state is $\mathbf{s}$
$h_\alpha(\mathbf{s})$	Relative cost function, i.e., $V_\alpha(\mathbf{s}) - V_\alpha(\mathbf{0})$ where $\mathbf{0}$ is a reference state
$V_{\alpha,n}(\mathbf{s})$	Minimum expected total α-discounted cost for the MDP Δ at the nth iteration in Eq. (4) when the initial state is $\mathbf{s}$
$X_i^{(m)}(t)$	Truncated age of information for u_i in slot t under the boundary size m
$\mathbf{S}^{(m)}(t)$	State for the MDP $\Delta^{(m)}$ in slot t
$V^{(m)}(\mathbf{s})$	Minimum average cost for the MDP $\Delta^{(m)}$ when the initial state is $\mathbf{s}$
$V_n^{(m)}(\mathbf{s})$	Minimum average cost for the MDP $\Delta^{(m)}$ at the nth iteration in Eq. (5) when the initial state is $\mathbf{s}$
$A(t)$	Action for the MDP Ω in slot t
c	Updating cost for the MDP Ω
$C(\mathbf{S}(t), A(t))$	Immediate cost under state $\mathbf{S}(t)$ and action $A(t)$ for the MDP Ω in slot t
μ	Policy for the MDP Ω
$J(\mu)$	Average cost under policy μ for the MDP Ω
$\mathscr{C}(\bar{X})$	Average cost for the MDP Ω with the threshold $\bar{X}$
$I(\mathbf{s})$	Whittle index for state $\mathbf{s}$
$\tilde{s}$	Post-action state
$\tilde{V}^{(m)}(\tilde{s})$	Post-action average cost under post-action state $\tilde{s}$ for the MDP $\Delta^{(m)}$
$\gamma(t)$	Step size for Algorithm 2 in slot t

Suppose that the BS can successfully transmit at most one packet during each slot, i.e., the BS can update at most one user in each slot. The paper focuses on the noiseless channel. See Section 8 for extending to noisy channels. By $D(t) \in \{0, 1, \cdots N\}$ we denote a decision of the BS in slot t, where $D(t) = 0$ if the BS does not transmit any packet and $D(t) = i$ for $i = 1, \ldots, N$ if user u_i is scheduled to be updated in slot t.

This paper focuses on the scenario without buffering arriving packets for future, referred to as a no-buffer network.[1] In the no-buffer network, an arriving packet is discarded if it is not transmitted in the arriving slot. The no-buffer network is not only simple to implement for practical systems, but also was shown to achieve good performance in a single link (see [10]). In Section 7, we will also numerically study networks with buffers.

1. The no-buffer network is analogous to the loss network [18] in queueing theory, where arriving packets that cannot be served upon arrival get lost. In fact, the BS needs to temporarily keep all arriving packets for decision.

2.2 Age of Information Model

We initialize the ages of all arriving packets *at the BS* to be zero. The age of information *at a user* becomes one on receiving a new packet, due to one slot of the transmission time. Let $X_i(t)$ be the age of information at user u_i in slot t *before* the BS makes a scheduling decision. Suppose that the age of information at a user increases linearly with slots if the user is not updated. Then, the dynamics of the age of information for user u_i is

$$X_i(t+1) = \begin{cases} 1 & \text{if } \Lambda_i(t) = 1 \text{ and } D(t) = i; \\ X_i(t) + 1 & \text{else,} \end{cases} \tag{1}$$

where the age of information in the next slot is one if the user gets updated on the new information; otherwise, the age increases by one. Let $\mathbf{X}(t) = (X_1(t), \ldots, X_N(t))$ be the age vector in slot t.

2.3 Problem Formulation

A *scheduling algorithm* $\pi = \{D(0), D(1), \cdots\}$ specifies a transmission decision for each slot. A scheduling algorithm is called an *offline* scheduling algorithm if the BS has arrival statistics p_i for all i as a prior. A scheduling algorithm is called an *online* scheduling algorithm if the BS knows the *present* arrival indicator $\Lambda_i(t)$ only, without having the arrival statistics for inferring future arrivals.

We define the *average age* under scheduling algorithm π by

$$\limsup_{T\to\infty} \frac{1}{T+1} E_\pi \left[\sum_{t=0}^{T} \sum_{i=1}^{N} X_i(t) \right],$$

where E_π represents the conditional expectation, given that scheduling algorithm π is employed. We remark that this paper focuses on the total age of users for delivering clean results; whereas our design and analysis can work perfectly for the *weighted sum* of the ages. Our goal is to develop *age-optimal* scheduling algorithms, defined below.

Definition 1. *A scheduling algorithm π is* age-optimal *if it minimizes the average age.*

In this paper, we will develop two offline scheduling algorithms and two online scheduling algorithms. Leveraging Markov decision process techniques and Whittle's methodology, we develop two offline scheduling algorithms in Sections 4 and 5, respectively, when the arrival statistics are available to the BS; later, two online versions of the offline algorithms are proposed in Section 6 for the case when the arrival statistics are oblivious to the BS. For clarity and continuity, we move some proofs of this paper to the appendices in the supplemental material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TMC.2019.2936199. In addition, please see Table 1 for summarizing the notation set used in the paper.

3 Overview of Main Theoretical Results

In this section, we provide an overview of our main results and their relations. In Section 4, we formulate an MDP for identifying an age-optimal scheduling algorithm. The MDP involves a long-run average cost minimization problem with an infinite state space. We deal with the MDP with its discounted cost case. The underlying idea is that the total discounted cost can be expressed as an iteration form in Proposition 4 (but the long-run average cost might not). Using an induction argument over the iteration from Proposition 4, Lemma 6 shows that there exists an age-optimal scheduling that is *deterministic stationary*; in particular, an age-optimal scheduling is a limit in the discount factor. Using the induction along with the limiting behavior from Lemma 6, Theorem 8 establishes that an age-optimal scheduling has a *switch* structure. In addition, we propose a sequence of finite-state approximate MDPs in Section 4.2, and show in Theorem 9 that the sequence converges to the original MDP. Then, exploiting the switch structure in an approximate finite-state MDP, we propose a structural MDP scheduling algorithm in Section 4.3. The algorithm is asymptotically age-optimal from Theorem 10.

In Section 5, we formulate another MDP for solving the Whittle index. Again, we deal with the MDP with its discounted cost case along with a similar induction argument, and then obtain similar results: Lemma 12 shows that an optimal policy for minimizing a long-run average cost is deterministic stationary and is also a limit in the discount factor; moreover, Theorem 14 establishes that an optimal policy has a *threshold* structure. With the two results, Lemma 15 can explicitly express the long-run average cost in terms of a threshold. By means of the cost expression, we can easily obtain the Whittle index as shown in Theorem 17 and show an *indexability* property in Theorem 20. Then, we propose an index scheduling algorithm in Section 5.4. The algorithm is age-optimal in a relaxed model; in particular, it is exactly age-optimal when arrival rates for all users are identical, as shown in Lemma 21.

Finally, we extend the proposed structural MDP scheduling algorithm and the index scheduling algorithm to the online setting in Section 6. Moreover, an MDP-based online scheduling algorithm is shown to be asymptotically age-optimal in Theorem 22.

4 A Structural MDP Scheduling Algorithm

Our first scheduling algorithm is driven by MDP techniques. According to [19], an MDP is a 4-tuple $\{\text{states, actions, transition probabilities, costs}\}$. We define the MDP $\Delta = \{\text{states, actions, transition probabilities, costs}\}$ with each object being following.

- *States*: We define the state $\mathbf{S}(t)$ of the MDP in slot t by $\mathbf{S}(t) = (X_1(t), \ldots, X_N(t),\ \Lambda_1(t), \ldots, \Lambda_N(t))$. Let $\mathbf{S}$ be the state space consisting of all possible ages and arrival indicators. A detailed description of the state space is in Appendix C, available in the online supplemental material. Note that $\mathbf{S}$ is a *countable infinite* set because the ages are possibly unbounded.
- *Actions*: We define the action of the MDP in slot t to be $D(t)$. Note that the action space is finite.
- *Transition probabilities*: By $P_{\mathbf{s},\mathbf{s}'}(d)$ we denote the transition probability of the MDP from state $\mathbf{s} = (x_1, \ldots, x_N, \lambda_1, \ldots, \lambda_N)$ to state $\mathbf{s}' = (x'_1, \ldots, x'_N, \lambda'_1, \ldots, \lambda'_N)$ under action $D(t) = d$. According to the age dynamics in Eq. (1) and the i.i.d. assumption of the arrivals, we can describe the non-zero $P_{\mathbf{s},\mathbf{s}'}(d)$ as

$$P_{\mathbf{s},\mathbf{s}'}(d) = \prod_{i:\lambda'_i=1} p_i \prod_{i:\lambda'_i=0} (1-p_i),$$

  if $x'_i = (x_i + 1) - x_i \mathbf{1}_{i=d \text{ and } \lambda_i=1}$ for all $i = 1, \ldots, N$, where $\mathbf{1}$ is the indicator function.
- *Costs*: Let $C(\mathbf{S}(t), D(t) = d)$ be the *immediate cost* of the MDP if action $D(t) = d$ is taken in slot t under state $\mathbf{S}(t)$, representing the resulting total age in the next slot:

$$\begin{aligned} C(\mathbf{S}(t), D(t) = d) &\triangleq \sum_{i=1}^{N} X_i(t+1) \\ &= \sum_{i=1}^{N} (X_i(t) + 1) - X_d(t) \cdot \Lambda_d(t), \end{aligned}$$

  where we define $X_0(t) = 0$ and $\Lambda_0(t) = 0$ for all t (for the no update case $d = 0$), while the last term indicates that user u_d is updated in slot t.

The objective of the MDP Δ is to find a *policy* π (with the same definition as the scheduling algorithm) that minimizes the *average cost* $V(\pi)$ defined by

$$V(\pi) = \limsup_{T\to\infty} \frac{1}{T+1} E_\pi \left[\sum_{t=0}^{T} C(\mathbf{S}(t), D(t)) \right]. \tag{2}$$

Definition 2. *A policy π for the MDP Δ is called* Δ-optimal *if it minimizes the average cost $V(\pi)$.*

Then, a Δ-optimal policy is an age-optimal scheduling algorithm. Moreover, policies of the MDP can be classified as follows. A policy of the MDP is *history dependent* if $D(t)$ depends on $D(0), \ldots, D(t-1)$ and $\mathbf{S}(0) \cdots, \mathbf{S}(t)$. A policy is *stationary* if $D(t_1) = D(t_2)$ when $\mathbf{S}(t_1) = \mathbf{S}(t_2)$ for any t_1, t_2. A *randomized* policy specifies a probability distribution on the set of decisions, while a *deterministic* policy makes a decision with certainty. A policy in general belongs to one of the following sets [19]:

- Π^{HR}: a set of randomized history dependent policies;
- Π^{SR}: a set of randomized stationary policies;
- Π^{SD}: a set of deterministic stationary policies.

Because $\Pi^{\text{SD}} \subseteq \Pi^{\text{SR}} \subseteq \Pi^{\text{HR}}$, the complexity of identifying an optimal policy in the set Π^{SD} is the lowest. However, an optimal policy can be outside the set Π^{SD} (according to [19]), i.e., no optimal policy lies in the set Π^{SD}. Thus, we want to investigate if there is a Δ-optimal policy in the set Π^{SD} in the next section.

4.1 Characterization of the Δ-Optimality

To characterize the Δ-optimality, we start with introducing an infinite horizon *α-discounted cost*, where $0 < \alpha < 1$ is a discount factor. We subsequently connect the discounted cost case to the average cost case, because

structures of a Δ-optimal policy usually depend on its discounted cost case.

Given initial state $\mathbf{S}(0) = \mathbf{s}$, the *expected total α-discounted cost* under scheduling algorithm π (that can be history dependent) is

$$V_\alpha(\mathbf{s};\pi) = \limsup_{T\to\infty} E_\pi\left[\sum_{t=0}^{T} \alpha^t C(\mathbf{S}(t), D(t)) | \mathbf{S}(0) = \mathbf{s}\right].$$

Definition 3. *A policy π of the MDP Δ is Δ_α-optimal if it minimizes the expected total α-discounted cost $V_\alpha(\mathbf{s};\pi)$. In particular, we define*

$$V_\alpha(\mathbf{s}) = \min_\pi V_\alpha(\mathbf{s};\pi).$$

Moreover, by $h_\alpha(\mathbf{s}) = V_\alpha(\mathbf{s}) - V_\alpha(\mathbf{0})$ we define the *relative cost function*, which is the difference of the minimum discounted costs between state $\mathbf{s}$ and a reference state $\mathbf{0}$. We can arbitrarily choose the reference state, e.g., $\mathbf{0} = (1, 2, \cdots, N,\ 1, \ldots, 1)$ in this paper. We then introduce the *discounted cost optimality equation* of $V_\alpha(\mathbf{s})$ below.

Proposition 4. *The minimum expected total α-discounted cost $V_\alpha(\mathbf{s})$, for initial state $\mathbf{s}$, satisfies the following* discounted cost optimality equation*:*

$$V_\alpha(\mathbf{s}) = \min_{d\in\{0,1,\ldots,N\}} C(\mathbf{s}, d) + \alpha E[V_\alpha(\mathbf{s}')], \tag{3}$$

where the expectation is taken over all possible next state $\mathbf{s}'$ reachable from the state $\mathbf{s}$, i.e., $E[V_\alpha(\mathbf{s}')] = \sum_{\mathbf{s}'\in\mathbf{S}} P_{\mathbf{s},\mathbf{s}'}(d) V_\alpha(\mathbf{s}')$. A deterministic stationary policy that realizes the minimum of the right-hand-side (RHS) of the discounted cost optimality equation in Eq. (3) is a Δ_α-optimal policy. Moreover, we can define a value iteration $V_{\alpha,n}(\mathbf{s})$ by $V_{\alpha,0}(\mathbf{s}) = 0$ and for any $n \geq 0$,

$$V_{\alpha,n+1}(\mathbf{s}) = \min_{d\in\{0,1,\ldots,N\}} C(\mathbf{s}, d) + \alpha E[V_{\alpha,n}(\mathbf{s}')]. \tag{4}$$

Then, $V_{\alpha,n}(\mathbf{s}) \to V_\alpha(\mathbf{s})$ as $n \to \infty$, for every $\mathbf{s}$ and α.

Proof. Please see Appendix A, available in the online supplemental material. □

The value iteration in Eq. (4) is helpful for characterizing $V_\alpha(\mathbf{s})$, e.g., showing that $V_\alpha(\mathbf{s})$ is a non-decreasing function in the following.

Proposition 5. *$V_\alpha(x_i, \mathbf{x}_{-i}, \boldsymbol{\lambda})$ is a non-decreasing function in x_i, given $\mathbf{x}_{-i} = (x_1, \ldots, x_N) - \{x_i\}$ and $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_N)$.*

Proof. Please see Appendix B, available in the online supplemental material. □

Using Propositions 4 and 5 for the discounted cost case, we show that the MDP Δ has a deterministic stationary Δ-optimal policy, as follows.

Lemma 6. *There exists a deterministic stationary policy that is Δ-optimal. Moreover, there exists a finite constant $V^* = \lim_{\alpha\to 1}(1-\alpha)V_\alpha(\mathbf{s})$ for every state $\mathbf{s}$ such that the minimum average cost is V^*, independent of initial state $\mathbf{s}$.*

Proof. Please see Appendix C, available in the online supplemental material. □

We want to further elaborate on Lemma 6.

- First, in general, a result like Lemma 6 needs some conditions to ensure that the reduced Markov chain by a deterministic stationary policy is positive recurrent; however, Lemma 6 does not. Intuitively, we can think of the age of our problem as an age-queueing system, consisting of an age-queue, input to the queue, and a server. The input rate is one per slot since the age increases by one for each slot, while the server can serve an *infinite* number of age-packets for each service opportunity. As such, we always can find a scheduling algorithm such that the average arrival rate is less than the service rate and thus the reduced Markov chain is positive recurrent. Please see Appendix C, available in the online supplemental material for details.
- Second, since our MDP Δ involves the *long-run average cost optimization* with the *countably infinite* state space and the *unbounded* immediate cost, a Δ-optimal policy of such an MDP might not satisfy the average cost optimality equation like Eq. (3) (see [20] for a counter-example), even though the optimality of a deterministic stationary policy is established in Lemma 6.

In addition to the optimality of deterministic stationary policies, we show that a Δ-optimal policy has a nice structure. To investigate such structural results not only facilitates the scheduling algorithm design in Section 4, but also simplifies the calculation of the Whittle index in Section 5.

Definition 7. *A* switch-type *policy is a special deterministic stationary policy of the MDP Δ: given $\mathbf{x}_{-i}$ and $\boldsymbol{\lambda}$, if the action of the policy for state $\mathbf{s} = (x_i, \mathbf{x}_{-i}, \boldsymbol{\lambda})$ is $d_\mathbf{s} = i$, then the action for state $\mathbf{s}' = (x_i + 1, \mathbf{x}_{-i}, \boldsymbol{\lambda})$ is $d_{\mathbf{s}'} = i$ as well.*

In general, showing that a Δ-optimal policy satisfies a structure relies on an optimality equation; however, as discussed, the average cost optimality equation for the MDP Δ might not be available. To resolve this issue, we first investigate the discounted cost case by the well-established value iteration in Eq. (4), and then extend to the average cost case.

Theorem 8. *There exists a switch-type policy of the MDP Δ that is Δ-optimal.*

Proof. First, we start with the discounted cost case, and show that a Δ_α-optimal scheduling algorithm is switch-type. Let $\nu_\alpha(\mathbf{s}; d) = C(\mathbf{s}, d) + \alpha E[V_\alpha(\mathbf{s}')]$. Then, $V_\alpha(\mathbf{s}) = \min_{d\in\{0,1,\ldots,N\}} \nu_\alpha(\mathbf{s}; d)$. Without loss of generality, we suppose that a Δ_α-optimal action at state $\mathbf{s} = (x_1, \mathbf{x}_{-1}, \boldsymbol{\lambda})$ is to update the user u_1 with $\lambda_1 = 1$. Then, according to the optimality of $d^*_{(x_1,\mathbf{x}_{-1},\boldsymbol{\lambda})} = 1$,

$$\nu_\alpha(x_1, \mathbf{x}_{-1}, \boldsymbol{\lambda}; 1) - \nu_\alpha(x_1, \mathbf{x}_{-1}, \boldsymbol{\lambda}; j) \leq 0,$$

for all $j \neq 1$.

Let $\mathbf{1} = (1, \ldots, 1)$ be the vector with all entries being one. Let $\mathbf{x}_i = (0, \ldots, x_i, \ldots, 0)$ be the zero vector except for the ith entry being replaced by x_i. To demonstrate the

switch-type structure, we consider the following two cases:

1) *For any other user* u_j *with* $\lambda_j = 1$: Since $V_\alpha(x_1, \mathbf{x}_{-1}, \boldsymbol{\lambda})$ is a non-decreasing function in x_1 (see Proposition 5), we have

$$\begin{aligned}
&\nu_\alpha(x_1+1, \mathbf{x}_{-1}, \boldsymbol{\lambda}; 1) - \nu_\alpha(x_1+1, \mathbf{x}_{-1}, \boldsymbol{\lambda}; j) \\
&= x_j - (x_1+1) + \alpha E[V_\alpha(1, \mathbf{x}_{-1} + \mathbf{1}, \boldsymbol{\lambda}') \\
&\quad - V_\alpha(x_1+2, \mathbf{x}_{-1} + \mathbf{1} - \mathbf{x}_j, \boldsymbol{\lambda}')] \\
&\le x_j - x_1 + \alpha E[V_\alpha(1, \mathbf{x}_{-1} + \mathbf{1}, \boldsymbol{\lambda}') \\
&\quad - V_\alpha(x_1+1, \mathbf{x}_{-1} + \mathbf{1} - \mathbf{x}_j, \boldsymbol{\lambda}')] \\
&= \nu_\alpha(x_1, \mathbf{x}_{-1}, \boldsymbol{\lambda}; 1) - \nu_\alpha(x_1, \mathbf{x}_{-1}, \boldsymbol{\lambda}; j) \le 0,
\end{aligned}$$

where $\boldsymbol{\lambda}'$ is the next arrival vector.

2) *For any other user* u_j *with* $\lambda_j = 0$: Similarly, we have

$$\begin{aligned}
&\nu_\alpha(x_1+1, \mathbf{x}_{-1}, \boldsymbol{\lambda}; 1) - \nu_\alpha(x_1+1, \mathbf{x}_{-1}, \boldsymbol{\lambda}; j) \\
&= -(x_1+1) + \alpha E[V_\alpha(1, \mathbf{x}_{-1} + \mathbf{1}, \boldsymbol{\lambda}') \\
&\quad - V_\alpha(x_1+2, \mathbf{x}_{-1} + \mathbf{1}, \boldsymbol{\lambda}')] \le 0.
\end{aligned}$$

According to the two cases, a Δ_α-optimal action for state $(x_1+1, \mathbf{x}_{-1}, \boldsymbol{\lambda})$ is still to update u_1, yielding the switch-type structure.

Then, we proceed to establish the optimality for the average cost case. Let $\{\alpha_n\}_{n=1}^\infty$ be a sequence of the discount factors. According to [21], if the both conditions in Appendix C, available in the online supplemental material hold, then there exists a subsequence $\{\beta_n\}_{n=1}^\infty$ such that a Δ-optimal policy is the limit point of the Δ_{β_n}-optimal policies. By induction on β_n again, we obtain that a Δ-optimal is switch-type as well. □

4.2 Finite-State MDP Approximations

The classical method for solving an MDP is to apply a value iteration method [19]. However, as mentioned, the average cost optimality equation might not exist. Even though average cost value iteration holds like Eq. (4), the value iteration cannot work in practice, as we need to update an infinite number of states for each iteration. To address the issue, we propose a sequence of finite-state approximate MDPs. In general, a sequence of approximate MDPs might not converge to the original MDP according to [22]. Thus, we will rigorously show the convergence of the proposed sequence.

Let $X_i^{(m)}(t)$ be a *virtual age* of information for user u_i in slot t, with the dynamic being

$$X_i^{(m)}(t+1) = \begin{cases} 1 & \text{if } \Lambda_i(t) = 1,\ D(t) = i; \\ \left[X_i^{(m)}(t) + 1\right]_m^+ & \text{else,} \end{cases}$$

where we define the notation $[x]_m^+$ by $[x]_m^+ = x$ if $x \le m$ and $[x]_m^+ = m$ if $x > m$, i.e., we truncate the real age by m. This is different from Eq. (1). While the real age $X_i(t)$ can go beyond m, the virtual age $X_i^{(m)}(t)$ is at most m. Here, we reasonably choose the truncation m to be greater than the number N of users, i.e., $m > N$. Later, in Appendix D, available in the online supplemental material (see Remark 23), we will discuss some mathematical reasons for the choice.

By $\{\Delta^{(m)}\}_{m=N+1}^\infty$ we define a sequence of approximate MDPs for Δ, where each MDP $\Delta^{(m)}$ is the same as the original MDP Δ except:

- *States*: The state in slot t is $\mathbf{S}^{(m)}(t) = (X_1^{(m)}(t), \cdots, X_N^{(m)}(t), \Lambda_1(t), \ldots, \Lambda_N(t))$. Let $\mathbf{S}^{(m)}$ be the state space.
- *Transition probabilities*: Under action $D(t) = d$, the transition probability $P_{\mathbf{s},\mathbf{s}'}^{(m)}(d)$ of the MDP $\Delta^{(m)}$ from state $\mathbf{s} = (x_1, \ldots, x_N, \lambda_1, \ldots, \lambda_N)$ to state $\mathbf{s}' = (x_1', \ldots, x_N', \lambda_1', \ldots, \lambda_N')$ is

$$P_{\mathbf{s},\mathbf{s}'}(d) = \prod_{i:\lambda_i'=1} p_i \prod_{i:\lambda_i'=0} (1-p_i),$$

if $x_i' = [(x_i+1) - x_i \mathbf{1}_{i=d \text{ and } \lambda_i=1}]_m^+$ for all $i = 1, \ldots, N$.

Next, we show that the proposed sequence of approximate MDPs converges to the Δ-optimum.

Theorem 9. *Let* $V^{(m)*}$ *be the minimum average cost for the MDP* $\Delta^{(m)}$. *Then,* $V^{(m)*} \to V^*$ *as* $m \to \infty$.

Proof. Please see Appendix D, available in the online supplemental material. □

4.3 Structural MDP Scheduling Algorithm

Now, for a given boundary size m, we are ready to propose a practical algorithm to solve the MDP $\Delta^{(m)}$. The traditional *relative value iteration algorithm* (RVIA), as follows, can be applied to obtain an optimal deterministic stationary policy for $\Delta^{(m)}$:

$$V_{n+1}^{(m)}(\mathbf{s}) = \min_{d \in \{0,1,\ldots,N\}} C(\mathbf{s}, d) + E[V_n^{(m)}(\mathbf{s}')] - V_n^{(m)}(\mathbf{0}), \tag{5}$$

for all $\mathbf{s} \in \mathbf{S}^{(m)}$ where the initial value function is $V_0^{(m)}(\mathbf{s}) = 0$. For each iteration, we need to update actions for *all* virtual states by minimizing the RHS of Eq. (5) as well as update $V^{(m)}(\mathbf{s})$ for *all* $\mathbf{s} \in \mathbf{S}^{(m)}$. As the size of the state space is $O(m^N)$, the computational complexity of updating all virtual states in each iteration of Eq. (5) is more than $O(m^N)$. The complexity primarily results from the boundary size m of the MDP $\Delta^{(m)}$ and the number N of users.

In this section, we propose *structural RVIA* in Algorithm 1 for reducing the complexity of the traditional RVIA by leveraging the switch-type structure. In Algorithm 1, we seek an optimal action $d_{\mathbf{s}}^*$ for each virtual state $\mathbf{s} \in \mathbf{S}^{(m)}$ by iteration. For each iteration, we update both the optimal action $d_{\mathbf{s}}^*$ and $V^{(m)}(\mathbf{s})$ for all virtual states. If a switch structure[2] is identified in Line 4, we can *immediately* determine an optimal action in Line 5, instead of *searching for* an optimal action in Line 7. Then, by $V_{\text{tmp}}(\mathbf{s})$ in Line 9 we temporarily keep the updated value, which will replace $V^{(m)}(\mathbf{s})$ in Line 11.

The complexity of Algorithm 1 depends on the switch curve. Take two users for example. If an optimal action for state $(x_1, x_2, \boldsymbol{\lambda})$ is to update user u_2, then an optimal action for state $(x_1, x, \boldsymbol{\lambda})$ with $x > x_2$ is to update user u_2 without further computation. Suppose that Algorithm 1 searches for an optimal policy in the order of age vectors $(1,1), \ldots,$

2. An optimal policy for the truncated MDPs is the switch-type as well, according to the same proof as Theorem 8.

$(1,m),(2,1),\ldots,(2,m),\cdots$ where m is the boundary size, then the complexity is the number of states whose optimal actions are to update user u_1, achieving a lower computational complexity than the conventional RVIA (with complexity of m^2).

Algorithm 1. Structural RVIA

```
V^(m)(s) ← 0 for all virtual states s ∈ S^(m);
while 1 do
    forall s ∈ S^(m) do
        if there exists ζ > 0 and i ∈ {1,...,N} such that
        d*_(x_i−ζ,x_−i,λ) = i then
            d*_s ← i;
        else
            d*_s ← arg min_{d∈{0,...,N}} C(s,d) + E[V^(m)(s')];
        end
        V_tmp(s) ← C(s,d*_s) + E[V^(m)(s')] − V^(m)(0);
    end
    V^(m)(s) ← V_tmp(s) for all s ∈ S^(m).
end
```

Next, we establish the optimality of the structural RVIA for the approximate MDP $\Delta^{(m)}$.

Theorem 10. *For MDP $\Delta^{(m)}$ with a given m, the limit point of $d^*_{\mathbf{s}}$ in Algorithm 1 is a $\Delta^{(m)}$-optimal action for every virtual state $\mathbf{s} \in \mathbf{S}^{(m)}$. In particular, Algorithm 1 converges to the $\Delta^{(m)}$-optimum in a finite number of iterations.*

Proof. (Sketch) According to [19, Theorem 8.6.6], we only need to verify that the truncated MDP is *unichain*. Please see Appendix E, available in the online supplemental material for details. □

Based on the structural RVIA in Algorithm 1, we propose the *structural MDP scheduling algorithm*: Given the actions for all state $\mathbf{s} \in \mathbf{S}^{(m)}$ from Algorithm 1, for each slot t the scheduling algorithm makes a decision according to the *virtual age* $X_i^{(m)}(t)$ for all i, instead of the *real age* $X_i(t)$. Then, combining Theorems 9 and 10 yields that the proposed algorithm is asymptotically Δ-optimal as m approaches infinity.

5 An Index Scheduling Algorithm

By mean of the MDP techniques, we have developed the structural MDP scheduling algorithm. The scheduling algorithm not only reduces the complexity from the traditional RVIA, but also was shown to be asymptotically age-optimal. However, the scheduling algorithm might be infeasible for large number of users because it is based on the RVIA and still suffers from the curse of dimensionality. Thus, a low-complexity scheduling algorithm for much more users is still needed.

To fill this gap, we investigate the scheduling problem from the perspective of *restless bandits* [23]. In a (multi-armed) restless bandit problem, a gambler can select one bandit in each slot, for minimizing a long-run average cost. Note that each user in our problem can be viewed as a restless bandit. Thus, we can cast our scheduling problem into a restless bandit problem, where each user is a restless bandit and the average cost $V(\pi)$ in Eq. (2) is the long-run average cost for our restless bandit problem.

A restless bandit problem, in general, is PSPACE-hard [23]. Whittle thus proposed a relaxed version and came up with an index policy. The intuitions underlying the relaxation and the index are following.

Relaxed Problem. Recall that a Δ-optimal policy for minimizing the average cost $V(\pi)$ can be stationary (see Lemma 6). For a given randomized stationary policy π, the average cost $V(\pi)$ can be expressed as a linear combination (as an objective function) of the probability distribution of the policy π. Next, following Whittle's suggestion, we relax the constraint on the number of bandits for each slot to its *expected number*. The expectation can be expressed as another linear combination (as a constraint) of the probability distribution of the policy π. Then, applying the Lagrange function to combine the two linear combinations, we can observe (found by Whittle) that the original N bandits problem can be decoupled into N sub-problems. Each sub-problem consists of one bandit and a Lagrange multiplier c, with the purpose of minimizing its average age minus the Lagrange multiplier c. In fact, the multiplier can be viewed as the cost of selecting the bandit.

Whittle Index. Whittle made another assumption to each decoupled sub-problem, called the *indexability*. The indexability implies that for each restless bandit there exists a (Whittle) index $I(\mathbf{s})$ for each state $\mathbf{s}$ such that an optimal action for this state is to pull the bandit if $I(\mathbf{s}) > c$ and to idle if $I(\mathbf{s}) < c$. Both actions are indifferent when $I(\mathbf{s}) = c$ (this is exactly the definition of the Whittle index in Def. 16 later). With the assumption, we can view the the index $I(\mathbf{s})$ as a "value" of selecting the bandit. This motivated Whittle to suggest selecting a bandit with the highest index for achieving the highest total value.

The Whittle index policy is optimal for the relaxed problem; moreover, in many practical systems, the low-complexity index policy performs remarkably well, e.g., see [24]. With the success of the Whittle index policy to solve the restless bandit problem, we apply the Whittle's approach to develop a low-complexity scheduling algorithm. While Section 5.1 formulates an MPD for solving the decoupled sub-problem, Section 5.3 derives the Whittle index and builds the indexability. However, to obtain the Whittle index in closed form and to establish the indexability can be very challenging [23]. To address the issues, we simplify the derivation of the Whittle index by investigating structural results in Section 5.2, like Section. 4.

5.1 Decoupled Sub-Problem

Applying the Whittle's approach, we can decouple our problem into N sub-problems. Each sub-problem consists of a single user u_i and adheres to the network model in Section 2 with $N = 1$, except for an additional cost c for updating the user. In fact, the cost c is a scalar Lagrange multiplier in the Lagrangian approach. In each decoupled sub-problem, we aim to determine whether or not the BS updates the user in each slot, for striking a balance between the updating cost and the cost incurred by age. Since each sub-problem consists of a single user only, hereafter in this section we omit the index i for simplicity.

Similarly, we cast the sub-problem into another MDP Ω, which is the same as the MDP Δ in Section 4 with a single user except:

- *Actions*: Let $A(t) \in \{0, 1\}$ be an action of the MDP Ω in slot t indicating the BS's decision, where $A(t) = 1$ if the BS decides *to update* the user and $A(t) = 0$ if the BS decides *to idle*. Note that the action $A(t)$ is different from the scheduling decision $D(t)$. The action $A(t)$ is used for the decoupled sub-problem. In Section 5.4, we will use the MDP Ω to decide $D(t)$.
- *Costs*: Let $C(\mathbf{S}(t), A(t))$ be an immediate cost if action $A(t)$ is taken in slot t under state $\mathbf{S}(t)$, with the definition as follows.

$$C\Big(\mathbf{S}(t) = (x, \lambda), A(t) = a\Big) \triangleq (x + 1 - x \cdot a \cdot \lambda) + c \cdot a, \tag{6}$$

where the first part $x + 1 - x \cdot a \cdot \lambda$ is the resulting age in the next slot and the second part is the incurred cost for updating the user.

A *policy* $\mu = \{A(0), A(1), \cdots\}$ for the MDP Ω specifies an action $A(t)$ for each slot t. The *average cost* $J(\pi)$ under policy μ is defined by

$$J(\mu) = \limsup_{T \to \infty} \frac{1}{T+1} E_\mu \left[\sum_{t=0}^{T} C(\mathbf{S}(t), A(t)) \right].$$

Again, the objective of the MDP Ω is to find an Ω-optimal policy defined as follows.

Definition 11. *A policy μ for the MDP Ω is called* Ω-optimal *if it minimizes the average cost $J(\mu)$.*

Traditionally, the Whittle index might be obtained by solving the optimality equation of $J(\mu)$, e.g., [14], [23]. However, as discussed, the average cost optimality equation for the MDP Ω might not exit, and even if it exists, solving an optimality equation might be tedious. To look for a simple way for obtaining the Whittle index, we investigate structures of an Ω-optimal policy instead, by looking at its discounted case again. It turns out that our structural results will further simplify the derivation of the Whittle index.

5.2 Characterization of the Ω-Optimality

First, we show that an Ω-optimal policy is stationary deterministic as follows.

Lemma 12. *There exists a deterministic stationary policy that is Ω-optimal, independent of the initial state.*

Proof. Please see Appendix F, available in the online supplemental material. □

Next, we show that an Ω-optimal policy is a special type of deterministic stationary policies.

Definition 13. *A* threshold-type *policy is a special deterministic stationary policy of the MDP Ω. The action for state $(x, 0)$ is to idle, for all x. Moreover, if the action for state $(x, 1)$ is to update, then the action for state $(x + 1, 1)$ is to update as well. In other words, there exists a threshold $\bar{X} \in \{1, 2, \cdots\}$ such that the action is to update if there is an arrival and the age is greater than or equal to $\bar{X}$; otherwise, the action is to idle.*

Theorem 14. *If the update cost $c \geq 0$, then there exists a threshold-type policy that is Ω-optimal.*

Proof. Please see Appendix G, available in the online supplemental material. □

Thus far, we have successfully identify the threshold structure of an Ω-optimal policy. The MDP Ω then can be reduced to a discrete-time Markov chain (DTMC) by applying a threshold-type policy. To find an optimal threshold for minimizing the average cost, in the next lemma we explicitly derive the average cost for a threshold-type policy.

Lemma 15. *Given the threshold-type policy μ with the threshold $\bar{X} \in \{1, 2, \cdots\}$, then the average cost $J(\mu)$, denoted by $\mathscr{C}(\bar{X})$, under the policy is*

$$\mathscr{C}(\bar{X}) = \frac{\frac{\bar{X}^2}{2} + (\frac{1}{p} - \frac{1}{2})\bar{X} + \frac{1}{p^2} - \frac{1}{p} + c}{\bar{X} + \frac{1-p}{p}}. \tag{7}$$

Proof. Please see Appendix H, available in the online supplemental material. □

5.3 Derivation of the Whittle Index

Now, we are ready to define the Whittle index as follows.

Definition 16. *The Whittle index $I(\mathbf{s})$ for each state $\mathbf{s}$ is the cost c such that to update and to idle are both Ω-optimal actions for state $\mathbf{s}$.*

In the next theorem, we will obtain a very simple expression for the Whittle index by combining Theorem 14 and Lemma 15.

Theorem 17. *The Whittle index of the sub-problem for state (x, λ) is*

$$I(x, \lambda) = \begin{cases} 0 & \text{if } \lambda = 0; \\ \frac{x^2}{2} - \frac{x}{2} + \frac{x}{p} & \text{if } \lambda = 1. \end{cases} \tag{8}$$

Proof. It is obvious that the Whittle index for state $(x, 0)$ is $I(x, 0) = 0$ as both actions result in the same immediate cost and the same age of next slot if $c = 0$.

Let $g(x) = \mathscr{C}(x)$ in Eq. (7) for the domain of $\{x \in \mathbb{R} : x \geq 1\}$. Note that $g(x)$ is strictly convex in the domain. Let x^* be the minimizer of $g(x)$. Then, an optimal threshold for minimizing the average cost $\mathscr{C}(\bar{X})$ is either $\lfloor x^* \rfloor$ or $\lceil x^* \rceil$: the optimal threshold is $\bar{X}^* = \lfloor x^* \rfloor$ if $\mathscr{C}(\lfloor x^* \rfloor) < \mathscr{C}(\lceil x^* \rceil)$ and $\bar{X}^* = \lceil x^* \rceil$ if $\mathscr{C}(\lceil x^* \rceil) < \mathscr{C}(\lfloor x^* \rfloor)$. If there is a tie, both choices are optimal, i.e, equally desirable.

Hence, both actions for state $(x, 1)$ are equally desirable if and only if the age x satisfies

$$\mathscr{C}(x) = \mathscr{C}(x + 1), \tag{9}$$

i.e., $x = \lfloor x^* \rfloor$ and both thresholds of x and $x + 1$ are optimal. By solving Eq. (9), we can obtain the cost, as stated in the theorem, to make both actions equally desirable. □

According to Theorem 17, both actions might have a tie. If there is a tie, we break the tie in favor of idling. Then, we can explicitly express the optimal threshold in the next theorem.

Lemma 18. *The optimal threshold $\bar{X}$ for minimizing the average cost $\mathscr{C}(\bar{X})$ is x if the cost c satisfies $I(x - 1, 1) \leq c < I(x, 1)$, for all $x = 1, 2, \ldots$*

Proof. Please see Appendix I, available in the online supplemental material. □

Next, according to [25], we have to demonstrate the *indexability* such that the Whittle index is feasible.

Definition 19. *For a given cost c, let $\mathbf{S}(c)$ be the set of states such that the optimal actions for the states are to idle. The sub-problem is* indexable *if the set $\mathbf{S}(c)$ monotonically increases from the empty set to the entire state space, as c increases from $-\infty$ to ∞.*

Theorem 20. *The sub-problem is indexable.*

Proof. If $c < 0$, the optimal action for every state is to update; as such, $\mathbf{S}(0) = \emptyset$. If $c \geq 0$, then $\mathbf{S}(c)$ is composed of the set $\{\mathbf{s} = (x, 0) : x = 1, 2, \cdots\}$ and a set of $(x, 1)$ for some x's. According to Lemma 18, the optimal threshold monotonically increases to infinity as c increases, and hence the set $\mathbf{S}(c)$ monotonically increases to the entire state space. □

5.4 Index Scheduling Algorithm

Now, we are ready to propose a low-complexity *index scheduling algorithm* based on the Whittle index. For each slot t, the BS observes age $X_i(t)$ and arrival indicator $\Lambda_i(t)$ for every user u_i; then, updates user u_i with the highest value of the Whittle index $I(X_i(t), \Lambda_i(t))$, i.e., $D(t) = \arg\max_{i=1,\ldots,N} I(X_i(t), \Lambda_i(t))$.

The optimality of the index scheduling algorithm for the relaxed problem was known in [23]. Next, we show that the proposed index scheduling algorithm is exactly age-optimal for the original problem (without relaxation), when the packet arrivals for all users are *stochastically identical*.

Lemma 21. *If the arrival rates of all information sources are the same, i.e., $p_i = p_j$ for all $i \neq j$, then the index scheduling algorithm is age-optimal.*

Proof. Note that, in this case, the index scheduling algorithm sends an arriving packet with the largest age of information, i.e., $D(t) = \arg\max_i X_i(t)\Lambda_i(t)$ for each slot t. Then, in Appendix J, available in the online supplemental material we show that the policy is Δ-optimal. □

We want to remark that the result in [26, Proposition 2] showed that the index scheduling algorithm (called the Round-Robin (RR) policy in their paper) is age-optimal when both the arrival rates and the arrival indicators for all information sources are the identical, i.e., $p_i = p_j$ and $\Lambda_i(t) = \Lambda_j(t)$, for all $i \neq j$ and all t. Our Lemma 21 generalizes their result to *heterogeneous* states. In Section 7 we will further validate the index scheduling algorithm for stochastically non-identical arrivals by simulations. Moreover, we remark that, in the proof of Lemma 21 , we can see a connection between the two defined MDPs, i.e., using the MDP Δ to establish the result (Lemma 21) of the MDP Ω.

6 Online Scheduling Algorithm Design

Thus far, we have developed two scheduling algorithms in Sections 4 and 5. Both algorithms are offline, as the structural MDP scheduling algorithm and the index scheduling algorithm need the arrival statistics as prior information to pre-compute an optimal action for each virtual state and the Whittle index, respectively. To solve the more challenging case when the arrival statistics are unavailable, in this section we develop online versions for both offline algorithms.

6.1 An MDP-Based Online Scheduling Algorithm

We first develop an online version of the MDP scheduling algorithm by leveraging *stochastic approximation* techniques [27]. The intuition is that, instead of updating $V^{(m)}(\mathbf{s})$ for all virtual states in each iteration of Eq. (5), we update $V^{(m)}(\mathbf{s})$ by following a *sample path*, which is a set of outcomes of the arrivals over slots. It turns out that the sample-path updates will converge to the Δ-optimal solution. To that end, we need a *stochastic version* of the RVIA. However, the RVIA in Eq. (5) is not suitable because the expectation is inside the minimization (see [28] for details). While minimizing the RHS of Eq. (5) for a given current state, we would need the transition probabilities to calculate the expectation. To tackle this, we design *post-action states* for our problem, similar to the proof of Lemma 15 in Appendix H, available in the online supplemental material.

We define post-action state $\tilde{\mathbf{s}}$ as the ages and the arrivals *after* an action. The state we used before is referred to as the *pre-action* state. If $\mathbf{s} = (x_1, \ldots, x_N, \lambda_1, \ldots, \lambda_N) \in \mathbf{S}^{(m)}$ is a virtual state of the MDP $\Delta^{(m)}$, then the virtual post-action state after action d is $\tilde{\mathbf{s}} = (\tilde{x_1}, \ldots, \tilde{x_N}, \tilde{\lambda_1}, \ldots, \tilde{\lambda_N})$ with

$$\tilde{x_i} = \begin{cases} 1 & \text{if } i = d \text{ and } \lambda_i = 1; \\ [x_i + 1]_m^+ & \text{else,} \end{cases}$$

and $\tilde{\lambda_i} = \lambda_i$ for all i.

Let $\tilde{V}^{(m)}(\tilde{\mathbf{s}})$ be the value function based on the post-action states defined by

$$\tilde{V}^{(m)}(\tilde{\mathbf{s}}) = E[V^{(m)}(\mathbf{s})],$$

where the expectation is taken over all possible pre-action states $\mathbf{s}$ reachable from the post-action state. We can then write down the post-action average cost optimality equation [28] for the virtual post-action state $\tilde{\mathbf{s}} = (\tilde{x}_1, \ldots, \tilde{x}_N, \tilde{\lambda}_1, \ldots, \tilde{\lambda}_N)$:

$$\begin{aligned} &\tilde{V}^{(m)}(\tilde{\mathbf{s}}) + V^{(m)*} \\ =&E\Bigg[\min_{d\in\{0,1,\ldots,N\}} C\big((\tilde{\mathbf{x}}, \tilde{\boldsymbol{\lambda}}'), d\big) \\ &\quad + \tilde{V}^{(m)}([\tilde{\mathbf{x}} + \mathbf{1} - \tilde{\mathbf{x}}_d \tilde{\lambda}'_d]_m^+, \tilde{\boldsymbol{\lambda}}')\Bigg], \end{aligned}$$

where $\tilde{\boldsymbol{\lambda}}'$ is the next arrival vector; $\tilde{\mathbf{x}}_i = (0, \ldots, \tilde{x}_i, \ldots, 0)$ denotes the zero vector except for the ith entry being replaced by $\tilde{x}_i$; $\tilde{\boldsymbol{\lambda}}'_i = (0, \ldots, \tilde{\lambda}_i, \ldots, 0)$ denotes the zero vector except for the ith entry being replaced by $\tilde{\lambda}_i$; the vector $\mathbf{1} = (1, \ldots, 1)$ is the unit vector. From the above optimality equation, the RVIA is as follows:

$$\begin{aligned} \tilde{V}_{n+1}^{(m)}(\tilde{\mathbf{s}}) =&E\Bigg[\min_{d\in\{0,1,\ldots,N\}} C\big((\tilde{\mathbf{x}}, \tilde{\boldsymbol{\lambda}}'), d\big) \\ &\quad + \tilde{V}_n^{(m)}([\tilde{\mathbf{x}} + \mathbf{1} - \mathbf{x}_d \tilde{\lambda}'_d]_m^+, \tilde{\boldsymbol{\lambda}}')\Bigg] - \tilde{V}_n^{(m)}(\mathbf{0}). \end{aligned} \tag{10}$$

Subsequently, we propose the *MDP-based online scheduling algorithm* in Algorithm 2 based on the stochastic version of the

RVIA. In Lines 1-3, we initialize $\tilde{V}^{(m)}(\tilde{\mathbf{s}})$ of all virtual post-action states and start from the reference point. Moreover, by v we record $\tilde{V}^{(m)}(\tilde{\mathbf{s}})$ for the current virtual post-action state. By observing the current arrivals $\mathbf{\Lambda}(t)$ and plugging in Eq. (10), the expectation in Eq. (10) can be removed; as such, in Line 5 we optimally update a user by minimizing Eq. (11). Then, we update $\tilde{V}^{(m)}(\tilde{\mathbf{s}})$ of the current virtual post-action state in Line 7, where $\gamma(t)$ is a *stochastic step-size* in slot t to strike a balance between the previous $\tilde{V}^{(m)}(\tilde{\mathbf{s}})$ and the updated value v. Finally, the next virtual post-action state is updated in Lines 8 and 9

Algorithm 2. MDP-Based Online Scheduling Algorithm

/ Initialization */*
1 $\tilde{V}^{(m)}(\tilde{\mathbf{s}}) \leftarrow 0$ for all states $\tilde{\mathbf{s}} \in \mathbf{S}^{(m)}$;
2 $\tilde{\mathbf{s}} \leftarrow \mathbf{0}$;
3 $v \leftarrow 0$;
4 **while** 1 **do**
/ Decision in slot t */*
5 We optimally make a decision $D^*(t)$ in slot t according to the current arrivals $\mathbf{\Lambda}(t) = (\Lambda_1(t), \ldots, \Lambda_N(t)$ in slot t:

$$D^*(t) = \underset{d\in\{0,1,\ldots,N\}}{\arg\min}\ C((\tilde{\mathbf{x}}, \mathbf{\Lambda}(t)), d) + \tilde{V}^{(m)}([\tilde{\mathbf{x}} + \mathbf{1} - \tilde{\mathbf{x}}_d \mathbf{\Lambda}_d(t)]_m^+, \mathbf{\Lambda}(t)); \quad (11)$$

/ Value update */*
6 $v \leftarrow C((\tilde{\mathbf{x}}, \mathbf{\Lambda}(t)), D^*(t)) + \tilde{V}^{(m)}([\tilde{\mathbf{x}} + \mathbf{1} - \tilde{\mathbf{x}}_{D^*(t)} \mathbf{\Lambda}_{D^*(t)}(t)]_m^+, \mathbf{\Lambda}(t)) - \tilde{V}^{(m)}(\mathbf{0})$;
7 $\tilde{V}^{(m)}(\tilde{\mathbf{s}}) \leftarrow (1-\gamma(t))\tilde{V}^{(m)}(\tilde{\mathbf{s}}) + \gamma(t)v$;
/ post-action state update */*
8 $\tilde{\mathbf{x}} \leftarrow [\tilde{\mathbf{x}} + \mathbf{1} - \tilde{\mathbf{x}}_{D^*(t)} \mathbf{\Lambda}_{D^*(t)}(t)]_m^+$;
9 $\tilde{\boldsymbol{\lambda}} \leftarrow \mathbf{\Lambda}(t)$.
15 **end**

Next, we show the optimality of the MDP-based online scheduling algorithm as slot t approaches infinity.

Theorem 22. *If $\sum_{t=0}^{\infty} \gamma(t) = \infty$ and $\sum_{t=0}^{\infty} \gamma^2(t) < \infty$, then Algorithm 2 converges to $\Delta^{(m)}$-optimum.*

Proof. According to [29], [30], we only need to verify that the truncated MDP is unichain, which has been completed in Appendix E, available in the online supplemental material. □

In the above theorem, $\sum_{t=0}^{\infty} \gamma(t) = \infty$ implies that Algorithm 2 needs an infinite number of iterations to learn the Δ-optimal solution, while the offline Algorithm 1 converges to the optimal solution in a finite number of iterations. Moreover, $\sum_{t=0}^{\infty} \gamma^2(t) < \infty$ means that the *noise* from measuring $\tilde{V}^{(m)}(\tilde{\mathbf{s}})$ can be controlled. Finally, we want to emphasize that the proposed Algorithm 2 is asymptotically Δ-optimal, i.e., it converges to the Δ-optimal solution when both the truncation m and the slot t go to infinity. In Section 6, we will also numerically investigate the algorithm over finite slots.

6.2 An Index-Based Online Scheduling Algorithm

Next, we note that the simple Whittle index $I(x, \lambda)$ in Eq. (8) depends on its arrival probability only. Thus, if the arrival probability is unknown, for each slot t we revise the index by

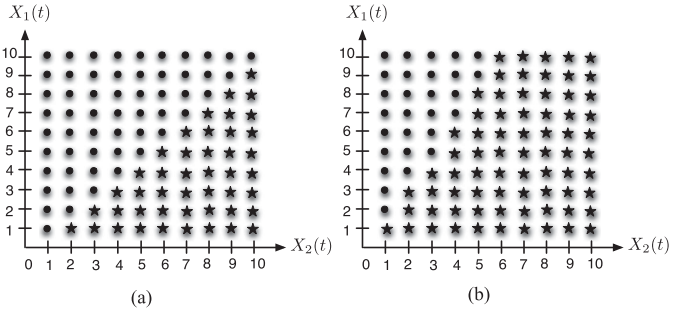

Fig. 3. Switch structure of Algorithm 1 for (a) $p_1 = p_2 = 0.9$; (b) $p_1 = 0.9$, $p_2 = 0.5$. The dots represent $D(t) = 1$ to update user u_1 and the stars mean $D(t) = 2$ to update user u_2.

$$I(x, \lambda, t) = \begin{cases} 0 & \text{if } \lambda = 0; \\ \frac{x^2}{2} - \frac{x}{2} + \frac{x}{p(t)} & \text{if } \lambda = 1, \end{cases}$$

where

$$p(t) = \frac{\sum_{\tau=0}^{t} \Lambda(\tau)}{t+1} = \frac{p(t-1)\cdot t + \Lambda(t)}{t+1}$$

is the running average arrival rate. Then, we propose the *index-based online scheduling algorithm* as follows. For each slot t, the BS observes age $X_i(t)$ and arrival indicator $\Lambda_i(t)$ for every user u_i; then, calculate $p_i(t)$ and update user u_i with the highest value of the revised Whittle index $I(X_i(t), \Lambda_i(t), t)$.

7 Simulation Results

In this section we conduct extensive computer simulations for the proposed four scheduling algorithms. We demonstrate the switch-type structure of Algorithm 1 in Section 7.1. In Section 7.2 we compare the proposed scheduling algorithms, especially to validate the performance of the online algorithms over finite slots. Finally, we study the wireless broadcast network with buffers at the BS in Section 7.3.

7.1 Switch-type Structure of Algorithm 1

Fig. 3a and 3b show the switch-type structure of Algorithm 1 for two users, when the BS has packets for both users. The experiment setting is as follows. We run Algorithm 1 with the boundary $m = 10$ over 100,000 slots to search for an optimal action for each virtual state. Moreover, we consider two arrival rate vectors, with (p_1, p_2) being $(0.9, 0.9)$ and $(0.9, 0.5)$ in Fig. 3a and 3b, respectively, where the *dots* represent $D(t) = 1$ and the *stars* mean $D(t) = 2$ when the BS has both arrivals in the same slot. We can observe the switch structure in the figures. For example in Fig. 3a, when age $X_2(t) = 3$ for user u_2 in some slot t, an optimal decision in slot t is to update user u_2 if age $X_1(t) \leq 2$ and to update u_1 if age $X_1(t) \geq 3$. Thus, Fig. 3a is consistent with the index scheduling algorithm in Section 5 by simply comparing the ages of the two users. Moreover, by fixing the arrival rate $p_1 = 0.9$ for the first user, the BS will give a higher priority to the second user as p_2 decreases in Fig. 3b. That is because u_2 spends more time to wait for the next arrival and becomes a bottleneck.

7.2 Numerical Studies of the Proposed Scheduling Algorithms

In this section, we examine the proposed four algorithms from various perspectives. First, we show the average ages of two

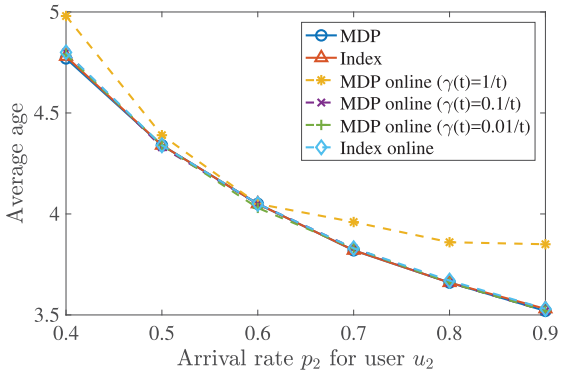

Fig. 4. Average ages for different arrival rates p_2, where we fix $N = 2$ and $p_1 = 0.6$.

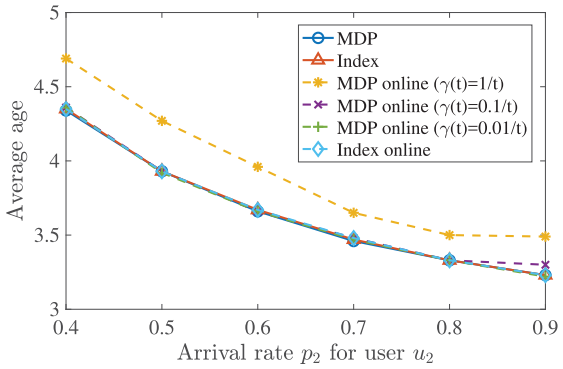

Fig. 5. Average ages for different arrival rates p_2, where we fix $N = 2$ and $p_1 = 0.8$.

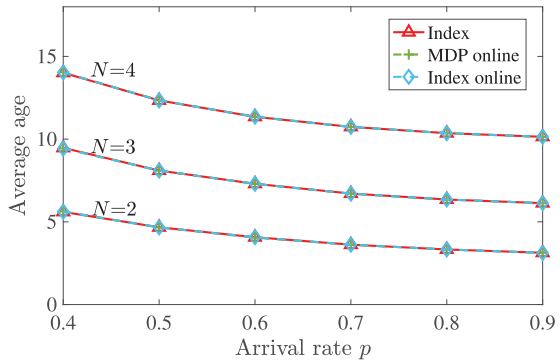

Fig. 6. Average ages for different arrival rates $p_1 = p_2 = p$, where we fix $N = 2, 3, 4$, respectively.

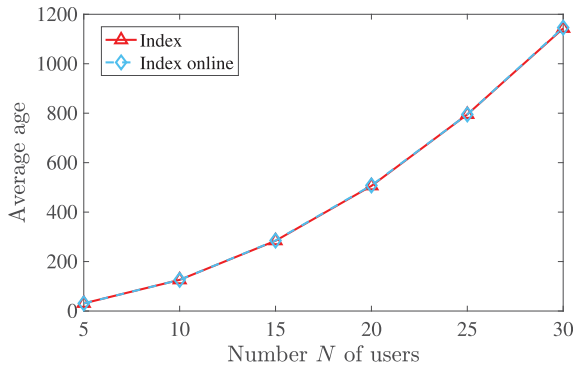

Fig. 7. Average ages for different different numbers N of users, where we fix the arrival rate $p_1 = p_2 = 1/N$.

users for different p_2 in Figs. 4 and 5 with fixed $p_1 = 0.6$ and $p_1 = 0.8$, respectively. Here, we set the boundary size $m = 30$ for the structural MDP scheduling algorithm. All the results are averaged over 100,000 slots. We note that the age of information for each slot (under the structural MDP scheduling algorithm) in our simulations never reached the boundary m and the average ages (see Figs. 4 and 5) are way below the boundary size. Thus, the boundary size would be large enough for the structural MDP scheduling algorithm to approach the minimum average age (as the the structural MDP scheduling algorithm is asymptotically age-optimal for a large boundary size, as shown in Theorem 9).

Compared with the structural MDP scheduling algorithm, the low-complexity index algorithm in Figs. 4 and 5 almost achieves the minimum average age. Without the knowledge of the arrival statistics, we set the boundary $m = 100$ for the MDP-based online scheduling algorithms; moreover, we consider different step sizes in both figures, i.e., $\gamma(t) = 1/t$, $\gamma(t) = 0.1/t$, and $\gamma(t) = 0.01/t$. From the simulations, we can observe in Figs. 4 and 5 that $\gamma(t) = 0.01/t$ performs the best among the three choices and almost achieves the minimum average age. How to choose the best step size with provably performance guarantee is interesting, but is out of scope of this paper. In addition, the index-based online scheduling algorithm also approaches the minimum average age in Figs. 4 and 5.

Second, because of the age-optimality of the index scheduling algorithm under homogeneous arrival rates (shown in Lemma 21), we show in Fig. 6 the average ages of more than two users (with $N = 2, 3, 4$) for different arrival rates $p_1 = p_2 = p$. From Fig. 6, we can find that both online scheduling algorithms are almost age-optimal again, where we use $\gamma(t) = 0.01/t$ only.

Third, because of the age-optimality of the index scheduling algorithm under homogeneous arrival rates again, we show in Fig. 7 the average ages for different numbers of users, where we fix $p_1 = p_2 = 1/N$. In this setting, the two MDP-based scheduling algorithms may be practically infeasible because the resulting huge state space; thus, we consider the two index-based scheduling algorithms only. We can find that the low-complexity index-based online scheduling algorithm achieves the minimum average age. In addition, Fig. 7 shows that the minimum average age increases super-linearly with the number of users.

By these numerical studies, we would suggest implementing the index-based online scheduling algorithm. It is not only simple to implement practically, but also has good performance.

7.3 Networks with Buffers

Thus far, we consider the no-buffer network only. Finally, we study buffers at the BS to store the latest information for each user. Similar to Section 4 we can find an age-optimal scheduling by an MDP. However, we need to redefine the states for the MDP Δ. In addition to the age $X_i(t)$ of information at *user* u_i, by $Y_i(t)$ we define the *initial age of the information* at the *buffer* for user u_i; precisely,

$$Y_i(t) = \begin{cases} 0 & \text{if } \Lambda(t) = 1; \\ Y_i(t-1) + 1 & \text{else.} \end{cases}$$

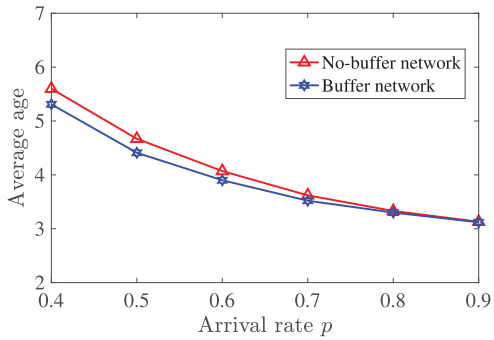

Fig. 8. Minimum average ages for the network with/without the buffers by running the MDP-based scheduling algorithms, where the arrival rates $p_1 = p_2 = p$.

Then, we redefine the state by $\mathbf{S}(t) = \{X_1(t), \ldots, X_N(t), Y_1(t), \ldots, Y_N(t)\}$. Moreover, the immediate cost is redefined as

$$C(\mathbf{S}(t), D(t) = d) = \sum_{i=1}^{N} (X_i(t) + 1) - (X_d(t) - Y_d(t)),$$

where we define $Y_0(t) = 0$ for all t. Then, similar to Section 4, we can show that

- there exists a deterministic stationary policy that is age-optimal;
- the similar sequence of approximate MDPs converges;
- an age-optimal scheduling algorithm is switch-type: for every user u_i, if a Δ-optimal action at state $\mathbf{s} = (x_i, \mathbf{x}_{-i}, \mathbf{y})$ is $d^*_{(x_i, \mathbf{x}_{-i}, \mathbf{y})} = i$, then $d^*_{(x_i+1, \mathbf{x}_{-i}, \mathbf{y})} = i$, where $\mathbf{y} = (y_1, \ldots, y_N)$ is the vector of all initial ages.

We then modify Algorithm 1 for the network with the buffers, as an age-optimal scheduling algorithm.

To study the effect of the buffers, we consider the truncated MDP with the boundary $m = 30$ and generate arrivals with $p_1 = p_2 = p$. After averaging the age over 100,000 slots, we can obtain the average ages in Fig. 8 for various p, where the red curve with the triangle markers indicates the no-buffer networks (by employing Algorithm 1) and the blue curve with the star markers indicates the network with the buffers.

In this setting, we see mild improvement of the average age from exploiting the buffers. The buffers reduce the average age by only around $(5.6 - 5.3)/5.6 \approx 5\%$ when $p = 0.4$, and even lower when p is higher. Let us discuss the following three cases when *both users have arrivals in some slot*:

- *When both p_1 and p_2 are high*: That means the user who is not updated currently has a new arrival in the next slot with a high probability; as such, the old packet in the buffer seems not that effective.
- *When both p_1 and p_2 are low*: Then, the possibility of the two arrivals in the same slot is very low. Hence, this would be a trivial case.
- *When one of p_1 and p_2 are high and the other is low*: In this case, the BS will give the user with the lower arrival rate a higher update priority, as a packet for the other user will arrive shortly.

According to the above discussions, we can observe that the buffers might not be that effective as expected. The no-buffer network is not only simple for practical implementation but also performs well.

8 Concluding Remarks

In this paper, we treated a wireless broadcast network, where many users are interested in different information that should be delivered by a base-station. We studied the age of information by designing and analyzing four scheduling algorithms, i.e., the structural MDP scheduling algorithm, the index scheduling algorithm, the MPD-based online scheduling algorithm, and the index-based online scheduling algorithm. We not only theoretically investigated the optimality of the proposed algorithms, but also validate their performance via the computer simulations. It turns out that the low-complexity index scheduling algorithm and both online scheduling algorithms almost achieve the minimum average age.

Some possible future works are discussed as follows. This paper focused on the no-buffer network. It is an issue to study provable effectiveness of the buffers and to characterize the regime under which the no-buffer network performs with marginal performance loss. It is also interesting to investigate structural results like ours for simplifying the calculation of the Whittle index for networks with buffers. In addition, this paper focused on the noiseless channel. For the case when a packet is likely to get lost in a slot, we can consider an equivalent model, where $\Lambda_i(t) = 1$ if a packet from source s_i arrives at the BS in slot t and also the channel to user u_i is ON in slot t. With the revised model, the proposed algorithms can be extended if the ON/OFF channel state for each slot can be estimated before decision. However, without perfect channel estimation, no re-transmission mechanism is considered in this work. This would be an another interesting future work. Finally, the paper treated the age of information only. It is interesting to explore networks concerning both throughput and age.

Acknowledgments

The work of Yu-Pin Hsu was supported by the Ministry of Science and Technology of Taiwan under Grant MOST 107-2221-E-305-007-MY3. The work of Eytan Modiano was supported by NSF under Grants AST-1547331, CNS-1713725, and CNS-1701964, and by Army Research Office (ARO) under Grant W911NF-17-1-0508. The work of Lingjie Duan was supported by the Singapore Ministry of Education Academic Research Fund Tier 2 under Grant MOE2016-T2-1-173. This paper was presented in part in the Proceedings of IEEE ISIT in 2017 [1] and 2018 [2].

References

[1] Y.-P. Hsu, E. Modiano, and L. Duan, "Age of information: Design and analysis of optimal scheduling algorithms," in *Proc. IEEE Int. Symp. Inf. Theory*, 2017, pp. 561–565.
[2] Y.-P. Hsu, "Age of information: Whittle index for scheduling stochastic arrivals," in *Proc. IEEE Int. Symp. Inf. Theory*, 2018, pp. 2634–2638.
[3] S. Kaul, R. D. Yates, and M. Gruteser, "Real-time status: How often should one update?" *Proc IEEE INFOCOM*, 2012, pp. 2731–2735.
[4] D. P. Bertsekas, *Dynamic Programming and Optimal Control Vol. I and II*. Belmont, MA, USA: Athena Scientific, 2012.
[5] J. Cho and H. Garcia-Molina, "Synchronizing a database to improve freshness," *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2000, pp. 117–128, 2000.

[6] C. Kam, S. Kompella, G. D. Nguyen, and A. Ephremides, "Effect of message transmission path diversity on status age," *IEEE Trans. Inf. Theory*, vol. 62, no. 3, pp. 1360–1374, Mar. 2016.
[7] Y. Sun, E. Uysal-Biyikoglu, R. D. Yates, C. E. Koksal, and N. B. Shroff, "Update or wait: How to keep your data fresh," *Proc. IEEE INFOCOM*, 2016, pp. 1–9.
[8] R. D. Yates, "Lazy is timely: Status updates by an energy harvesting source," *Proc. IEEE Int. Symp. Inf. Theory*, 2015, pp. 3008–3012.
[9] B. T. Bacinoglu, E. T. Ceran, and E. Uysal-Biyikoglu, "Age of information under energy replenishment constraints," *Proc. Inf. Theory Appl. Workshop*, 2015, pp. 25–31.
[10] M. Costa, M. Codreanu, and A. Ephremides, "On the age of information in status update systems with packet management," *IEEE Trans. Inf. Theory*, vol. 62, no. 4, pp. 1897–1910, Apr. 2016.
[11] Q. He, D. Yuan, and A. Ephremides, "Optimal link scheduling for age minimization in wireless systems," *IEEE Trans. Inf. Theory*, vol. 64, no. 7, pp. 5381–5394, Jul. 2017.
[12] C. Joo and A. Eryilmaz, "Wireless scheduling for information freshness and synchrony: Drift-based design and heavy-traffic analysis," in *Proc. 15th Int. Symp. Model. Optimization Mobile Ad Hoc Wireless Netw.*, 2017, pp. 1–8.
[13] Y. Sun, E. Uysal-Biyikoglu, and S. Kompella, "Age-optimal updates of multiple information flows," *Proc. IEEE Conf. Comput. Commun. Workshops*, 2018, pp. 136–141.
[14] I. Kadota, A. Sinha, and E. Modiano, "Optimizing age of information in wireless networks with throughput constraints," *Proc. IEEE INFOCOM*, 2018, pp. 1844–1852.
[15] R. D. Yates, P. Ciblat, A. Yener, and M. Wigger, "Age-optimal constrained cache updating," in *Proc. IEEE Int. Symp. Inf. Theory*, 2017, pp. 141–145.
[16] Z. Jiang, B. Krishnamachari, S. Zhou, and Z. Niu, "Can decentralized status update achieve universally near-optimal age-of-information in wireless multiaccess channels?" *Proc 30th Int. Teletraffic Congr.*, 2018, vol. 1, pp. 144–152.
[17] Z. Jiang, S. Zhou, Z. Niu, and Y. Cheng, "A unified sampling and scheduling approach for status update in multiaccess wireless networks," *Proc. of IEEE INFOCOM*, 2019, pp. 208–216.
[18] R. Srikant and L. Ying, *Communication Networks: An Optimization, Control, and Stochastic Networks Perspective*. Cambridge, U.K.: Cambridge Univ. Press, 2013.
[19] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Cambridge, MA, USA: MIT Press, 1994.
[20] R. Cavazos-Cadena, "A counterexample on the optimality equation in Markov decision chains with the average cost criterion," *Syst. Control Lett.*, vol. 16, no. 5, pp. 387–392, 1991.
[21] L. I. Sennott, "Average cost optimal stationary policies in infinite state markov decision processes with unbounded costs," *Oper. Res.*, vol. 37, pp. 626–633, 1989.
[22] L. I. Sennott, *Stochastic Dynamic Programming and the Control of Queueing Systems*. Hoboken, NJ, USA: Wiley, 1998.
[23] J. Gittins, K. Glazebrook, and R. Weber, *Multi-Armed Bandit Allocation Indices*. Hoboken, NJ, USA: Wiley, 2011.
[24] M. Larranaga, U. Ayesta, and I. M. Verloop, "Stochastic and fluid index policies for resource allocation problems," *Proc. IEEE INFOCOM*, 2015, pp. 1230–1238, 2015.
[25] P. Whittle, "Restless bandits: Activity allocation in a changing world," *J. Appl. Probability*, vol. 25, pp. 287–298, 1988.
[26] R. Li, A. Eryilmaz, and B. Li, "Throughput-optimal wireless scheduling with regulated inter-service times," *Proc. IEEE INFOCOM*, 2013, pp. 2616–2624.
[27] V. S. Borkar, *Stochastic Approximation: A Dynamical Systems Viewpoint*. Cambridge, U.K.: Cambridge Univ. Press, 2008.
[28] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Hoboken, NJ, USA: Wiley, 2011.
[29] V. S. Borkar and S. P. Meyn, "The ODE method for convergence of stochastic approximation and reinforcement learning," *SIAM J. Control Optimization*, vol. 38, no. 2, pp. 447–469, 2000.
[30] J. Abounadi, D. Bertsekas, and V. Borkar, "Learning algorithms for Markov decision processes with average cost," *SIAM J. Control Optimization*, vol. 40, no. 3, pp. 681–698, 2001.
[31] L. Georgiadis, M. J. Neely, and L. Tassiulas, *Resource Allocation and Cross-Layer Control in Wireless Networks*. Delft, The Netherlands: Now Publishers Inc, 2006.
[32] L. I. Sennott, "On computing average cost optimal policies with application to routing to parallel queues," *Math. Methods Oper. Res.*, vol. 45, pp. 45–62, 1997.
[33] R. G. Gallager, *Discrete Stochastic Processes*, vol. 321. New York, NY, USA: Springer Science & Business Media, 2012.
[34] N. Salodkar, A. Bhorkar, A. Karandikar, and V. S. Borkar, "An online learning algorithm for energy efficient delay constrained scheduling over a fading channel," *IEEE J. Select. Areas Commun.*, vol. 26, no. 4, pp. 732–742, May 2008.

Yu-Pin Hsu received the BS and MS degrees from National Chiao Tung University, Taiwan, in 2005 and 2007, respectively, and the PhD degree from Texas A&M University, in 2014. He was a post-doctoral fellow with the Singapore University of Technology and Design, Singapore, in 2014, and with the Massachusetts Institute of Technology, in 2015. In 2016, he joined National Taipei University, Taiwan, where he is currently an assistant professor. His research interests center around communication networks with focus on algorithmic and control-theoretic aspects.

Eytan Modiano received the BS degree in electrical engineering and computer science from the University of Connecticut at Storrs, in 1986 and the MS and PhD degrees, both in electrical engineering from the University of Maryland, College Park, Maryland, in 1989 and 1992, respectively. He was a Naval Research Laboratory Fellow between 1987 and 1992 and a National Research Council Post Doctoral Fellow during 1992-1993. Between 1993 and 1999, he was with MIT Lincoln Laboratory. Since 1999, he has been on the faculty with MIT, where he is a professor and an associate department head with the Department of Aeronautics and Astronautics, and an associate director of the Laboratory for Information and Decision Systems (LIDS). His research is on communication networks and protocols with emphasis on satellite, wireless, and optical networks. He is the co-recipient of the MobiHoc 2016 Best Paper Award, the Wiopt 2013 Best Paper Award, and the Sigmetrics 2006 Best Paper Award. He is an editor-in-chief for the *IEEE/ACM Transactions on Networking*, and served as an associate editor for the *IEEE Transactions on Information Theory* and the *IEEE/ACM Transactions on Networking*. He was the technical program co-chair for IEEE Wiopt 2006, IEEE Infocom 2007, ACM MobiHoc 2007, and DRCN 2015. He is a fellow of the IEEE and an associate fellow of the AIAA, and served on the IEEE fellows committee.

Lingjie Duan (S'09-M'12-SM'16) received the PhD degree from The Chinese University of Hong Kong, in 2012. He is currently an assistant professor with the Engineering Systems and Design Pillar, Singapore University of Technology and Design, Singapore. His current research interests include network economics and game theory, cognitive and cooperative communications, energy harvesting wireless communications, mobile crowdsourcing, and wireless information surveillance. He was a recipient of the 2016 SUTD Excellence in Research Award, and the 10th IEEE ComSoc Asia-Pacific Outstanding Young Researcher Award in 2015. He was also the finalist of the Hong Kong Young Scientist Award 2014 under Engineering Science track. He is now an editor of the *IEEE Transactions on Wireless Communications* and the *IEEE Communications Surveys and Tutorials*. In 2016, he was a guest editor of the *IEEE Journal on Selected Areas in Communications* Special Issue on Human-in-the-loop Mobile Networks, and was also a guest editor of the *IEEE Wireless Communications Magazine* for feature topic Sustainable Green Networking and Computing in 5G Systems. He is also a regular TPC member of a number of leading conferences on wireless communications and networking (e.g., IEEE INFOCOM, SECON, WiOPT, GLOBECOM, ICC, and ACM MobiHoc).