1 Reinforcement Learning for Minimizing Age of Information over Wireless Links

Elif Tuğçe Ceran[a] , Deniz Gündüz[b] , and András György[c]

1.1 Introduction

In this chapter, we study the age of information (AoI) when status updates of an underlying process of interest, sampled and recorded by a *source* node, must be transmitted to one or more *destination* nodes over error prone wireless channels. We consider the practical setting, in which the statistics of the system are not known *a priori*, and must be learned in an online fashion. This requires designing reinforcement learning (RL) algorithms that can adapt their policy dynamically through interactions with the environment. Accordingly, the aim of this chapter is to design and analyze RL algorithms to minimize the average AoI at the destination nodes taking into account *retransmissions* due to channel errors.

Retransmissions are essential for providing reliability of status updates over error-prone channels, particularly in wireless settings, and are incorporated into almost all wireless communication standards. In the standard automatic repeat request (ARQ) protocol, failed transmissions are repeated until they are successfully received, or a maximum retransmission count is reached. Some of the recent standards including ZigBee (Alliance 2008), Bluetooth *IEEE 802.15.1*, WiFi *IEEE 802.11ac* and UWB (Ultra-wideband) *IEEE 802.15.4a* (Oppermann, Hamalainen & Iinatti 2004) use cyclic redundancy check (CRC) together with ARQ. On the other hand, in the hybrid ARQ (HARQ) protocol, the receiver combines information from previous transmission attempts of the same packet in order to increase the success probability of decoding. Recent communication standards including *IEEE 802.16m*, 3GPP LTE, LTE-A (E-UTRA 2013), *IEEE 802.11be*, and Narrow-Band IoT (NB-IoT) have adopted HARQ techniques to enhance the system performance, typically through a combination of CRC and forward error correction (FEC) (802.16e 2005 2006). In this chapter, we study both ARQ and HARQ protocols for the minimization of AoI.

Until recently, prior literature in the AoI framework assumed that the perfect statistical information regarding the random processes governing the status-update system is available to the source. However, an increasing number of works are focusing on the practically relevant problem (e.g. sensors embedded in unknown or time-varying environments) and study RL for AoI optimization (Hsu, Modiano & Duan

[a] Department of Electrical and Electronic Engineering, Middle East Technical University, Ankara, Turkey
[b] Department of Electrical and Electronic Engineering, Imperial College London, UK
[c] DeepMind, London, UK

2017, Ceran, Gündüz & György 2018, Ceran, Gündüz & György 2018, Ceran, Gündüz & György 2019, Sert, Sönmez, Baghaee & Uysal-Biyikoglu 2018, Ceran, Gündüz & György 2019, Beytur & Uysal 2019, Leng & Yener 2019, Abd-Elmagid, Ferdowsi, Dhillon & Saad 2019, Elgabli, Khan, Krouka & Bennis 2019, Abd-Elmagid, Dhillon & Pappas 2020, Hatami, Jahandideh, Leinonen & Codreanu 2020). Sert et al. (2018) considers an end-to-end IoT application running over the Internet without prior assumptions about the network topology and apply a deep RL algorithm. An RL approach to minimize the AoI in an ultra-reliable low-latency communication system is considered by Elgabli et al. (2019). (Hsu et al. 2017, Beytur & Uysal 2019) investigate the scheduling decisions with multiple receivers over a perfect channel, where the goal is to learn data arrival statistics. Q-learning (Sutton & Barto 1998) is used for a generate-at-will model by Hsu et al. (2017), while policy gradients and DQN methods are used for a queue-based multi-flow AoI-optimal scheduling problem by Beytur & Uysal (2019). In (Leng & Yener 2019), policy gradients and DQN methods are employed for AoI minimization in a wireless ad-hoc network, where nodes exchange status updates with one another over a shared spectrum. Average-cost RL algorithms are proposed by Ceran, Gündüz & György (2018),Ceran, Gündüz & György (2019) and Ceran, Gündüz & György (2018) to learn the decoding error probabilities in a status-update system with HARQ. The work of Ceran, Gündüz & György (2019) exploits RL methods in order to learn both decoding error probabilities and energy harvesting characteristics.

The rest of the chapter is organized as follows. Section 1.2 provides a brief background on Markov decision processes (MDPs) and RL methods, which will be used to model and solve the AoI minimization problems addressed in this chapter. Section 1.3 investigates a point-to-point status-update system with HARQ under a resource constraint and exploits an average-cost RL algorithm to minimize the average AoI. Section 1.4 extends the results in Section 1.3 to a multi-user status-update system, and presents various RL algorithms with different complexity-performance trade-offs. Section 1.5 considers an energy harvesting status-update system with HARQ and considers sensing cost at the source node, as well as the transmission cost of the updates. Finally, Section 1.6 concludes the chapter.

1.2 Preliminaries

Reinforcement learning is an important area of machine learning where a learning agent learns how to behave in an environment by performing actions and observing the results of its actions in the form of state transitions and costs in order to learn to minimize some notion of cumulative cost (Sutton & Barto 1998). In recent years, RL methods have attracted significant attention thanks to groundbreaking achievements in this area of research. Examples include AlphaGo, which incorporates deep RL, beating the world champions at the game of Go (Silver et al. 2016) as well as the Deep Q-Network (DQN) algorithm (Mnih, Kavukcuoglu, Silver, Rusu, Veness, Bellemare, Graves, Riedmiller, Fidjeland, Ostrovski, Petersen, Beattie, Sadik, Antonoglou, King, Kumaran, Wierstra, Legg & Hassabis 2015) beating humans playing numerous Atari

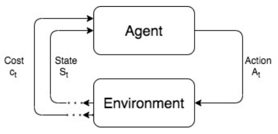

Figure 1.1 Illustrations of the interactions between the agent and the environment in the RL framework.

video games. RL methods have also been widely adopted for many wireless networking and mobile communication systems and applications (Clancy, Hecker, Stuntebeck & O'Shea 2007, Somuyiwa, György & Gündüz 2018, Luong, Hoang, Gong, Niyato, Wang, Liang & Kim 2019).

In the RL framework, as depicted in Figure 1.1, an agent repeatedly interacts with its environment: At time t the state of the environment is S_t. The agent takes an action A_t, which makes the environment to transition to another state S_{t+1}, and the agent suffers a cost c_t. The agent's goal is to minimize its long term costs.

This process can be conveniently modeled as a *Markov decision process (MDP)* (Puterman 1994): An MDP is defined with a tuple $\langle \mathcal{S}, \mathcal{A}, \mathrm{P}, c \rangle$, where $\mathcal{S}$ denotes a countable set of states and $\mathcal{A}$ denotes a countable set of actions.[1] The transition kernel $\mathrm{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ defines the transition probabilities: that is, if action $a \in \mathcal{A}$ is taken in state $s \in \mathcal{S}$, the environment transitions to state $s' \in \mathcal{S}$ with probability $\mathrm{P}(s'|s,a)$, independently of previous states and actions (note that $\mathrm{P}(\cdot|s,a)$ defines a distribution over $\mathcal{S}$ and hence $\sum_{s' \in \mathcal{S}} \mathrm{P}(s'|s,a) = 1$ for all $s \in \mathcal{S}, a \in \mathcal{A}$). Thus, if S_t and A_t denote the state and action at time t, then $\mathrm{P}(s'|s,a) = \Pr(S_{t+1} = s'|S_t = s, A_t = a)$, and for any $s_0, \ldots, s_{t+1} \in \mathcal{S}$ and $a_0, \ldots, a_t \in \mathcal{A}$, the state action sequence $S_0, A_0, S_1, A_1, \ldots, S_t, A_t, S_{t+1}$ satisfies $\Pr(S_{t+1} = s_{t+1}|S_t = s_t, A_t = a_t) = \Pr(S_{t+1} = s_{t+1}|S_t = s_t, A_t = a_t, \ldots, S_0 = s_0, A_0 = a_0)$. Finally, the cost suffered by the agent is determined by the state of the environment and the action taken in that state via the cost function $c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$.

In the MDP formulation it is assumed that in every time step the agent observes the state of the MDP, and it can select its action based on its past observations. Therefore, an agent's strategy can be described by a *policy*, defined as a sequence of decision rules $\pi_t : (\mathcal{S} \times \mathcal{A})^t \to [0,1]$, which maps the past states and actions and the current state to a distribution over the actions. That is, after the state-action sequence $s_0, a_0, \ldots, s_{t-1}, a_{t-1}, s_t$, action a_t is selected (in state s_t) with probability $\pi_t(a_t|s_t, a_{t-1}, s_{t-1} \ldots, a_0, s_0)$. We use s_t^π and a_t^π to denote the sequences of states and actions, respectively, induced by policy $\pi = \{\pi_t\}$. A policy $\pi = \{\pi_t\}$ is called *stationary* if the distribution of the next action is independent of the past states and actions given the current state, and it is time invariant; that is, with a slight abuse of notation, $\pi_t(a_t|s_t, a_{t-1}, s_{t-1} \ldots, s_0, a_0) = \pi(a_t|s_t)$ for all t and $(s_i, a_i) \in \mathcal{S} \times \mathcal{A}$, $i = 1, \ldots, t$. Finally, a policy is said to be *deterministic* if it chooses an action with probability one; with a slight abuse of notation, we use $\pi(s)$ to denote the action taken with probability one in state s by a stationary deterministic policy.

The goal of the agent is to select a policy that minimizes its expected average cost

[1] Assuming that $\mathcal{S}$ and $\mathcal{A}$ are countable is not necessary, but simplifies the treatment of MDPs and is sufficient for our applications concerning the age of information (what is more, we also assume in the rest of the chapter that the action set is finite).

suffered after starting from state $s_0 \in \mathcal{S}$:

$$J^\pi(s_0) \triangleq \limsup_{T\to\infty} \frac{1}{T+1}\, \mathbb{E}\left[\sum_{t=0}^{T} c(s_t^\pi, a_t^\pi)\Big| s_0\right] .$$

A policy π^* achieving the minimum is called optimal. Under general conditions, there exists an optimal policy which is stationary, deterministic, and is independent of the start state s_0 (Puterman 1994).

Oftentimes, in practical problems, the agent has constraints on the actions it can take. For example, in an energy harvesting system it is not possible to make a transmission if the transmitter's battery does not contain enough energy (Gündüz, Stamatiou, Michelusi & Zorzi 2014). While such information can be included in the state, it is often simpler to keep the original state space and introduce some extra constraints governing the behavior of the agent. This can be modeled using a *constrained Markov decision process* (CMDP) (Altman 1999), which is an extension of an MDP: A CMDP is defined by the 5-tuple $\langle \mathcal{S}, \mathcal{A}, \mathrm{P}, c, d\rangle$, where $\mathcal{S}, \mathcal{A}, \mathrm{P}$ and c are defined as before, but an additional cost function $d : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, is introduced to describe the constraints to the system (in the update systems we consider, this can be the energy cost of a transmission).

Letting $C^\pi(s_0)$ denote the infinite horizon average cost for the constraint, starting from state $s_0 \in \mathcal{S}$, the goal of the agent in a CMDP is to minimize its average cost J^π subject to a constraint C_{max} on C^π; that is, to find and use a policy π solving the optimization problem

$$\begin{aligned}\text{Minimize } J^\pi(s_0) &\triangleq \limsup_{T\to\infty} \frac{1}{T+1}\, \mathbb{E}\left[\sum_{t=0}^{T} c(s_t^\pi, a_t^\pi)\Big| s_0\right],\\ \text{subject to } C^\pi(s_0) &\triangleq \limsup_{T\to\infty} \frac{1}{T+1}\, \mathbb{E}\left[\sum_{t=0}^{T} d(s_t^\pi, a_t^\pi)\Big| s_0\right] \le C_{max}.\end{aligned}$$

An optimal policy in an CMDP is a solution of the above problem. Under general conditions, an optimal policy is stationary and deterministic except for a single state (Altman 1999, Sennott 1993).[2]

The *differential value function* $h^\pi : \mathcal{S} \to \mathbb{R}$ and the *action-value function* $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ of a policy π are defined as

$$\begin{aligned}h^\pi(s_0) &= \limsup_{T\to\infty} \mathbb{E}\left[\sum_{t=0}^{T} c(s_t^\pi, a_t^\pi) - J^\pi(s_t^\pi)\Big| s_0\right] ;\\ Q^\pi(s_0, a_0) &= \limsup_{T\to\infty} = \mathbb{E}\left[\sum_{t=0}^{T} c(s_t^\pi, a_t^\pi) - J^\pi(s_t^\pi)\Big| s_0, a_0\right] .\end{aligned}$$

Under general conditions (Puterman 1994, Bertsekas 2000), h^π and Q^π are the unique solutions (up to an additive constant) of the so-called *Bellman equations*: for all states

[2] In general, there could be more than one constraints in a CMDP, in which case the optimal policy needs to randomize in more states. In fact, the number of states where randomization is necessary is equal to the number of constraints (Altman 1999, Sennott 1993).

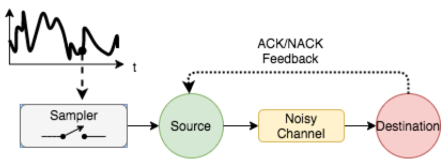

Figure 1.2 System model of a status-update system over an error-prone point-to-point link in the presence of ACK/NACK feedback from the destination.

$s \in \mathcal{S}$ and actions $a \in \mathcal{A}$,

$$Q^\pi(s,a) = c(s,a) - J^\pi(s,a) + \sum_{s' \in \mathcal{S}} P(s'|s,a) h^\pi(s');$$

$$h^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) Q^\pi(s,a) .$$

An optimal policy π^* for an MDP satisfies a slightly modified version of these equations, called the *Bellman optimality equations*: for all $s \in \mathcal{S}$,

$$h^{\pi^*}(s) + J^{\pi^*}(s) = \min_a c(s,a) + \sum_{s' \in \mathcal{S}} P(s'|s,a^{\pi^*}) h^{\pi^*}(s').$$

On the other hand, no sub-optimal policy can satisfy these equations. We will use Q and J to denote the state-action value function and the differential value function of an optimal policy π^*. It is easy to see that the Bellman optimality equations imply that, in every state s, an optimal policy chooses the greedy action minimizing the action-value function $Q(s,a)$ in a. There exists several algorithms in the literature which are based on the Bellman optimality equations and iteratively improve a policy whenever it violates these optimality conditions.

In the following sections, we model the AoI minimization problem under resource constraints using the MDP formulation defined above. We study many RL techniques for the AoI minimization problem in different settings, and compare their performances under different scenarios when the system characteristics are not known in advance, or change with time. We present average-cost RL algorithms to learn transmission policies when the environment determined by the status-update system is not known a priori, including, in particular, the case of unknown decoding error probabilities in a status-update system with HARQ (Ceran, Gündüz & György 2018, Ceran, Gündüz & György 2019, Ceran, Gündüz & György 2018), and unknown energy harvesting characteristics of the source node (Ceran, Gündüz & György 2019).

1.3 RL for Minimizing AoI in Point-to-Point Status-Update Systems

In this section, we consider a point-to-point wireless status-update system. The source monitors an underlying time-varying process, and can generate a status update at any time slot. The status updates are communicated from the source node to the destination over a time-varying channel (see Figure 1.2). Each transmission attempt of a status update takes constant time, set as one time slot. Throughout the chapter, we will normalize all time durations by the duration of one time slot. We assume that the wireless channel changes randomly from one time slot to the next in an independent and identically distributed (i.i.d.) fashion, and the channel state information is available only at the destination node. We further assume the availability of an error- and delay-free single-bit feedback from the destination to the source node for each transmission attempt. Successful receipt of a status update is acknowledged by an ACK signal, while a NACK signal is sent in case of a failure. In the classical ARQ protocol, a packet is retransmitted after each NACK feedback, until it is successfully decoded (or a maximum number of allowed retransmissions is reached), and the received signal is discarded after each failed transmission attempt. Therefore, the probability of error is the same for all retransmissions. However, in the AoI framework there is no point in retransmitting a failed out-of-date status packet if it has the same error probability as that of a fresh update. Hence, we assume that if the ARQ protocol is adopted, the source always removes failed packets and transmits a fresh status update. If the HARQ protocol is used, the received signals from all previous transmission attempts for the same packet are combined for decoding. Therefore, the probability of error decreases with every retransmission. In general, the error probability of each retransmission attempt depends on the particular combination technique used by the decoder, as well as on the channel conditions.

AoI measures the timeliness of the information at the receiver. It is defined as the number of time slots elapsed since the generation of the most up-to-date packet successfully decoded at the receiver. Formally, denoting the latter generation time for any time slot t by $U(t)$, the AoI, denoted by δ_t, is defined as

$$\delta_t \triangleq t - U(t). \tag{1.1}$$

A transmission decision is made at the beginning of each slot. The AoI increases by one when the transmission fails. When it is successfully received, it decreases to one in the case of ARQ, or to the number of retransmissions plus one in the case of HARQ (minimum age is set to 1 to reflect that the transmission is one slot long).

The probability of error after r retransmissions, denoted by $g(r)$, depends on r and the HARQ scheme. We assume that $g(r)$ is non-increasing in the number of retransmissions r. For simplicity, we assume that $0 < g(0) < 1$, that is, the channel is noisy and there is a possibility that the first transmission is successful. Also, we will denote the maximum number of retransmissions by r_{max}, which may take the value ∞, unless otherwise stated. However, if $g(r) = 0$ for some r (i.e., a packet is always correctly decoded after r retransmissions), we set r_{max} to be the smallest such r. Note that practical HARQ methods only allow a finite number of retransmissions (802.16e 2005 2006).

Let $\delta_t \in \mathbb{Z}^+$ denote the AoI at the beginning of the time slot t, and $r_t \in \{0, \dots, r_{max}\}$ denote the number of previous transmission attempts. Then the state of the system can be described by $s_t \triangleq (\delta_t, r_t)$. In each time slot, the source node takes one of the three actions, denoted by $a \in \mathcal{A}$, where $\mathcal{A} = \{\mathrm{i}, \mathrm{n}, \mathrm{x}\}$: (i) remain idle ($a = \mathrm{i}$); (ii) transmit a new status update ($a = \mathrm{n}$); or (iii) retransmit the previously failed update ($a = \mathrm{x}$).

If no resource constraint is imposed on the source, remaining idle is clearly suboptimal since it does not contribute to decreasing the AoI. However, continuous transmission is typically not possible in practice due to energy or interference constraints. Accordingly, we impose a constraint on the average number of transmissions, and we require that the long-term average number of transmissions do not exceed $C_{max} \in (0,1]$ (note that $C_{max} = 1$ corresponds to the case in which transmission is allowed in every slot).

This leads to the CMDP formulation, defined in Section 1.2: The countable set of states $(\delta, r) \in \mathcal{S}$ and the finite action set $\mathcal{A} = \{\mathrm{i}, \mathrm{n}, \mathrm{x}\}$ have already been defined. P will be explicitly defined in (1.4). The cost function $c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, is the AoI at the destination, and is defined as $c((\delta, r), a) = \delta$ for any $(\delta, r) \in \mathcal{S}$, $a \in \mathcal{A}$, independently of action a. The transmission cost $d : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is independent of the state and depends only on the action a, where $d = 0$ if $a = \mathrm{i}$, and $d = 1$ otherwise.

Let $J^\pi(s_0)$ and $C^\pi(s_0)$ denote the infinite horizon average age and the average number of transmissions, respectively. The CMDP problem can be stated as follows:

Problem 1

$$\text{Minimize } J^\pi(s_0) \triangleq \limsup_{T\to\infty} \frac{1}{T+1}\, \mathbb{E}\left[\sum_{t=0}^{T} \delta_t^\pi \middle| s_0\right], \tag{1.2}$$

$$\text{subject to } C^\pi(s_0) \triangleq \limsup_{T\to\infty} \frac{1}{T+1}\, \mathbb{E}\left[\sum_{t=0}^{T} \mathbb{1}[a_t^\pi \neq \mathrm{i}] \middle| s_0\right] \leq C_{max}. \tag{1.3}$$

Without loss of generality, we assume that the sender and the receiver are synchronized, that is, $s_0 = (1,0)$; and we omit s_0 from the notation for simplicity.

Before formally defining the transition function P, we present a simple observation that simplifies P: Retransmitting a packet immediately after a failed attempt is better than retransmitting it after waiting for some slots. This is true since waiting increases the age, without increasing the success probability.

Proposition 1 For any policy π there exists another policy π' (not necessarily distinct from π) such that $J^{\pi'}(s_0) \leq J^\pi(s_0)$, $C^{\pi'}(s_0) \leq C^\pi(s_0)$, and π' takes a retransmission action only following a failed transmission, that is, the probability $Pr(a_{t+1}^{\pi'} = \mathrm{x} | a_t^{\pi'} = \mathrm{i}) = 0$.

P are given as follows (omitting the parenthesis from the state variables (δ, r)):

$$\begin{aligned}
&\mathrm{P}(\delta+1, 0|\delta, r, \mathrm{i}) = 1,\\
&\mathrm{P}(\delta+1, 1|\delta, r, \mathrm{n}) = g(0),\\
&\mathrm{P}(1, 0|\delta, r, \mathrm{n}) = 1 - g(0),\\
&\mathrm{P}(\delta+1, r+1|\delta, r, \mathrm{x}) = g(r),\\
&\mathrm{P}(r+1, 0|\delta, r, \mathrm{x}) = 1 - g(r),
\end{aligned} \tag{1.4}$$

and $\mathrm{P}(\delta', r'|\delta, r, a) = 0$ otherwise. Note that the above equations set the retransmission count to 0 after each successful transmission, and it is not allowed to take a retransmission action in states where the transmission count is 0. Also, the property in Proposition 1 is enforced by the first equation in (1.4), that is, $\mathrm{P}(\delta+1, 0|\delta, r, \mathrm{i}) = 1$ (since retransmissions are not allowed in states $(\delta, 0)$). Since the starting state is $(1,0)$, it also follows that the state set of the CMDP can be described as

$$\mathcal{S} = \{(\delta, r) : r < \min\{\delta, r_{max} + 1\}, \delta, r \in \mathbb{N}\}\,. \tag{1.5}$$

1.3.1 Lagrangian Relaxation and the Structure of the Optimal Policy

In this section, we derive the structure of the optimal policy for Problem 1 based on (Sennott 1993). A detailed treatment of finite state-finite action CMDPs is considered by Altman (1999), but here we need more general results that apply to countable state spaces. These results require certain technical conditions; roughly speaking, there must exist a deterministic policy that satisfies the transmission constraint while maintaining a finite average AoI, and any "reasonable" policy must induce a positive recurrent Markov chain. The precise formulation of the requirements is given by Ceran, Gündüz & György (2019), wherein Proposition 2 of Ceran, Gündüz & György (2019) shows that the conditions of Sennott (1993) are satisfied for Problem 1. Given this result, we follow (Sennott 1993) to characterize the optimal policy.

While there exists a stationary and deterministic optimal policy for countable-state finite-action average-cost MDPs (Sennott 1989, Puterman 1994, Bertsekas 2000), this is not necessarily true for CMDPs (Sennott 1993, Altman 1999). To solve the CMDP, we start by rewriting the problem in its Lagrangian form. The average Lagrangian cost of a policy π with Lagrange multiplier $\eta \geq 0$ is defined as

$$L_\eta^\pi = \lim_{T\to\infty} \frac{1}{T+1} \left(\mathbb{E}\left[\sum_{t=0}^{T} \delta_t^\pi \right] + \eta \mathbb{E}\left[\sum_{t=0}^{T} \mathbb{1}[a_t^\pi \neq \mathrm{i}] \right] \right), \tag{1.6}$$

and, for any η, the optimal achievable cost L_η^* is defined as $L_\eta^* \triangleq \min_\pi L_\eta^\pi$. If the constraint on the transmission cost is less than one (i.e., $C_{max} < 1$), then we have $\eta > 0$, which will be assumed throughout the chapter.[3] This formulation is equivalent to an unconstrained countable-state average-cost MDP with overall cost $\delta_t + \eta\mathbb{1}[a_t^\pi \neq \mathrm{i}]$. A policy π is called η-optimal if it achieves L_η^*. Since the assumptions of Proposition 3.2 of Sennott (1993) are satisfied by Proposition 2 of Ceran, Gündüz & György (2019), the former implies that there exists a *differential cost function* $h_\eta(\delta, r)$ satisfying

$$h_\eta(\delta, r) + L_\eta^* = \min_{a\in\{\mathrm{i,n,x}\}} \left(\delta + \eta \cdot \mathbb{1}[a \neq \mathrm{i}] + \mathbb{E}\left[h_\eta(\delta', r')\right]\right), \tag{1.7}$$

for all states $(\delta, r) \in \mathcal{S}$, where (δ', r') is the next state after taking action a.

We also introduce the *state-action cost function* defined as

$$Q_\eta(\delta, r, a) \triangleq \delta + \eta \cdot \mathbb{1}[a \neq \mathrm{i}] + \mathbb{E}\left[h_\eta(\delta', r')\right] \tag{1.8}$$

for all $(\delta, r) \in \mathcal{S}, a \in \mathcal{A}$. Then, also implied by Proposition 3.2 of Sennott (1993), the optimal deterministic policy for the Lagrangian problem with a given η takes, for any $(\delta, r) \in \mathcal{S}$, the action achieving the minimum in (1.8):

$$\pi_\eta^*(\delta, r) \in \operatorname*{arg\,min}_{a\in\{\mathrm{i,n,x}\}} Q_\eta(\delta, r, a) . \tag{1.9}$$

Focusing on deterministic policies, we can characterize the optimal policies for the CMDP problem: Based on Theorem 2.5 of Sennott (1993), we can prove the following:

[3] If $C_{max} = 1$, a transmission is allowed in every time slot, and we have an infinite state-space MDP with unbounded cost. It follows directly from part (ii) of the theorem of Sennott (1989) (whose conditions can be easily verified for Problem 1) that there exists an optimal stationary policy that satisfies the Bellman equations. In this chapter, we concentrate on the more interesting constrained case.

THEOREM 1.1 *There exists an optimal stationary policy for the CMDP in Problem 1 that is optimal for the unconstrained problem considered in* (1.6) *for some $\eta = \eta^*$, and randomizes in at most one state. This policy can be expressed as a mixture of two deterministic policies $\pi^*_{\eta^*,1}$ and $\pi^*_{\eta^*,2}$ that differ in at most a single state s, and are both optimal for the Lagrangian problem* (1.6) *with $\eta = \eta^*$. More precisely, there exists $\mu \in [0,1]$ such that the mixture policy $\pi^*_{\eta^*}$, which selects, in state s, $\pi^*_{\eta^*,1}(s)$ with probability μ and $\pi^*_{\eta^*,2}(s)$ with probability $1-\mu$, and otherwise follows these two policies (which agree in all other states) is optimal for Problem 1, and* (1.3) *is satisfied with equality.*

Proof By Proposition 2 of Ceran, Gündüz & György (2019), Theorem 2.5, Proposition 3.2, and Lemma 3.9 of Sennott (1993) hold for Problem 1. By Theorem 2.5 of Sennott (1993), there exists an optimal stationary policy that is a mixture of two deterministic policies, $\pi^*_{\eta^*,1}$ and $\pi^*_{\eta^*,2}$, which differ in at most one state and are η^*-optimal by Proposition 3.2 of Sennott (1993) satisfying (1.7) and (1.8). From Lemma 3.9 of Sennott (1993), the mixture policy π^*_{μ}, for any $\mu \in [0,1]$, also satisfies (1.7) and (1.8), and is optimal for the unconstrained problem in (1.6) with $\eta = \eta^*$. From the proof of Theorem 2.5 of Sennott (1993), there exists a $\mu \in [0,1]$ such that $\pi^*_{\eta^*}$ satisfies the constraint in (1.3) with equality. This completes the proof of the theorem. □

Some other results of Sennott (1993) will be useful in determining $\pi^*_{\eta^*}$. For any $\eta > 0$, let C_η and J_η denote the average number of transmissions and average AoI, respectively, for the optimal policy π^*_η. Note that these are multivalued functions since there might be more than one optimal policy for a given η. Note also that, C_η and J_η can be computed directly by finding the stationary distribution of the chain, or estimated empirically by running the MDP with policy π^*_η. From Lemma 3.4 of Sennott (1993), L^*_η, C_η and J_η are monotone functions of η: if $\eta_1 < \eta_2$, we have $C_{\eta_1} \geq C_{\eta_2}$, $J_{\eta_1} \leq J_{\eta_2}$ and $L^*_{\eta_1} \leq L^*_{\eta_2}$. This statement is also intuitive since η effectively represents the cost of a single transmission in (1.7) and (1.8), as η increases, the average number of transmissions of the optimal policy cannot increase, and as a result, the AoI cannot decrease.

To determine the optimal policy, one needs to find η^*, the policies $\pi^*_{\eta^*,1}$ and $\pi^*_{\eta^*,2}$, and the weight μ. In fact, (Sennott 1993) shows that η^* is defined as

$$\eta^* \triangleq \inf\{\eta > 0 : C_\eta \leq C_{max}\}, \tag{1.10}$$

where the inequality $C_\eta \leq C_{max}$ is satisfied if it is satisfied for at least one value of (multivalued) C_η. By Lemma 3.12 of Sennott (1993), η^* is finite, and $\eta^* > 0$ if $C_{max} < 1$.

If $C^{\pi^*_{\eta^*,i}} = C_{max}$ for $i = 1$ or $i = 2$, then it is the optimal policy, that is, $\pi^*_\mu = \pi^*_{\eta^*,i}$ and $\mu = 1$ if $i = 1$ and 0 if $i = 2$. Otherwise one needs to select μ such that $C^{\pi^*_\mu} = C_{max}$: that is, if $C^{\pi^*_{\eta^*,2}} < C_{max} < C^{\pi^*_{\eta^*,1}}$, then

$$\mu = \frac{C_{max} - C^{\pi^*_{\eta^*,2}}}{C^{\pi^*_{\eta^*,1}} - C^{\pi^*_{\eta^*,2}}}, \tag{1.11}$$

which results in an optimal policy.

In practice, finding both η^* and the policies $\pi^*_{\eta^*,1}$ and $\pi^*_{\eta^*,2}$ is hard. However, given two monotone sequences $\eta_n \uparrow \eta^*$ and $\eta'_n \downarrow \eta^*$, there is a subsequence of η_n (resp., η'_n) such that the corresponding subsequence of the η_n-optimal policies $\pi^*_{\eta_n}$ (η'_n-optimal policies

$\pi^*_{\eta'_n}$, resp.) satisfying the Bellman equation (1.7) converge[4]. Then the limit points π and π' are η^*-optimal by Lemma 3.7 (iii) of Sennott (1993) and $C^{\pi} \geq C_{max} \geq C^{\pi'}$ by the monotonicity of C_η and the same Lemma 3.7. Although there is no guarantee that π and π' only differ in a single point, we can combine them to get an optimal randomized policy using μ defined in (1.11). In this case, Lemma 3.9 of Sennott (1993) implies that the policy that first randomly selects if it should use π or π' (choosing π with probability μ) and then uses the selected policy forever is η^*-optimal. However, since $(1, 0)$ is a positive recurrent state of both policies and they have a single recurrent class by Proposition 3.2 of Sennott (1993), we can do the random selection of between π and π' independently every time the system gets into state $(1, 0)$ without changing the long-term average or expected AoI and transmission cost (note that one cannot choose randomly between the two policies in, e.g., every step). Thus, the resulting randomized policy is η^*-optimal, and since μ is selected in such a way that the total transmission cost is C_{max}, it is also an optimal solution of Problem 1 by Lemma 3.10 of Sennott (1993). Note that to derive two η^*-optimal policies, which provably differ only in a single state, a much more elaborate construction is used in (Sennott 1993). However, in practice, π and π' obtained above are often the same except for a single state. Furthermore, we can approximate π_1 and π_2 by $\pi^*_{\eta_n}$ and $\pi^*_{\eta'_n}$ for n large enough.

Theorem 1.1 and the succeeding discussion present the general structure of the optimal policy. In Section 1.3.2, for practical implementation, a computationally efficient heuristic algorithm is proposed based upon the discussion in this section.

1.3.2 Relative Value Iteration (RVI)

While the state space is countably infinite, since the age can be arbitrarily large, in practice we can approximate the countable state space with a large but finite space by setting an upper bound on the age (which will be denoted by N), and by selecting a finite r_{max}. When we consider the finite state space approximation of the problem, we can employ the *relative value iteration* (RVI) (Puterman 1994) algorithm to solve (1.7) for any given η, and hence find (an approximation of) the optimal policy π^*_η.

Starting with an initialization of $h_0(\delta, r)$, $\forall(\delta, r)$, and setting an arbitrary but fixed reference state (δ^{ref}, r^{ref}), a single iteration for the RVI algorithm is given as follows:

$$Q_{n+1}(\delta, r, a) \leftarrow \delta + \eta \cdot \mathbb{1}[a^{\pi} \neq \mathrm{i}] + \mathbb{E}\left[h_n(\delta', r')\right], \tag{1.12}$$

$$V_{n+1}(\delta, r) \leftarrow \min_a(Q_{n+1}(\delta, r, a)), \tag{1.13}$$

$$h_{n+1}(\delta, r) \leftarrow V_{n+1}(\delta, r) - V_{n+1}(\delta^{ref}, r^{ref}), \tag{1.14}$$

where $Q_n(\delta, r, a)$, $V_n(\delta, r)$ and $h_n(\delta', r')$ denote the state action value function, value function and differential value function for iteration n, respectively. Then, h_n converges to h_η, and $\pi^*_n(\delta, r) \triangleq \arg\min_a Q_n(\delta, r, a)$ converges to $\pi^*_\eta(\delta, r)$ (Puterman 1994).

After computing the optimal deterministic policy π^*_η for any given η (more precisely, an arbitrarily close approximation in the finite approximate MDP), we need to find η^* as

[4] $\pi_n \rightarrow \pi$ if for any state s, $\pi_n(s) = \pi(s)$ for n large enough.

Algorithm 1 Average-cost SARSA with Softmax

```
Input: Lagrange parameter $\eta$                /* error probability $g(r)$ is unknown */
 1: $n \leftarrow 0$                                          /* time iteration */
 2: $\tau \leftarrow 1$                          /* softmax temperature parameter */
 3: $Q_\eta^{N\times M\times 3} \leftarrow 0$
 4: $L_\eta \leftarrow 0$                               /* initialization of the gain */
 5: for $n$ do
 6:     Observe current state $s_n$
 7:     for $a \in \mathcal{A}$ do $\pi(a|s_n) = \frac{\exp(-Q_\eta(s_n,a)/\tau)}{\sum_{a'\in\mathcal{A}} \exp(-Q_\eta(s_n,a')/\tau)}$   /* since it is a minimization problem, use minus Q function in softmax */
 8:     end for
 9:     Sample $a_n$ from $\pi(a|S_n)$
10:     Observe next state $s_{n+1}$ and cost $c_n = \delta_n + \eta 1_{\{a_n=1,2\}}$
11:     for $a \in \mathcal{A}$ do
            $\pi(a|s_{n+1}) = \frac{\exp(-Q_\eta(s_{n+1},a_{n+1})/\tau)}{\sum_{a'_{n+1}\in\mathcal{A}} \exp(-Q_\eta(s_{n+1},a'_{n+1})/\tau)}$
12:     end for
13:     Sample $a_{n+1}$ from $\pi(a_{n+1}|s_{n+1})$
14:     Update
15:     $\alpha_n \leftarrow 1/\sqrt{n}$                                /* update parameter */
16:     $Q_\eta(s_n,a_n) \leftarrow Q_\eta(s_n,a_n) + \alpha_n[\delta + \eta \cdot \mathbb{1}[a_n \neq \text{i}] - J_\eta + Q_\eta(s_{n+1},a_{n+1}) - Q_\eta(s_n,a_n)]$
17:     $L_\eta \leftarrow L_\eta + 1/n[\delta + \eta \cdot \mathbb{1}[a_n \neq \text{i}] - J_\eta]$      /* update $L_\eta$ at every step */
18:     $n \leftarrow n+1$                                /* increase iteration count */
19: end for
```

defined by (1.10). We can use the following heuristic: With the aim of finding a single η value with $C_\eta \approx C_{max}$, we start with an initial parameter η^0, and update η iteratively as $\eta^{m+1} = \eta^m + \alpha_m(C_{\eta^m} - C_{max})$ for a step size parameter α_m[5].

1.3.3 Practical RL Algorithms

We now assume that the source does not have *a priori* information about the decoding error probabilities, and has to learn them. The literature for average-cost RL is quite limited compared to discounted cost problems (Mahadevan 1996, Sutton & Barto 1998). SARSA (Sutton & Barto 1998) is a well-known RL algorithm, originally proposed for discounted MDPs, that iteratively computes the optimal state-action value function $Q(s,a)$ and the optimal policy for an MDP based on the action performed by the current policy in a recursive manner. We employ an average-cost version of SARSA with *Boltzmann* (*softmax*) exploration to learn $g(r)$ without degrading the performance significantly. The resulting algorithm is called *average-cost SARSA with softmax*.

Average-cost SARSA with softmax starts with an initial estimate of $Q_\eta(s,a)$ and finds the optimal policy by estimating state-action values in a recursive manner. In the n^{th} iteration, after taking action a_n, the source observes the next state s_{n+1} and the instantaneous cost value c_n. Based on this, the estimate of $Q_\eta(s,a)$ is updated by weighing the

[5] α_m is a positive decreasing sequence and satisfies the following conditions: $\sum_m \alpha_m = \infty$ and $\sum_m \alpha_m^2 < \infty$ from the theory of stochastic approximation (Kushner & Yin 1997).

previous estimate and the estimated expected value of the current policy in the next state s_{n+1}. The instantaneous cost c_n is the sum of AoI and the weighted cost of transmission, i.e. $\delta_n + \eta \cdot \mathbb{1}[a_n \neq \mathrm{i}]$; hence, it is readily known at the source node. In each time slot, the learning algorithm (see Algorithm 1)

- observes the current state $s_n \in \mathcal{S}$,
- selects and performs an action $a_n \in \mathcal{A}$,
- observes the next state $s_{n+1} \in \mathcal{S}$ and the instantaneous cost c_n,
- updates its estimate of $Q_\eta(s_n, a_n)$ using the current estimate of η by

$$Q_\eta(s_n, a_n) \leftarrow Q_\eta(s_n, a_n) + \alpha_n[\delta + \eta \cdot \mathbb{1}[a_n \neq \mathrm{i}] - L_\eta + Q_\eta(s_{n+1}, a_{n+1}) - Q_\eta(s_n, a_n)], \tag{1.15}$$

  where α_n is the update parameter (learning rate) in the n^{th} iteration.
- updates its estimate of L_η based on empirical average.

As we discussed earlier, with the accurate estimate of $Q_\eta(s, a)$ at hand the transmitter can decide for the optimal actions for a given η as in (1.9). However, until the state-action cost function is accurately estimated, the transmitter action selection method should balance the *exploration* of new actions with the *exploitation* of actions known to perform well. In particular, the *Boltzmann* action selection method, which chooses each action probabilistically relative to its expected cost, is used in this chapter. The source assigns a probability to each action for a given state s_n, denoted by $\pi(a|s_n)$:

$$\pi(a|s_n) \triangleq \frac{\exp(-Q_\eta(s_n, a)/\tau)}{\sum\limits_{a' \in \mathcal{A}} \exp(-Q_\eta(s_n, a')/\tau)}, \tag{1.16}$$

where τ is called the temperature parameter such that high τ corresponds to more uniform action selection (exploration) whereas low τ is biased toward the best action (exploitation).

The constrained structure of the average AoI problem requires additional modifications to the algorithm, which is achieved by updating the Lagrange multiplier according to the empirical resource consumption. In each time slot, we keep track of a value η resulting in a transmission cost close to C_{max}, and then find and apply a policy that is optimal (given the observations so far) for the MDP with Lagrangian cost as in Algorithm 1.

1.3.4 Simulation Results

For the numerical simulation, we assume that decoding error reduces exponentially with the number of retransmission, that is, $g(r) \triangleq p_0\lambda^r$ for some $\lambda \in (0, 1)$, where p_0 denotes the error probability of the first transmission, and r is the retransmission count (set to 0 for the first transmission). The exact value of λ depends on the particular HARQ protocol and the channel model. Note that ARQ corresponds to the case with $\lambda = 1$ and $r_{max} = 0$. Following the *IEEE 802.16* standard(802.16e 2005 2006), the maximum number of retransmissions is set to $r_{max} = 3$.

Figure 1.3 shows the evolution of the average AoI over time with the average-cost

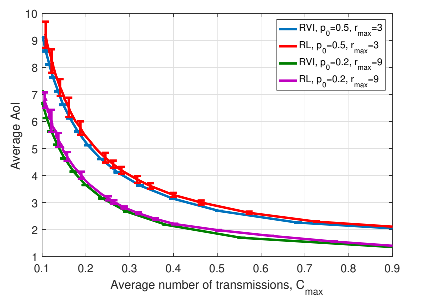

Figure 1.3 Performance of the average-cost SARSA for $r_{max} = 3$, $p_0 = 0.5$, $\lambda = 0.5$, $C_{max} = 0.4$ and $n = 10000$, averaged over 1000 runs (both the mean and the variance are shown).

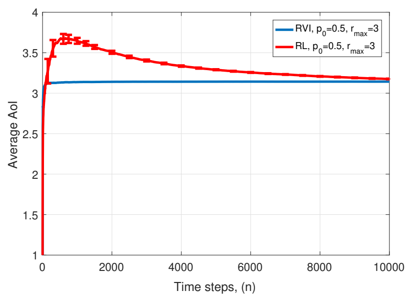

Figure 1.4 Performance of the proposed RL algorithm (average-cost SARSA) and its comparison with the RVI algorithm for $n = 10000$ iterations, and values are averaged over 1000 runs for different p_0 and r_{max} values when $\lambda = 0.5$ (both the mean and the variance are shown).

SARSA algorithm. The average AoI achieved by Algorithm 1, denoted by *RL* in the figure, converges to the one obtained from the RVI algorithm, which has *a priori* knowledge of $g(r)$. We can observe from Figure 1.3 that the performance of SARSA approaches that of RVI in about 10000 iterations. Figure 1.4 shows the performance of the two algorithms (with again 10000 iterations in SARSA) as a function of C_{max} in two different setups. We can see that SARSA performs very close to RVI with a gap that is roughly constant for the whole range of C_{max} values. We can also observe that the variance of the average AoI achieved by SARSA is much larger when the number of transmissions is limited, which also limits the algorithm's learning capability.

1.4 RL for Minimizing AoI in Multi-User Status-Update Systems

In this section, we consider a status-update system with M users. The source can transmit the status update to only a single user in each time slot. This can be either because of dedicated orthogonal links to the users, or because the users are interested in distinct processes. As before, each transmission attempt takes one time slot, and the channels change randomly from one time slot to the next in an i.i.d. fashion, with states known only at the corresponding receivers. Successful reception of the status update is acknowledged by an ACK signal (denoted by $K_t = 1$), while a NACK signal is sent in case of a failure (denoted by $K_t = 0$).

Since, in practice, the utility of status updates typically becomes zero beyond a certain age, and also to simplify the analysis, we assume that the age cannot grow larger than some finite constant δ_{max}. Then, if the most up-to-date packet received by the j^{th} user ($j \in [M] \triangleq \{1, \ldots, M\}$) before time slot t was generated in slot $U_j(t)$, then the AoI at user j at the beginning of time slot t is defined as $\delta_{j,t} \triangleq \min\{t - U_j(t), \delta_{max}\} \in [\delta_{max}] \triangleq \{1, \ldots, \delta_{max}\}$.

In each time slot t, the source node takes an action a_t from the set of actions $\mathcal{A} = \{\mathrm{i}, \mathrm{n}_1, \mathrm{x}_1, \ldots, \mathrm{n}_M, \mathrm{x}_M\}$: in particular, the source can i) remain idle ($a_t = \mathrm{i}$); ii) generate and transmit a new status update packet to the j^{th} user ($a_t = \mathrm{n}_j$, $j \in [M]$); or, iii) retransmit the most recent failed status update to the j^{th} user ($a_t = \mathrm{x}_j$, , $j \in [M]$). We have $|\mathcal{A}| = 2M + 1$. For the j^{th} user, the probability of error after r retransmissions, denoted by $g_j(r)$.

Let $\delta^{tx}_{j,t}$ denote the number of time slots elapsed since the generation of the most recently transmitted (whether successfully or not) packet to user j at the transmitter, while $\delta^{rx}_{j,t}$ denote the AoI of the most recently received status update at receiver of the user j. $\delta^{tx}_{j,t}$ resets to 1 if a new status update is generated in time slot $t-1$, and increases by one (up to δ_{max}) otherwise, i.e.,

$$\delta^{tx}_{j,t+1} = \begin{cases} 1 & \text{if } a_t = \mathrm{n}_j; \\ \min(\delta^{tx}_{j,t} + 1, \delta_{max}) & \text{otherwise.} \end{cases}$$

On the other hand, the AoI at the receiver side evolves as follows:

$$\delta^{rx}_{j,t+1} = \begin{cases} 1 & \text{if } a_t = \mathrm{n}_j \text{ and } K_t = 1; \\ \min(\delta^{tx}_{j,t} + 1, \delta_{max}) & \text{if } a_t = \mathrm{x}_j \text{ and } K_t = 1; \\ \min(\delta^{rx}_{j,t} + 1, \delta_{max}) & \text{otherwise .} \end{cases}$$

Note that once the AoI at the receiver is at least as large as at the transmitter, this relationship holds forever; thus it is enough to consider cases when $\delta^{rx}_t \geq \delta^{tx}_t$.

Therefore, $\delta^{rx}_{j,t}$ increases by 1 when the source chooses to transmit to another user, or if the transmission fails, while it decreases to 1 or, in the case of HARQ, to $\min(\delta^{tx}_{j,t} + 1, \delta_{max})$, when a status update is successfully decoded. Also, $\delta^{tx}_{j,t}$ increases by 1 if the source chooses not to generate a new packet and transmit it to user j ($a_t \neq \mathrm{n}_j$).

For the j^{th} user, let $r_{j,t} \in [r_{max}] \triangleq \{0, \ldots, r_{max}\}$ denote the number of previous transmission attempts of the most recent packet. Thus, the number of retransmissions is zero for a newly sensed and generated status update and increases up to r_{max} as we keep retransmitting. Then, the state of the system can be described by $s_t \triangleq (\delta^{rx}_{1,t}, \delta^{tx}_{1,t}, r_{1,t}, \ldots, \delta^{rx}_{M,t}, \delta^{tx}_{M,t}, r_{M,t})$, where $s_t \in \mathcal{S} \subset ([\delta_{max}] \times [\delta_{max}] \times [r_{max}])^M$.

Similarly to previous section, we impose a constraint on the average number of transmissions, denoted by $C_{max} \in (0, 1]$. This leads to CMDP formulation, defined in Section 1.2: The set of states $\mathcal{S}$ and the finite set of actions $\mathcal{A}$ have already been defined. $\mathcal{P}$

can be summarized as follows.

$$\mathcal{P}_{s,s'}(a) = \begin{cases} 1 & \text{if } a = \mathrm{i},\ \delta_i^{rx'} = \min\{\delta_i^{rx}+1, \delta_{max}\},\ \delta_i^{tx'} = \min\{\delta_i^{tx}+1, \delta_{max}\}, \\ & r_i' = r_i,\ \forall i; \\ 1 - g_j(0) & \text{if } a = \mathrm{n}_j,\ \delta_j^{rx'} = 1,\ \delta_j^{tx'} = 1,\ r_j' = 0,\ \delta_i^{rx'} = \min\{\delta_i^{rx}+1, \delta_{max}\}, \\ & \delta_i^{tx'} = \min\{\delta_i^{tx}+1, \delta_{max}\},\ r_i' = r_i,\ \forall i \neq j; \\ g_j(0) & \text{if } a = \mathrm{n}_j,\ \delta_j^{rx'} = \min\{\delta_j^{rx}+1, \delta_{max}\},\ \delta_j^{tx'} = 1,\ r_j' = 1,\ r_i' = r_i; \\ & \delta_i^{rx'} = \min\{\delta_i^{rx}+1, \delta_{max}\},\ \delta_i^{tx'} = \min\{\delta_i^{tx}+1, \delta_{max}\}, \forall i \neq j; \\ 1 - g_j(r_j) & \text{if } a = \mathrm{x}_j,\ \delta_j^{rx'} = \delta_j^{tx}+1,\ \delta_j^{tx'} = \min\{\delta_j^{tx}+1, \delta_{max}\},\ r_j' = 0,\ r_i' = r_i, \\ & \delta_i^{rx'} = \min\{\delta_i^{rx}+1, \delta_{max}\},\ \delta_i^{tx'} = \min\{\delta_i^{tx}+1, \delta_{max}\}, \forall i \neq j; \\ g_j(r_j) & \text{if } a = \mathrm{x}_j,\ \delta_j^{rx'} = \min\{\delta_j^{rx}+1, \delta_{max}\},\ \delta_j^{tx'} = \min\{\delta_j^{tx}+1, \delta_{max}\}, \\ & \delta_i^{rx'} = \min\{\delta_i^{rx}+1, \delta_{max}\},\ \delta_i^{tx'} = \min\{\delta_i^{tx}+1, \delta_{max}\}, \\ & r_j' = \min\{r_j'+1, r_{max}\},\ r_i' = r_i, \forall i \neq j; \\ 0 & \text{otherwise.} \end{cases} \tag{1.17}$$

The instantaneous cost function $c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is defined as the weighted sum of AoI for multiple users, independently of a. Formally, $c(s,a) = \Delta \triangleq w_1\delta_1^{rx} + \cdots + w_M\delta_M^{rx}$, where the weight $w_j > 0$ represents priority of user j. The instantaneous transmission cost $d : \mathcal{A} \to \mathbb{R}$ is defined as $d(\mathrm{i}) = 0$ and $d(a) = 1$ if $a \neq \mathrm{i}$. We use $s_t^\pi = (\delta_{1,t}^{rx\,\pi}, \delta_{1,t}^{tx\,\pi}, r_{1,t}^\pi, \dots, \delta_{M,t}^{rx\,\pi}, \delta_{M,t}^{tx\,\pi}, r_{M,t}^\pi)$ and a_t^π to denote the sequences of states and actions, respectively, induced by policy π, while $\Delta_t^\pi \triangleq \sum_{j=1}^M w_j\delta_{j,t}^{rx\pi}$ denotes the instantaneous weighted cost.

The infinite horizon expected weighted average AoI for policy π starting from the initial state $s_0 \in \mathcal{S}$ is defined as

$$J^\pi(s_0) \triangleq \limsup_{T\to\infty} \frac{1}{T+1}\mathbb{E}\left[\sum_{t=0}^T \Delta_t^\pi \Big| s_0\right], \tag{1.18}$$

while the average number of transmissions is given by (1.3).

As before, we assume that the source and the users are synchronized at the beginning of the problem, that is, $s_0 = (1, 0, 2, 0, \dots, M, 0)$; and we omit s_0 from the notation for simplicity.

Problem 2 $\quad \underset{\pi\in\Pi}{\text{Minimize}}\ J^\pi(s_0)$ over $\pi \in \Pi$ such that $C^\pi(s_0) \leq C_{max}$.

Similarly to Section 1.3.1, we can rewrite the problem in its Lagrangian form under policy π with Lagrange multiplier $\eta \geq 0$, denoted by L_η^π,

$$L_\eta^\pi = \lim_{T\to\infty} \frac{1}{T+1}\left(\mathbb{E}\left[\sum_{t=0}^T \Delta_t^\pi\right] + \eta\mathbb{E}\left[\sum_{t=0}^T \mathbb{1}[a_t^\pi \neq \mathrm{i}]\right]\right) \tag{1.19}$$

and, for any η, the optimal achievable cost is defined as $L_\eta^* \triangleq \inf_\pi L_\eta^\pi$ under policy π_n^*. This formulation is equivalent to an unconstrained finite-state average-cost MDP with the instantaneous overall cost $\Delta_t^\pi + \eta\mathbb{1}[a_t^\pi \neq \mathrm{i}]$.

THEOREM 1.2 *An optimal stationary policy π_n^* which minimizes* (1.19) *exists with constant L_η^* for the unconstrained MDP with Lagrangian parameter η.*

The proof of Theorem 1.2 can be found in (Ceran, Gündüz & György 2020). Also, as in Section 1.3.2, an iterative algorithm to minimize average AoI in multi-user systems can be designed by applying RVI algorithm (Puterman 1994).

1.4.1 AoI with Standard ARQ Protocol

In this section, we assume that the system adopts the standard ARQ protocol. The action space reduces to $\mathcal{A} = \{\mathrm{i}, \mathrm{n}_1, \ldots, \mathrm{n}_M\}$ and the state space reduces to $(\delta_1^{rx}, \delta_2^{rx}, \ldots, \delta_M^{rx})$ as $r_{j,t} = 0,\ \forall j, t$, and there is no need to store the AoI at the transmitter-side. The probability of error of each status update is $p_j \triangleq g_j(0)$ for user j. State transitions in (1.17), the Bellman optimality equations, and the RVI algorithm can all be simplified accordingly. Thanks to these simplifications, we are able to show the structure of the optimal policy and to derive a low-complexity suboptimal policy.

Structure of the Optimal Policy

THEOREM 1.3 *There exists an optimal stationary policy for Problem 2 under standard ARQ that is optimal for the unconstrained problem considered in* (1.6) *for some $\eta = \eta^*$, and randomizes in at most one state. This policy can be expressed as a mixture of two deterministic policies $\pi_{\eta^*,1}^*$ and $\pi_{\eta^*,2}^*$ that differ in at most a single state $\hat{s}$, and are both optimal for the Lagrangian problem* (1.6) *with $\eta = \eta^*$. More precisely, there exist two deterministic policies $\pi_{\eta^*,1}^*$, $\pi_{\eta^*,2}^*$ as described above and $\mu \in [0, 1]$, such that the mixture policy $\pi_{\eta^*}^*$, which selects, in state $\hat{s}$, $\pi_{\eta^*,1}^*(\hat{s})$ with probability μ and $\pi_{\eta^*,2}^*(\hat{s})$ with probability $1 - \mu$, and otherwise follows these two policies (which agree in all other states) is optimal for Problem 2, and the constraint in (1.3) is satisfied with equality.*

Theorem 1.3 is proved by Ceran et al. (2020). Some other results of Altman (1999) and Beutler & Ross (1985) are useful in determining $\pi_{\eta^*}^*$. For any $\eta > 0$, let C_η and J_η denote the average number of transmissions and average AoI, respectively, for the optimal policy π_η^*. Note that, C_η and J_η can be computed directly by finding the stationary distribution of the chain, or estimated empirically by running the MDP with policy π_η^*.

A detailed discussion on finding both η^* and the policies $\pi_{\eta^*,1}^*$ and $\pi_{\eta^*,2}^*$ are given by Ceran, Gündüz & György (2019) and also in Section 1.3.

The Whittle Index Policy

Although the RVI algorithm (Puterman 1994) provides an optimal solution to Problem 2, its computational complexity is significant for large networks consisting of many users. Instead, we can derive a low-complexity policy for the multi-user AoI minimization problem with standard ARQ based on Whittle's approach (Whittle 1988), by modelling the problem as a restless multi-armed bandit problem (Gittins, Glazebrook & Weber 2011). The Whittle index (WI) policy in Section 1.4.1 gives a possibly suboptimal yet computationally efficient policy, which often performs very well in practice.

Multi-armed bandit (MAB) problems (Gittins et al. 2011) constitute a class of RL problems with a single state. In the restless MAB (RMAB) problem (Whittle 1988), each arm is associated with a state that evolves over time, and the reward distribution of the arm depends on its state (in contrast, in stochastic MAB problems, rewards are i.i.d.). The multi-user AoI minimization problem with ARQ can be formulated as a RMAB with $M+1$ arms: choosing arm j is associated with transmitting to user j, while arm $M+1$ represents the action of staying idle ($a = \mathrm{i}$). RMAB problems are known to be PSPACE-hard in general (Gittins et al. 2011); however, a low-complexity heuristic policy can be found for certain problems by relaxing the constraint that in every round only a single arm can be selected, and instead introducing a bound on the expected number of arms chosen (Whittle 1988). The resulting policy, known as the WI policy, is a sub-optimal policy, but it is known to perform close to optimal in many settings (Whittle 1988).

Following Whittle's approach, we decouple the problem into M sub-problems each corresponding to a single user, and treat these problems independently. The cost of transmitting to a user (called *subsidy for passivity* (Whittle 1988)) is denoted by C, which will be later used to derive the index policy. Writing the Bellman equation (1.8) for each subproblem, we obtain the optimality equations for the single user AoI minimization problem with the standard ARQ protocol where the action space is $\{\mathrm{i}, \mathrm{n}_j\}$

$$h_C(\delta_j^{rx}) + L_j^* = \min\{Q(\delta_j^{rx}, \mathrm{n}_j), Q(\delta_j^{rx}, \mathrm{i})\}, \tag{1.20}$$

and the optimal policy to each subproblem is given

$$\pi_C^*(\delta_j^{rx}) \in \underset{a\in\{\mathrm{i},\mathrm{n}_j\}}{\arg\min}\{Q(\delta_j^{rx}, a)\}, \text{ where} \tag{1.21}$$

$$Q(\delta_j^{rx}, \mathrm{n}_j) \triangleq w_j\delta_j^{rx} + C + p_j h_C(\delta_j^{rx}+1) + (1-p_j)h_C(1), \quad Q(\delta_j^{rx}, \mathrm{i}) \triangleq w_j\delta_j^{rx} + h_C(\delta_j^{rx}+1).$$

Given (1.20) and (1.21), let $S_j^{\mathrm{n}_j}(C)$ represent the set of states the optimal action is equal to n_j for a given C, that is, $S_j^{\mathrm{n}_j}(C) = \{s : \pi_C^*(\delta_j^{rx}) = \mathrm{n}_j\}$. Then, we define indexability as follows.

DEFINITION 1.4 An arm is indexable if the set $S_j^{\mathrm{n}_j}(C)$ as a function of C is monotonically decreasing for $C \in \mathbb{R}$, and $\lim_{C\to\infty} S_j^{\mathrm{n}_j}(C) = \varnothing$ and $\lim_{C\to-\infty} S_j^{\mathrm{n}_j}(C) = \mathcal{S}$ (Whittle 1988, Gittins et al. 2011). The problem is indexable if every arm is indexable.

Note that if a problem is indexable as defined in Definition 1.4, $S_j^a(C_1) \subset S_j^a(C_2)$ for $C_1 \geq C_2$, and there exist a C such that both actions are *equally desirable*, that is, $Q(\delta_j^{rx}, \mathrm{i}) = Q(\delta_j^{rx}, \mathrm{n}_j)$ for all δ_j^{rx}. The WI is defined as follows.

DEFINITION 1.5 *The WI for user j at state δ_j^{rx}, denoted by $I_j(\delta_j^{rx})$, is defined as the cost C that makes both actions* n_j *and* i *equally desirable.*

The next proposition, proved by Ceran et al. (2020), gives a closed for expression for the WI in our setting:

Proposition 2 Problem 2 with standard ARQ is indexable and the WI for each user j

and state δ_j^{rx} can be computed as

$$I_j(\delta_j^{rx}) = \frac{1}{2} w_j \delta_j^{rx}(1 - p_j)\left(\delta_j^{rx} + \frac{1 + p_j}{1 - p_j}\right), \ \forall j \in [M], \tag{1.22}$$

where the WI for the idle action is $I_{M+1} = \eta$.

The WI policy, parametrized by the Lagrange multiplier $\eta > 0$, is defined as follows: in state $(\delta_1^{rx}, \delta_2^{rx}, \ldots, \delta_M^{rx})$, compare the highest index with η, and if η is smaller, then the source transmits to the user with the highest index, otherwise the source remains idle. This policy tends to transmit to the user with a high weight (w_j), low error probability (p_j) and high AoI (δ_j^{rx}). Formally,

$$\pi(\delta_1, \delta_2, \ldots, \delta_M) = \begin{cases} \mathrm{n}_{\arg\max_j (I_j(\delta_j^{rx}))} & \text{if } \max_j I_j(\delta_j^{rx}) \geq \eta, \\ \mathrm{i} & \text{otherwise.} \end{cases} \tag{1.23}$$

The effectiveness of the WI policy is demonstrated numerically in Section 1.4.3.

Lower Bound on the Average AoI under a Resource Constraint

In this section, we derive a closed-form lower bound for the constrained MDP, for which the proof is given in (Ceran et al. 2020):

THEOREM 1.6 *For Problem 2 with the standard ARQ protocol, we have $J_{LB} \leq J^{\pi}$, $\forall \pi \in \Pi$, where*

$$J_{LB} = \frac{1}{2C_{max}}\left(\sum_{j=1}^{M} \sqrt{\frac{w_j}{1 - p_j}}\right)^2 + \frac{C_{max} w_{j^*} p_{j^*}}{2(1 - p_{j^*})} + \frac{1}{2}\sum_{j=1}^{M} w_j, \text{and } j^* \triangleq \arg\min_j \frac{w_j p_j}{2(1 - p_j)}.$$

Previously, (Kadota, Uysal-Biyikoglu, Singh & Modiano 2018) proposed a lower bound on the average AoI for a source node sending time-sensitive information to multiple users through unreliable channels, without any resource constraint (i.e. $C_{max} = 1$). The lower bound in Theorem 1.6 shows the effect of the constraint C_{max}, and even for $C_{max} = 1$, it is tighter than the one provided by Kadota et al. (2018).

1.4.2 Practical RL Algorithms

In Section 1.3, we presented a simple *average-cost SARSA* algorithm to minimize the average AoI for a single user. Due to the large state space of the multi-user network considered in this section, alternative lower-complexity learning algorithms are proposed.

UCRL2 with HARQ

The upper confidence RL (UCRL2) algorithm (Auer, Jaksch & Ortner 2009) is a well-known RL algorithm for finite state and action MDP problems, with strong theoretical performance guarantees. However, the computational complexity of the algorithm scales quadratically with the size of the state space, which makes it unsuitable for large state spaces. UCRL2 has been initially proposed for generic MDPs with unknown rewards and transition probabilities; which need to be learned for each state-action pair. For the average AoI problem, the rewards are known (i.e., AoIs) while the transition probabilities are unknown. Moreover, the number of parameters to be learned can be reduced to the number of transmission error probabilities to each user; thus, the computational complexity can be reduced significantly.

For a generic tabular MDP, UCRL2 keeps track of the possible MDP models (transition probabilities and expected immediate rewards) in a high-probability sense and finds a policy that has the best performance in the best possible MDP. In our problem, it is enough to optimistically estimate the error probabilities $g_j(r)$, and find an optimal policy for this optimistic MDP. This is possible since the performance corresponding to a fixed sequence of transmission decisions improves if the error probabilities decrease. We will guarantee the average transmission constraint by updating the Lagrange multiplier according to the empirical resource consumption. The details are given in Algorithm 2.

UCRL2 exploits the optimistic MDP characterized by the optimistic estimation of error probabilities within a certain confidence interval, where $\hat{g}_j(r)$ and $\tilde{g}_j(r)$ represent the empirical and the optimistic estimates of the error probability for user j after r retransmissions. In each episode, we keep track of a value η resulting in a transmission cost close to C_{max}, and then find and apply a policy that is optimal for the optimistic MDP (i.e., the MDP with the smallest total cost from among all plausible ones given the observations) with Lagrangian cost. In contrast to the original UCRL2 algorithm, finding the optimistic MDP in this case is easy (choosing lower estimates of the error probabilities), and we can use standard value iteration (VI) to compute the optimal policy (instead of the more complex extended VI used in UCRL2). Thus, the computational complexity, which is the main drawback of UCRL2 algorithm, reduces significantly for the average AoI problem. UCRL2 is employed for Problem 2 in this chapter since it is an online algorithm (i.e., it does not need any previous training) and it enjoys strong theoretical guarantees for $C_{max} = 1$. The resulting algorithm will be called UCRL2-VI.

A Heuristic Version of the UCRL2 for Standard ARQ

Next, we consider the standard ARQ protocol with unknown error probabilities $p_j = g_j(0)$. The estimation procedure of UCRL2-VI can be immediately simplified, as it only needs to estimate M parameters. In order to reduce the computational complexity, we can replace the costly VI in the algorithm to find the $\tilde{\pi}_k$ with the suboptimal WI policy given in Section 1.4.1. The resulting algorithm, called *UCRL2-Whittle*, selects policy $\tilde{\pi}_k$ in step 16 following the WI policy in Section 1.4.1. The details are given in Algorithm 3, where $\hat{p}(j)$ and $\tilde{p}(j)$ denote the empirical and the optimistic estimate of the error probability for user j.

Average-Cost SARSA with LFA

In Section 1.3.3, the average-cost SARSA algorithm is employed with *Boltzmann* (*softmax*) exploration for the average AoI problem with a single user. When the state-space $\mathcal{S}$ is small and a simulator is available for the system, updates similar to Section 1.3.3 can be computed for all state-action pairs. However, this is not possible for large state

Algorithm 2 UCRL2-VI

```
Input: Confidence parameter $\rho \in (0,1)$, update parameter $\alpha$, $C_{max}$, confidence bound constant $U$, $|\mathcal{S}|$, $|\mathcal{A}|$
 1: $\eta = 0, t = 1$
 2: Observe initial state $s_1$
 3: for episodes $k = 1, 2, \dots$ do set $t_k \triangleq t$
 4:     for $j \in [M]$, $r \in [r_{max}]$ do
 5:         $N_k(j,r) \triangleq |\{\tau < t_k : a_\tau = \mathrm{x}_j, r_{j,\tau} = r\}|$,  $N_k(j,0) \triangleq |\{\tau < t_k : a_\tau = \mathrm{n}_j\}|$
 6:         $E_k(j,r) \triangleq |\{\tau < t_k : a_\tau = \mathrm{x}_j, r_{j,\tau} = r, NACK\}|$, $E_k(j,0) \triangleq |\{\tau < t_k : a_\tau = \mathrm{n}_j, NACK\}|$
 7:         $\hat{g}_j(r) \triangleq \frac{E_k(j,r)}{\max\{N_k(j,r),1\}}$
 8:     end for
 9:     $C_k \triangleq |\{\tau < t_k : a_\tau \neq \mathrm{i}\}|$
10:     $\eta \leftarrow \eta + \alpha(C_k/t_k - C_{max})$
11:     Compute optimistic error probability estimates:    $\tilde{g}_j(r) \triangleq \max\left\{0, \hat{g}_j(r) - \sqrt{\frac{U\log(|\mathcal{S}||\mathcal{A}|t_k/\rho)}{max\{1,N_k(j,r)\}}}\right\}$
12:     Use $\tilde{g}_j(r)$ and VI to find a policy $\tilde{\pi}_k$
13:     Set $v_k(j,r) \leftarrow 0, \forall j, r$
14:     while $v_k(j,r) < N_k(j,r)$ do                                   /* run policy $\tilde{\pi}_k$ */
15:         Choose action $a_t = \tilde{\pi}_k(s_t)$, and if $a_t \neq \mathrm{i}$, set $j_t$ as target user, otherwise $j_t = 0$
16:         Obtain cost $\sum_{j=1}^{M} w_j \delta_j^{rx} + \eta \mathbb{1}[a_t \neq \mathrm{i}]$ and observe $s_{t+1}$
17:         Update $v_k(j_t, r) = v_k(j_t, r) + 1$ and set $t \leftarrow t + 1$
18:     end while
19: end for
```

Algorithm 3 UCRL2 for Average AoI with ARQ

```
Input: Confidence parameter $\rho \in (0,1)$, update parameter $\alpha$, $C_{max}$, confidence bound constant $U$, $|\mathcal{S}|$, $|\mathcal{A}|$
 1: $\eta = 0, t = 1$
 2: Observe initial state $s_1$
 3: for episodes $k = 1, 2, \dots$ do and set $t_k \triangleq t$,
 4:     $N_k(j) \triangleq |\{\tau < t_k : a_\tau = \mathrm{n}_j\}|$,  $E_k(j) \triangleq |\{\tau < t_k : a_\tau = \mathrm{n}_j, NACK\}|$
 5:     $\hat{p}(j) \triangleq \frac{E_k(j)}{\max\{N_k(j),1\}}$,  $C_k \triangleq |\{\tau < t_k : a_\tau \neq \mathrm{i}\}|$,
 6:     $\eta \leftarrow \eta + \alpha(C_k/t_k - C_{max})$
 7:     Compute optimistic error probabilities:    $\tilde{p}(j) \triangleq \max\{0, \hat{p}(j) - \sqrt{\frac{U\log(|\mathcal{S}||\mathcal{A}|t_k/\rho)}{max\{1,N_k(j)\}}}\}$
 8:     Use $\tilde{p}(j)$ to find a policy $\tilde{\pi}_k$ and execute policy $\tilde{\pi}_k$
 9:     while $v_k(j) < N_k(j)$ do
10:         Choose action $a_t = \tilde{\pi}_k(s_t)$,
11:         Obtain cost $\sum_{j=1}^{M} w_j \delta_j^{rx} + \eta * \mathbb{1}[a_t \neq \mathrm{i}]$ and observe $s_{t+1}$
12:         Update $v_k(j) = v_k(j) + 1$ and set $t \leftarrow t + 1$;
13:     end while
14: end for
```

spaces, or if the Q functions are learned online: that is, to collect data about some states, the system needs to be driven to that state, which may be very costly, severely limiting the set of states for which the updates can be computed. For the problem with multiple users, the cardinality of the state-action space is large and it is difficult to even store a matrix that has the size of the state-action space. Hence, average-cost SARSA with LFA is employed, where a linear function of features can be used to approximate the Q-function in SARSA (Puterman 1994). Average-cost SARSA with LFA is an online algorithm similar to average-cost SARSA and UCRL2 algorithms. It improves the performance of average-cost SARSA by improving the convergence rate significantly for multi-user systems and its application is much simpler than the UCRL2 algorithm.

We approximate the Q function with a linear function Q_θ defined as: $Q_\theta(s,a) \triangleq$

Algorithm 4 Average-cost SARSA with LFA

```
Input: Lagrange parameter $\eta$, update parameters $\alpha, \beta, \gamma, \mathcal{A}$
 1: Set $t \leftarrow 1$ , $\theta \leftarrow 0$, $J_\eta \leftarrow 0$
 2: for $t = 1, 2, \dots$ do
 3:     Find parameterized policies with Boltzmann exploration: $\pi(a|s_t) = \frac{\exp(-\theta^T \phi(s_t,a))}{\sum_{a' \in \mathcal{A}} \exp(-\theta^T \phi(s_t,a'))}$
 4:     Sample and execute action $a_t$ from $\pi(a|s_t)$
 5:     Observe next state $s_{t+1}$ and cost $\sum_{j=1}^{M} \delta_j^{rx} + \eta * \mathbb{1}[a_t \neq \text{i}]$.
 6:     $\pi(a|s_{t+1}) = \frac{\exp(-\theta^T \phi(s_{t+1},a))}{\sum_{a'_{t+1} \in \mathcal{A}} \exp(-\theta^T \phi(s_{t+1},a'))}$
 7:     Sample $a_{t+1}$ from $\pi(a|s_{t+1})$
 8:     Compute $C_\eta$
 9:     Update linear coefficients: $\theta \leftarrow \theta + \alpha_t[\Delta_t + \eta \cdot \mathbb{1}[a_t \neq \text{i}] - J_\eta + \theta^T \phi(s_{t+1}, a_{t+1}) - \theta^T \phi(s_t, a_t)]\phi(s_t, a_t)$
10:     Update gain: $J_\eta \leftarrow J_\eta + \beta_t[\Delta_t + \eta \cdot \mathbb{1}[a_t \neq \text{i}] - J_\eta]$
11:     Update Lagrange multiplier: $\eta \leftarrow \eta + \gamma_t(C_\eta - C_{max})$
12: end for
```

$\theta^T \phi(s, a)$, where $\phi(s, a) \triangleq (\phi_1(s, a), \dots, \phi_d(s, a))^T$ is a given feature associated with the pair (s, a). In our experiments, we set $\{\phi_i(s, a)\}_{i=1}^{M}$ as the weighted age of each user $(w_j \delta_j^{rx})$ and $\{\phi_i(s, a)\}_{i=M+1}^{2M}$ as the retransmission number of each user (r_j) given an action $a \in \mathcal{A}$ is chosen in state $s \in \mathcal{S}$:

$$Q_\theta(s, a) = \theta_{(0,a)} + \theta_{(1,a)} w_1 \delta_1 + \dots + \theta_{(M,a)} w_M \delta_M + \theta_{(M+1,a)} r_1 + \dots + \theta_{(2M,a)} r_M, \quad (1.24)$$

where $\theta_{(0,a)}$ denotes the constant variable. The dimension of θ is $d = (2M + 1)|\mathcal{A}|$. The outline of the algorithm is given in Algorithm 4.

The performance of average-cost SARSA with LFA is demonstrated in Section 1.4.3. We note that linear approximators are not always effective, and the performance can be improved in general by using a non-linear approximator. However; the performance also depends on the availability of data, i.e., the linear approximator may perform better if the available data set is limited.)

Deep Q-Network (DQN)

A DQN uses a multi-layered neural network to estimate $Q(s, a)$; that is, for a given state s, DQN outputs a vector of state-action values, $Q_\theta(s, a)$, where θ denotes the parameters of the network. The neural network is a function from $2M$ inputs to $|\mathcal{A}|$ outputs which are the estimates of the Q-function $Q_\theta(s, a)$. We apply the DQN algorithm of Mnih et al. (2015) to learn a scheduling policy. We create a fairly simple feed-forward neural network of 3 layers, one of which is the hidden layer with 24 neurons. We also use *Huber loss* (Huber 1964) and the *Adam* algorithm (Kingma & Ba 2015) to conduct stochastic gradient descent to update the weights of the neural network.

We exploit two important features of DQNs as proposed by Mnih et al. (2015): *experience replay* and a *fixed target network*, both of which provide algorithm stability. For *experience replay*, instead of training the neural network with a single observation $< s, a, s', c(s, a) >$ at the end of each step, many experiences (i.e., (state, action, next state, cost) quadruplets) can be stored in the replay memory for batch training, and a minibatch of observations randomly sampled at each step can be used. The DQN uses two neural networks: a target network and an online network. The *target network*, with

Table 1.1 Hyperparameters of DQN algorithm used in the chapter

Parameter	Value	Parameter	Value	Parameter	Value
discount factor γ	0.99	optimizer	Adam	activation function	ReLU
minibatch size	32	loss function	Huber loss	hidden size	24
replay memory length	2000	exploration coefficient ϵ_0	1	episode length T	1000
learning rate α	10^{-4}	ϵ decay rate β	0.9	ϵ_{min}	0.01

parameters θ^-, is the same as the online network except that its parameters are updated with the parameters θ of the online network after every T steps, and θ^- is kept fixed in other time slots. For a minibatch of of observations for training, temporal difference estimation error e for a single observation can be calculated as

$$e = Q_\theta(s,a) - (-c(s,a) + \gamma Q_{\theta^-}(s', \arg\max Q_\theta(s',a))). \tag{1.25}$$

Huber loss is defined by the squared error term for small estimation errors, and a linear error term for high estimation errors, allowing less dramatic changes in the value functions and further improving the stability. For a given estimation error e and loss parameter d, the Huber loss function, denoted by $\mathrm{L}^d(e)$, and the average loss over the minibatch, denoted by $\mathcal{B}$, are computed as

$$\mathrm{L}^d(e) = \begin{cases} e^2 & \text{if } e \le d \\ d(|e| - \frac{1}{2}d)) & \text{if } e > d, \end{cases} \quad \text{and} \quad \mathrm{L}_\mathcal{B} = \frac{1}{|\mathcal{B}|} \sum_{<s,a,s',c(s,a)>\in\mathcal{B}} \mathrm{L}^d(e).$$

We apply the ϵ-greedy policy to balance exploration and exploitation. We let ϵ decay gradually from ϵ_0 to ϵ_{min}; in other words, the source explores more at the beginning of training, and exploits more at the end. The hyperparameters of the DQN algorithm are tuned experimentally, and are given in Table 1.1.

1.4.3 Simulation Results

In this section, we provide numerical results for the proposed learning algorithms, and compare the achieved average performances. Figure 1.5 illustrates the mean and variance of the average AoI with standard ARQ with respect to the size of the network when there is no constraint on the average number of transmissions (i.e. $C_{max} = 1$) and the performance of the UCRL2-Whittle is compared with the lower bound (UCRL2-VI is omitted since its performance is very similar to UCLR2-Whittle and has a much higher computational complexity, especially for large M). The performance of UCRL2-Whittle is close to the lower bound and is very similar to that of the WI policy, which requires a priori knowledge of the error probabilities. Moreover, UCRL2-Whittle outperforms the greedy benchmark policy which always transmits to the user with the highest age and the round robin policy, which transmits to each user in turns. We can also observe that the variances of the average AoI achieved by benchmark policies are much larger, which also limits their performance.

Figure 1.6 shows the performance of the learning algorithms for the HARQ protocol

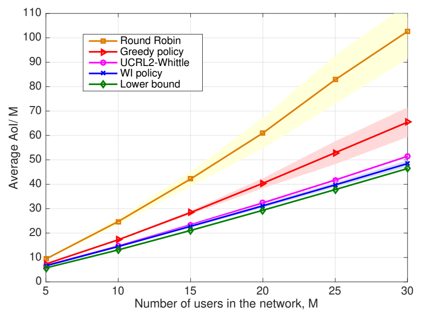

Figure 1.5 Average AoI for networks with different sizes, where $p_j = (j-1)/M$, $C_{max} = 1$, and $w_j = 1,\ \forall j$. The results are obtained after 10^4 time steps and averaged over 100 runs (both the mean and the variance are shown).

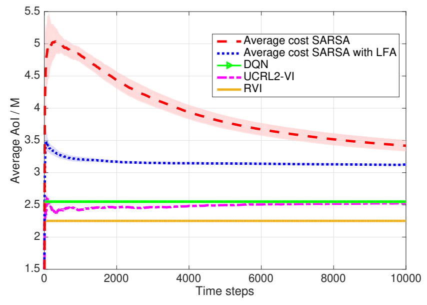

Figure 1.6 Average AoI for a 2-user HARQ network with error probabilities $g_1(r_1) = 0.5 \cdot 2^{-r_1}$ and $g_2(r_2) = 0.2 \cdot 2^{-r_2}$, where $C_{max} = 1$ and $w_j = 1,\ \forall j$. The simulation results are averaged over 100 runs (both the mean and the variance are shown).

(the mean and the variance of the average AoI) for a 2-user scenario. DQN is trained for 500 episodes with configuration in Table 1.1. It is worth noting that although UCRL2-VI converges to the optimal policy in fewer iterations than average-cost SARSA and average-cost SARSA with LFA, iterations in UCRL2-VI are computationally more demanding since the algorithm uses VI in each epoch. Therefore, UCRL2-VI is not practical for problems with large state spaces, in a case for large M. On the other hand, UCRL2-Whittle can handle a large number of users since it is based on a simple index policy instead of VI. As illustrated in Figure 1.6 that LFA significantly improves the performance of average-cost SARSA and DQN with neural network estimator, and UCRL2-Whittle improves the performance of RL even more.

We concluded that the choice of the learning algorithm to be adopted depends on the scenario and system characteristics. It has been shown that average-cost SARSA is not effective considering the large state space of the multi-user problem. Different state-of-the-art RL methods are presented including SARSA with LFA, UCRL2, and DQN. The performance of UCRL2-VI algorithm is close to optimal for small networks, i.e. consisting of 1-5 users, and enjoys theoretical guarantees. However, UCRL2-VI is not favorable for large networks due to its computational complexity, and UCRL2-Whittle is preferable. On the other hand, UCRL2-Whittle cannot be employed for a general HARQ multi-user system. Similarly, SARSA with LFA has decreased the average AoI significantly for small-size networks with HARQ; however, is not effective for large networks and SARSA with LFA lacks stability. A non-linear approximation with DQN performs for large networks; which is not fully online and requires a training time before running the algorithm.

1.5 AoI in Energy Harvesting Status-Update Systems

Many status-update systems are powered by scavenging energy from renewable sources (e.g., solar cells, wind turbines, or piezoelectric generators, etc.). Harvesting energy from ambient sources provides environmentally-friendly and ubiquitous operation for remote sensing systems. Therefore, there has been a growing interest in maximizing the timeliness of information in energy harvesting (EH) communication systems (Bacinoglu, Ceran & Uysal-Biyikoglu 2015, Yates 2015, Abd-Elmagid et al. 2020, Arafa, Yang, Ulukus & Poor 2020, Stamatakis, Pappas & Traganitis 2019). In this section, we assume that the source can sense the underlying time-varying process and generate a status update at a certain energy cost.

At the end of each time slot t, a random amount of energy is harvested and stored in a rechargeable battery at the transmitter, denoted by $E_t \in \mathcal{E} \triangleq \{0, 1, \ldots, E_{max}\}$, following a first-order discrete-time Markov model, characterized by the stationary transition probabilities $p_E(e_1|e_2)$, defined as $p_E(e_1|e_2) \triangleq Pr(E_{t+1} = e_2|E_t = e_1)$, $\forall t$ and $\forall e_1, e_2 \in \mathcal{E}$. It is also assumed that $p_E(0|e) > 0$, $\forall e \in \mathcal{E}$. Harvested energy is first stored in a rechargeable battery with a limited capacity of $B_{\max}$ energy units. The energy consumption for status sensing is denoted by $E^{\mathrm{s}} \in \mathbb{Z}^+$, while the energy consumption for a transmission attempt is denoted by $E^{\mathrm{tx}} \in \mathbb{Z}^+$.

The battery state at the beginning of time slot t, denoted by B_t, can be written as follows:

$$B_{t+1} = \min(B_t + E_t - (E^{\mathrm{s}} + E^{\mathrm{tx}})\mathbb{1}[A_t = \mathrm{n}] - E^{\mathrm{tx}}\mathbb{1}[A_t = \mathrm{x}], B_{\max}), \tag{1.26}$$

and the energy causality constraints are given as:

$$(E^{\mathrm{s}} + E^{\mathrm{tx}})\mathbb{1}[A_t = \mathrm{n}] + E^{\mathrm{tx}}\mathbb{1}[A_t = \mathrm{x}] \leq B_t, \tag{1.27}$$

where the indicator function $\mathbb{1}[C]$ is equal to 1 if event C holds, and zero otherwise. (1.26) implies that the battery overflows if energy is harvested when the battery is full, while (1.27) imposes that the energy consumed by sensing or transmission operations in time slot t is limited by the energy B_t available in the battery at the beginning of that time slot. We set $B_0 = 0$ so that the battery is empty at time $t = 0$.

Let δ_t^{tx} denote the number of time slots elapsed since the generation of the most recently sensed status update at the transmitter side, while δ_t^{rx} denote the AoI of the most recently received status update at the receiver side. δ_t^{tx} resets to 1 if a new status update is generated in time slot $t-1$, and increases by one (up to δ_{max}) otherwise, i.e.,

$$\delta_{t+1}^{tx} = \begin{cases} 1 & \text{if } A_t = \mathrm{n}; \\ \min(\delta_t^{tx} + 1, \delta_{max}) & \text{otherwise.} \end{cases}$$

On the other hand, the AoI at the receiver side evolves as follows:

$$\delta_{t+1}^{rx} = \begin{cases} \min(\delta_t^{rx}+1, \delta_{max}) & \text{if } A_t = \text{i or } K_t = 0; \\ 1 & \text{if } A_t = \text{n and } K_t = 1; \\ \min(\delta_t^{tx}+1, \delta_{max}) & \text{if } A_t = \text{x and } K_t = 1. \end{cases}$$

Note that once the AoI at the receiver is at least as large as at the transmitter, this relationship holds forever; thus it is enough to consider cases when $\delta_t^{rx} \geq \delta_t^{tx}$.

To determine the success probability of a transmission, we need to keep track of the number of current retransmissions. The number of retransmissions is zero for a newly sensed and generated status update and increases up to r_{max} as we keep retransmitting the same packet. It is easy to see that retransmitting when $\delta_{t+1}^{tx} = \delta_{max}$ is suboptimal, therefore we explicitly exclude this action by setting the retransmission count to 0 in this case. Also, it is suboptimal to generate a new update and retransmit the old one, thus whenever a new status update is generated, the previous update at the transmitter is dropped and cannot be retransmitted. Thus, the evolution of the retransmission count is given as follows:

$$R_{t+1} = \begin{cases} 0 & \text{if } K_t = 1 \text{ or } \delta_{t+1}^{tx} = \delta_{max}; \\ 1 & \text{if } A_t = \text{n and } K_t = 0; \\ r_t & \text{if } A_t = \text{i and } \delta_{t+1}^{tx} \neq \delta_{max}; \\ \min(r_t+1, r_{max}) & \text{if } A_t = \text{x}, K_t = 0 \text{ and } \delta_{t+1}^{tx} \neq \delta_{max}. \end{cases}$$

The state of the system is formed by five components $S_t = (E_t, B_t, \delta_t^{rx}, \delta_t^{tx}, r_t)$. In each time slot, the transmitter knows the state, and takes action from the set $\mathcal{A} = \{\text{i, n, x}\}$. The goal is to find a policy π which minimizes the expected average AoI at the receiver over an infinite time horizon, which is given by:

Problem 3

$$J^* \triangleq \min_{\pi} \lim_{T\to\infty} \frac{1}{T+1}\mathbb{E}\left[\sum_{t=0}^{T} \delta_t^{rx}\right] \quad \text{subject to (1.26) and (1.27).}$$

In Section 1.3.3, we have considered status updates with HARQ under an average power constraint. In that case, it is possible to show that it is suboptimal to retransmit a failed update after an idle period. Restricting the actions of the transmitter accordingly, the AoI at the receiver after a successful transmission event is equal to the number of retransmissions of the corresponding update. Therefore in addition to the AoI at the receiver, we only need to track the retransmission count. However, in the current scenario, retransmissions of a status update can be interrupted due to energy outages, which means that we also need to keep track of the AoI at the transmitter side (hence we need to have both δ_t^{rx} and δ_t^{tx} in the state of the system).

Problem 3 can be formulated as an average-cost finite-state MDP: The finite set of states $\mathcal{S}$ is defined as $\mathcal{S} = \{s = (e, b, \delta^{rx}, \delta^{tx}, r) : e \in \mathcal{E}, b \in \{0, \ldots, B_{\max}\}, \delta^{rx}, \delta^{tx} \in \{1, \ldots, \delta_{max}\}, r \in \{0, \ldots, r_{max}\}, \delta^{rx} \geq \delta^{tx}\}$, while the finite set of actions $\mathcal{A} = \{\text{i, n, x}\}$ is already defined. Note that action x cannot be taken in states with retransmission count

$r = 0$. P is characterized by the EH statistics and channel error probabilities. The cost function $c : \mathcal{S} \times \mathcal{A} \to \mathbb{Z}$ is the AoI at the receiver, and is defined as $c(s, a) = \delta^{rx}$ for any $s \in \mathcal{S}$, $a \in \mathcal{A}$, independent of the action taken, where δ^{rx} is the component of s describing the AoI at the receiver.

It is easy to see that MDP formulated for Problem 3 is a communicating MDP by Proposition 8.3.1 of Puterman (1994)[6], that is, for every pair of $(s, s') \in \mathcal{S}$, there exists a deterministic policy under which s' is accessible from s. By Theorem 8.3.2 of Puterman (1994), an optimal stationary policy exists with constant gain. In particular, there exists a function $h : \mathcal{S} \to \mathbb{R}$, called the *differential cost function* for all $s = (e, b, \delta^{rx}, \delta^{tx}, r) \in \mathcal{S}$, satisfying the following *Bellman optimality equations* for the average-cost finite-state finite-action MDP (Puterman 1994):

$$h(s) + J^* = \min_{a \in \{\text{i,n,x}\}} (\delta^{rx} + \mathbb{E}\left[h(s')|a\right]), \tag{1.28}$$

where $s' \triangleq (e', b', \delta^{rx\prime}, \delta^{tx\prime}, r')$ is the next state obtained from $(e, b, \delta^{rx}, \delta^{tx}, r)$ after taking action a, and J^* represents the optimal achievable average AoI under policy π^*.

We also introduce the *state-action cost function*:

$$Q((e, b, \delta^{rx}, \delta^{tx}, r), a) \triangleq \delta^{rx} + \mathbb{E}\left[h(e', b', \delta^{rx\prime}, \delta^{tx\prime}, r')|a\right]. \tag{1.29}$$

Then an optimal policy, for any $(e, b, \delta^{rx}, \delta^{tx}, r) \in \mathcal{S}$, takes the action achieving the minimum in (1.29):

$$\pi^*(e, b, \delta^{rx}, \delta^{tx}, r) \in \arg\min_{a \in \{\text{i,n,x}\}} (Q((e, b, \delta^{rx}, \delta^{tx}, r), a)). \tag{1.30}$$

An optimal policy solving (1.28), (1.29) and (1.30) defined above can be found by RVI for finite-state finite-action average-cost MDPs from Sections 8.5.5 and 9.5.3 of Puterman (1994): Starting with an arbitrary initialization of $h_0(s)$, $\forall s \in \mathcal{S}$, and setting an arbitrary but fixed reference state $s^{ref} \triangleq (e^{ref}, b^{ref}, \delta^{rxref}, \delta^{txref}, r^{ref})$, a single iteration of the RVI algorithm $\forall (s, a) \in \mathcal{S} \times \mathcal{A}$ is given as follows:

$$Q_{n+1}(s, a) \leftarrow \delta_n^{rx} + \mathbb{E}\left[h_n(s')\right], \tag{1.31}$$

$$V_{n+1}(s) \leftarrow \min_a (Q_{n+1}(s, a)), \tag{1.32}$$

$$h_{n+1}(s) \leftarrow V_{n+1}(s) - V_{n+1}(s^{ref}), \tag{1.33}$$

where $Q_n(s, a)$, $V_n(s)$ and $h_n(s)$ denote the state-action value function, value function and differential value function at iteration n, respectively. By Theorem 8.5.7 of Puterman (1994), h_n converges to h, and $\pi_n^*(s) \triangleq \arg\min_a Q_n(s, a)$ converges to $\pi^*(s)$.

1.5.1 Structure of the Optimal Policy

Next, we investigate the structure of the optimal policy and show that the solution to Problem 3 is of threshold-type. The proof can be obtained by following the steps in Section 6.11 of Puterman (1994).

[6] By Proposition 8.3.1 of Puterman (1994), the MDP is communicating since there exists a stationary policy which induces a recurrent Markov chain, e.g., a state $(0, B_0, \delta_{max}, \delta_{max}, R_0)$ is reachable from all other states considering a policy which never transmits and in a scenario where no energy is harvested.

THEOREM 1.7 *There exists an optimal stationary policy $\pi^*(s)$ that is monotone with respect to δ_t^{rx}, that is, $\pi^*(s)$ is of threshold-type.*

Following Theorem 1.7, the optimal policy can be formally expressed as

$$A_t = \begin{cases} \mathrm{i} & \text{if } \delta_t^{rx} < \mathcal{T}_\mathrm{n}(e, b, \delta^{tx}, r), \\ \mathrm{n} & \text{if } \mathcal{T}_\mathrm{n}(e, b, \delta^{tx}, r) \le \delta_t^{rx} < \mathcal{T}_\mathrm{x}(e, b, \delta^{tx}, r), \\ \mathrm{x} & \text{if } \delta_t^{rx} \ge \mathcal{T}_\mathrm{x}(e, b, \delta^{tx}, r), \end{cases} \tag{1.34}$$

for some threshold values denoted by $\mathcal{T}_\mathrm{n}(e, b, \delta^{tx}, r)$ and $\mathcal{T}_\mathrm{x}(e, b, \delta^{tx}, r)$, where $\mathcal{T}_\mathrm{n}(e, b, \delta^{tx}, r) \le \mathcal{T}_\mathrm{x}(e, b, \delta^{tx}, r)$. We will refer to such a policy as a *double-threshold policy* in the rest of the section.

We can simplify the problem by making an assumption on the policy space in order to obtain a simpler *single-threshold policy*, which will result in a more efficient learning algorithm: We assume that a packet is retransmitted until it is successfully decoded, provided that there is enough energy in the battery, that is, the transmitter is not allowed to preempt an undecoded packet and transmit a new one.

The solution to the simplified problem is also of threshold-type, that is,

$$A_t = \begin{cases} \mathrm{i} & \text{if } \delta_t^{rx} < \mathcal{T}(e, b, \delta^{tx}, r), \\ \mathrm{n} & \text{if } \delta_t^{rx} \ge \mathcal{T}(e, b, \delta^{tx}, r), \text{ and } r = 0 \\ \mathrm{x} & \text{if } \delta_t^{rx} \ge \mathcal{T}(e, b, \delta^{tx}, r) \text{ and } r \ne 0, \end{cases} \tag{1.35}$$

for some $\mathcal{T}(e, b, \delta^{tx}, r)$.

1.5.2 Practical RL Algorithms

In some scenarios, it can be assumed that the channel and energy arrival statistics remain the same or change very slowly and the same environment is experienced for a sufficiently long time before the time of deployment. Accordingly, we can assume that the statistics regarding the error probabilities and energy arrivals are available before the time of transmission. In such scenarios, RVI algorithm (Puterman 1994) can be used. However, in most practical scenarios, channel error probabilities for retransmissions and the EH characteristics are not known at the time of deployment, or may change over time. In this section, we assume that the transmitter does not know the system characteristics *a-priori*, and has to learn them. Ceran, Gündüz & György (2018) and Ceran, Gündüz & György (2019) employed learning algorithms for constrained problems with countably infinite state spaces such as average-cost SARSA. While these algorithms can be adopted to the current framework by considering an average transmission constraint of 1, they do not have convergence guarantees. However, Problem 3 constitutes an unconstrained MDP with finite state and action spaces, and there exist RL algorithms for unconstrained MDPs which enjoy convergence guarantees. Moreover, we have shown the optimality of a threshold type policy for Problem 3, and RL algorithms which exploit this structure can be considered. Thus, we employ three different RL algorithms, and compare their performances in terms of the average AoI as well as the convergence speed. First, we employ a value-based RL algorithm, namely GR-learning (Gosavi 2004), which converges to an optimal policy. Next, we consider a structured policy search algorithm, namely FDPG, which does not necessarily find the optimal policy but performs very well in practice, as demonstrated through simulations in Section 1.5.3. We also note that GR-learning learns from a single trajectory generated during learning steps while FDPG uses Monte-Carlo roll-outs for each policy update. Thus, GR-learning is more amendable for applications in real-time systems. Finally, we employ the DQN algorithm, which implements a non-linear neural network estimator in order to learn the optimal status update policy.

GR-Learning with Softmax

For Problem 3, we employ a modified version of the *GR-learning* algorithm proposed by Gosavi (2004) with *Boltzmann* (*softmax*) exploration. The resulting algorithm is called *GR-learning with softmax* and can find the optimal policy π^* using (1.30) without knowing P, characterized by $g(r)$ and p_E. Notice that, similar to *average-cost SARSA with softmax* in Section 1.3.3, *GR-learning with softmax* starts with an initial estimate of $Q_0(s, a)$ and finds the optimal policy by estimating state-action values in a recursive manner. Update rules for $Q_n(S_n, A_n)$ and J_n in Section 1.3.3 are modified as:

$$Q_{n+1}(S_n, A_n) \leftarrow Q_n(S_n, A_n) + \alpha(m(S_n, A_n, n))[\delta_n^{rx} - J_n + Q_n(S_{n+1}, A_{n+1}) - Q_n(S_n, A_n)], \tag{1.36}$$

where $\alpha(m(S_n, A_n, n))$ is the update parameter (learning rate) in the n^{th} iteration, and depends on the function $m(S_n, A_n, n)$, which is the number of times the state–action pair (S_n, A_n) has been visited till the n^{th} iteration.

$$J_{n+1} \leftarrow J_n + \beta(n)\left[\frac{nJ_n + \delta_n^{rx}}{n+1} - J_n\right] \tag{1.37}$$

where $\beta(n)$ is the update parameter in the n^{th} iteration.

The transmitter action selection method should balance the *exploration* of new actions with the *exploitation* of actions known to perform well. We use the *Boltzmann* (*softmax*) action selection method, which chooses each action randomly relative to its expected cost as defined in Section 1.3.3:

According to Theorem 2 of Gosavi (2004), if α, β satisfy $\sum_{m=1}^{\infty} \alpha(m), \sum_{m=1}^{\infty} \beta(m) \to \infty$, $\sum_{m=1}^{\infty} \alpha^2(m), \sum_{m=1}^{\infty} \beta^2(m) < \infty$, $\lim_{x\to\infty} \frac{\beta(m)}{\alpha(m)} \to 0$, *GR*-Learning converges to an optimal policy.

Finite-Difference Policy Gradient (FDPG)

GR-learning in Section 1.5.2 is a value-based RL method, which learns the state-action value function for each state-action pair. In practice, δ_{max} can be large, which might slow down the convergence of GR-learning due to a prohibitively large state space.

In this section, we are going to propose a learning algorithm which exploits the structure of the optimal policy exposed in Theorem 1.7. A monotone policy is shown to be average optimal in the previous section; however, it is not possible to compute the threshold values directly for Problem 3.

Note that, $A_t = \mathrm{i}$ if $B_t < E^{\mathrm{tx}}$ ($B_t < E^{\mathrm{tx}} + E^{\mathrm{s}}$) for $r \geq 1$ ($r = 0$); that is, $\mathcal{T}(e, b, \delta^{tx}, r) = \delta_{max} + 1$ if $b < E^{\mathrm{tx}}$ for $r \geq 1$ and $b < E^{\mathrm{tx}} + E^{\mathrm{s}}$ for $r = 0$. This ensures that energy causality constraints in (1.27) hold. Other thresholds will be determined using PG. In order to employ the PG method, we approximate the policy by a parameterized smooth function with parameters $\theta(e, b, \delta^{tx}, r)$, and convert the discrete policy search problem into estimating the optimal values of these continuous parameters, which can be numerically solved by stochastic approximation algorithms (Spall 2003). Continuous stochastic approximation is much more efficient than discrete search algorithms in general.

In particular, with a slight abuse of notation, we let $\pi_\theta(e, b, \delta^{rx}, \delta^{tx}, r)$ denote the probability of taking action $A_t = \mathrm{n}$ ($A_t = \mathrm{x}$) if $r = 0$ ($r \neq 0$), and consider the parameterized sigmoid function:

$$\pi_\theta(e, b, \delta^{rx}, \delta^{tx}, r) \triangleq \frac{1}{1 + e^{-\frac{\delta^{rx} - \theta(e,b,\delta^{tx},r)}{\tau}}}. \tag{1.38}$$

We note that $\pi_\theta(e, b, \delta^{rx}, \delta^{tx}, r) \to \{0, 1\}$ and $\theta(e, b, \delta^{tx}, r) \to \mathcal{T}(e, b, \delta^{tx}, r)$ as $\tau \to 0$. Therefore, in order to converge to a deterministic policy π, $\tau > 0$ can be taken as a sufficiently small constant, or can be decreased gradually to zero. The total number of parameters to be estimated is $|\mathcal{E}| \times B_{\max} \times \delta_{max} \times r_{max} + 1$ minus the parameters corresponding to $b < E^{\mathrm{tx}}$ ($b < E^{\mathrm{tx}} + E^{\mathrm{s}}$) for $r > 0$ ($r = 0$) due to energy causality constraints as stated previously.

With a slight abuse of notation, we map the parameters $\theta(e, b, \delta^{tx}, r)$ to a vector $\overline{\theta}$ of size $d \triangleq |\mathcal{E}| \times B_{\max} \times \delta_{max} \times r_{max} + 1$. Starting with some initial estimates of $\overline{\theta}_0$, the parameters can be updated in each iteration n using the gradients as follows:

$$\overline{\theta}_{n+1} = \overline{\theta}_n - \gamma(n)\, \partial J / \partial \overline{\theta}_n, \tag{1.39}$$

where the step size parameter $\gamma(n)$ is a positive decreasing sequence and satisfies the first two convergence properties given at the end of Section 1.5.2 from the theory of stochastic approximation (Kushner & Yin 1997).

Computing the gradient of the average AoI directly is not possible; however, several methods exist in the literature to estimate the gradient (Spall 2003). In particular, we employ the FDPG (Peters & Schaal 2006) method. In this method, the gradient is estimated by estimating J at slightly perturbed parameter values. First, a random perturbation vector D_n of size d is generated according to a predefined probability distribution, e.g., each component of D_n is an independent Bernoulli random variable with parameter $q \in (0, 1)$. The thresholds are perturbed with a small amount $\sigma > 0$ in the directions defined by D_n to obtain $\overline{\theta}_n^{\pm}(e, b, \delta^{tx}, r) \triangleq \overline{\theta}_n(e, b, \delta^{tx}, r) \pm \sigma D_n$. Then, empirical estimates $\hat{J}^{\pm}$ of the average AoI corresponding to the perturbed parameters $\overline{\theta}_n^{\pm}$, obtained from Monte-Carlo rollouts, are used to estimate the gradient:

$$\partial J / \partial \overline{\theta}_n \approx (D_n^{\mathsf{T}} D_n)^{-1} D_n^{\mathsf{T}} \frac{(\hat{J}^{+} - \hat{J}^{-})}{2\sigma}, \tag{1.40}$$

where D_n^{T} denotes the transpose of vector D_n. The pseudo code of the finite difference policy gradient algorithm is given in Algorithm 5.

Similar steps can be followed to find the thresholds for the double-threshold policy

Algorithm 5 FDPG

```
 1: $\tau_0 \leftarrow 0.3$,                                   /* temperature parameter */
 2: $\zeta \leftarrow 0.99$,                                  /* decaying coefficient for $\tau$ */.
 3: $\overline{\theta}_0 \leftarrow 0$                        /* initialization of $\overline{\theta}$ */
 4: for $n = \{1, 2, \ldots\}$ do
 5:     GENERATE random perturbation vector
          $D_n = binomial(\{0, 1\}, q = 0.5, d)$
 6:     PERTURB parameters $\overline{\theta}_n$
          $\overline{\theta}_n^+ = \overline{\theta}_n + \beta D_n, \overline{\theta}_n^- = \overline{\theta}_n - \beta D_n$
 7:     ESTIMATE $\hat{J}_n^\pm$ from Monte-Carlo rollouts using policies $\pi_{\theta_n^\pm}$:
 8:     for $t \in \{1, \ldots, T\}$ do
 9:         OBSERVE current state $S_t$ and USE policy $\pi_{\theta_n^\pm}$
10:     end for
11:     ESTIMATE $\hat{J}_n^\pm$ from Monte-Carlo rollouts using policy $\pi_{\theta_n^\pm}$
          $\hat{J}_n^\pm = \frac{1}{T+1} \sum_{t=0}^{T} \delta_t^{rx}$
12:     COMPUTE estimate of the gradient $\partial J / \partial \overline{\theta}_n$
13:     UPDATE
          $\overline{\theta}_{n+1} = \overline{\theta}_n - \gamma(n)\, \partial J / \partial \overline{\theta}_n$
          $\tau_{n+1} \leftarrow \zeta \tau_n$                /* decrease $\tau$ */
14: end for
```

where $\mathcal{T}(e, b, \delta^{tx}, r)$ and $\theta(e, b, \delta^{tx}, r)$ are replaced by $\mathcal{T}_{\mathrm{n}}(e, b, \delta^{tx}, r)$, $\mathcal{T}_{\mathrm{x}}(e, b, \delta^{tx}, r)$ and $\theta_{\mathrm{n}}(e, b, \delta^{tx}, r)$, $\theta_{\mathrm{x}}(e, b, \delta^{tx}, r)$ respectively.

1.5.3 Simulation Results

In this section, we provide numerical results for all the proposed algorithms, and compare the achieved average AoI. In the experiments, the maximum number of retransmissions is set to $r_{max} = 3$, while λ and p_0 are set to 0.5. E^{tx} and E^s are both assumed to be constant and equal to 1 unit of energy unless otherwise stated. δ_{max} is set to 40.

We choose the exact step sizes for the learning algorithms by fine-tuning in order to balance the algorithm stability in the early time steps with nonnegligible step sizes in the later time steps. In particular, we use step size parameters of $\alpha(m)$,$\beta(m)$, $\gamma(m) = y/(m + 1)^z$, where $0.5 < z \leq 1$ and $y > 0$ (which satisfy the convergence conditions), and choose y and z such that the oscillations are low and the convergence rate is high. We have observed that a particular choice of parameters gives similar performance results for scenarios addressed in simulations results. DQN algorithm in this section is configured as in Table 1.1 and trained for 500 episodes. The average AoI for DQN is obtained after 10^5 time steps and averaged over 100 runs.

As a baseline, we have also included the performance of a greedy policy, which senses and transmits a new status update whenever there is sufficient energy. It retransmits the last transmitted status update when the energy in the battery is sufficient only for transmission, and it remains idle otherwise.

Next, we investigate the performance when the EH process has temporal correlations. A symmetric two-state Markovian EH process is assumed, such that $\mathcal{E} = \{0, 1\}$ and $Pr(E_{t+1} = 1|E_t = 0) = Pr(E_{t+1} = 0|E_t = 1) = 0.3$. That is, if the transmitter is in harvesting state, it is more likely to continue harvesting energy, and vice versa.

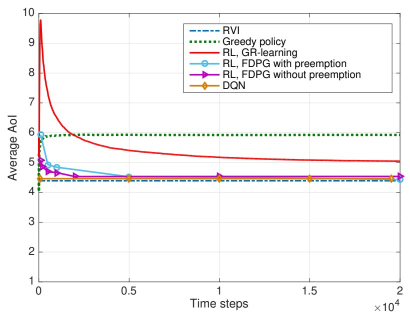

Figure 1.7 The performance of RL algorithms when $B_{\max} = 5$, $p_E(1, 1)$, $p_E(0, 0) = 0.7$ and $E^{\mathrm{s}}, E^{\mathrm{tx}} = 1$. FDPG with and without preemption represent the double-threshold and the single-threshold policies, respectively.

Figure 1.7 shows the evolution of the average AoI over time when the average-cost RL algorithms are employed in this scenario. It can be observed again that the average AoI achieved by the FDPG method in Section 1.5.2 performs very close to the one obtained by the RVI algorithm, which has *a priori* knowledge of $g(r)$ and p_e. GR-learning, on the other hand, outperforms the greedy policy but converges to the optimal policy much more slowly, and the gap between the two RL algorithms is larger compared to the i.i.d. case. Tabular methods in RL, like GR-learning, need to visit each state-action pair infinitely often for RL to converge (Sutton & Barto 1998). GR-learning in the case of temporally correlated EH does not perform as well as in the i.i.d. case since the state space becomes larger with the addition of the EH state. We also observe that the gap between the final performances of single- and double-threshold FDPG solutions is larger compared to the memoryless EH scenario, while the single threshold solutions till converges faster. DQN algorithm performs better than GR-learning but it requires a training time before running the simulation and does not have convergence guarantees. Moreover, it still falls short of the final performance of double-threshold FDPG.

Next, we investigate the impact of the burstiness of the EH process, measured by the correlation coefficient between E_t and E_{t+1}. Figure 1.8 illustrates the performance of the proposed RL algorithms for different correlation coefficients, which can be computed easily for the 2-state symmetric Markov chain; that is, $\rho \triangleq (2p_E(1, 1) - 1)$. Note that $\rho = 0$ corresponds to memoryless EH with $p_e = 1/2$. We note that the average AoI is minimized by transmitting new packets successfully at regular intervals, which has been well investigated in previous works (Bacinoglu et al. 2015, Ceran, Gündüz & György 2018, Yates 2015). Intuitively, for highly correlated EH, there are either successive transmissions or successive idle time slots, which increases the average AoI. Hence, the AoI is higher for higher values of ρ. Figure 1.8 also shows that both RL algorithms result in much lower average AoI than the greedy policy and FDPG outperforms GR-learning since it benefits from the structural characteristics of a threshold policy. We can also conclude that the single threshold policy can be preferable in practice especially in highly dynamic environments, as its performance is very close to that of the double threshold FDPG, but with faster convergence.

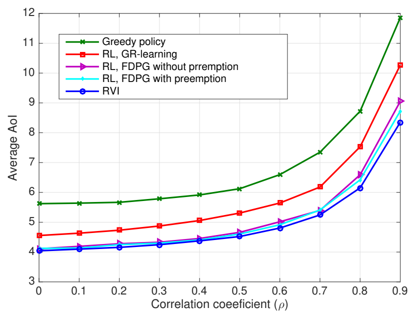

Figure 1.8 The performance of RL algorithms obtained after $2 \cdot 10^4$ time steps and averaged over 1000 runs for different temporal correlation coefficients.

1.6 Conclusions

This chapter investigated communication systems transmitting time-sensitive data over imperfect channels with the average AoI as the performance measure, which quantifies the timeliness of the data available at the receiver. Considering both the classical ARQ and the HARQ protocols, preemptive scheduling policies have been proposed by taking into account retransmissions under a resource constraint. Scenarios in which the system characteristics are not known a priori, and must be learned in an online fashion are also considered. RL algorithms are employed to balance exploitation and exploration in an unknown environment, so that the source node can quickly learn the environment based on the ACK/NACK feedback, and can adapt its scheduling policy accordingly, exploiting its limited resources in an efficient manner. The proposed algorithms and the established results in this chapter are also relevant to different systems concerning the timeliness of information, in addition to systems under average resource constraint or energy replenishment constraints. The proposed methodology and the results regarding the structure of the optimal policies can be used in other wireless communication problems.

References

802.16e 2005, I. S. (2006), ‘IEEE standard for local and metropolitan area networks part 16: Air interface for fixed and mobile broadband wireless access systems amendment 2 (incorporated into IEEE standard 802.16e-2005 and std 802.16-2004/cor1-2005)’.

Abd-Elmagid, M. A., Ferdowsi, A., Dhillon, H. S. & Saad, W. (2019), Deep reinforcement learning for minimizing age-of-information in UAV-assisted networks, *in* ‘2019 IEEE Global Communications Conference (GLOBECOM)’, pp. 1–6.

Abd-Elmagid, M., Dhillon, H. & Pappas, N. (2020), ‘A reinforcement learning framework for optimizing age-of-information in rf-powered communication systems’, *IEEE Transactions on Communications* .

Alliance, Z. (2008), ‘ZigBee specification (document 053474r17)’, *Luettu* **21**.

Altman, E. (1999), *Constrained Markov Decision Processes*, Stochastic modeling, Chapman & Hall/CRC.

Arafa, A., Yang, J., Ulukus, S. & Poor, H. V. (2020), ‘Age-minimal transmission for energy harvesting sensors with finite batteries: Online policies’, *IEEE Transactions on Information Theory* **66**(1), 534–556.

Auer, P., Jaksch, T. & Ortner, R. (2009), Near-optimal regret bounds for reinforcement learning, *in* ‘Advances in Neural Information Processing Systems 21’, Curran Associates, Inc., pp. 89–96.

Bacinoglu, B. T., Ceran, E. T. & Uysal-Biyikoglu, E. (2015), Age of information under energy replenishment constraints, *in* ‘Inf. Theory and Applications Workshop (ITA)’, pp. 25–31.

Bertsekas, D. P. (2000), *Dynamic Programming and Optimal Control*, 2nd edn, Athena Scientific.

Beutler, F. J. & Ross, K. W. (1985), ‘Optimal policies for controlled Markov chains with a constraint’, *Journal of Mathematical Analysis and Applications* **112**(1), 236 – 252.

Beytur, H. B. & Uysal, E. (2019), Age minimization of multiple flows using reinforcement learning, *in* ‘International Conference on Computing, Networking and Communications (ICNC)’, pp. 339–343.

Ceran, E. T., Gündüz, D. & György, A. (2018), Reinforcement learning approach to age of information in multi-user networks, *in* ‘IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)’.

Ceran, E. T., Gündüz, D. & György, A. (2019), Reinforcement learning to minimize age of information with an energy harvesting sensor with HARQ and sensing cost, *in* ‘IEEE Conf. on Computer Comms. Workshops (INFOCOM WKSHPS)’.

Ceran, E. T., Gündüz, D. & György, A. (2020), ‘A reinforcement learning approach to age of information in multi-user networks with HARQ’, *IEEE Journal on Selected Areas in Communications* . (in press).

Ceran, E. T., Gündüz, D. & György, A. (2018), Average age of information with hybrid arq under a resource constraint, *in* 'IEEE Wireless Communications and Networking Conference (WCNC)', pp. 1–6.

Ceran, E. T., Gündüz, D. & György, A. (2019), 'Average age of information with hybrid ARQ under a resource constraint', *IEEE Transactions on Wireless Communications* **18**, 1900–1913.

Clancy, C., Hecker, J., Stuntebeck, E. & O'Shea, T. (2007), 'Applications of machine learning to cognitive radio networks', *IEEE Wireless Communications* **14**(4), 47–52.

E-UTRA (2013), 'LTE PHY; General Description,', *3GPP TS 36.201* **21**.

Elgabli, A., Khan, H., Krouka, M. & Bennis, M. (2019), Reinforcement learning based scheduling algorithm for optimizing age of information in ultra reliable low latency networks, *in* 'IEEE Symposium on Computers and Communications (ISCC)', pp. 1–6.

Gittins, J., Glazebrook, K. D. & Weber, R. (2011), *Multi-Armed Bandit Allocation Indices*, Wiley-Blackwell, London.

Gosavi, A. (2004), 'Reinforcement learning for long-run average cost', *European Journal of Operational Research* **155**, 654 – 674.

Gündüz, D., Stamatiou, K., Michelusi, N. & Zorzi, M. (2014), 'Designing intelligent energy harvesting communication systems', *IEEE Communications Magazine* **52**, 210–216.

Hatami, M., Jahandideh, M., Leinonen, M. & Codreanu, M. (2020), Age-aware status update control for energy harvesting iot sensors via reinforcement learning, *in* '2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications', pp. 1–6.

Hsu, Y. P., Modiano, E. & Duan, L. (2017), Age of information: Design and analysis of optimal scheduling algorithms, *in* 'IEEE International Symposium on Information Theory (ISIT)', pp. 561–565.

Huber, P. J. (1964), 'Robust estimation of a location parameter', *The Annals of Math. Statistics* **35**, 73–101.

Kadota, I., Uysal-Biyikoglu, E., Singh, R. & Modiano, E. (2018), 'Scheduling policies for minimizing age of information in broadcast wireless networks', *IEEE/ACM Transactions on Networking* **26**, 2637–2650.

Kingma, D. P. & Ba, J. (2015), Adam: A method for stochastic optimization, *in* '3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings'.

Kushner, H. J. & Yin, G. G. (1997), *Stochastic Approximation Algorithms and Applications*, New York: Springer-Verlag, Orlando, FL, USA.

Leng, S. & Yener, A. (2019), Age of information minimization for wireless ad hoc networks: A deep reinforcement learning approach, *in* '2019 IEEE Global Communications Conference (GLOBECOM)', pp. 1–6.

Luong, N. C., Hoang, D. T., Gong, S., Niyato, D., Wang, P., Liang, Y. & Kim, D. I. (2019), 'Applications of deep reinforcement learning in communications and networking: A survey', *IEEE Communications Surveys Tutorials* **21**(4), 3133–3174.

Mahadevan, S. (1996), 'Average reward reinforcement learning: Foundations, algorithms, and empirical results', *Machine Learning* **22**(1), 159–195.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M. A., Fidjeland, A., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S. & Hassabis, D. (2015), 'Human-level control through deep reinforcement learning', *Nature* **518**(7540), 529–533.

Oppermann, Hamalainen, M. & Iinatti, J. (2004), 'UWB theory and applications', *Wiley* .

Peters, J. & Schaal, S. (2006), Policy gradient methods for robotics, *in* '2006 IEEE/RSJ International Conference on Intelligent Robots and Systems', pp. 2219–2225.

Puterman, M. L. (1994), *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, USA.

Sennott, L. I. (1989), 'Average cost optimal stationary policies in infinite state Markov decision processes with unbounded costs', *Operations Research* **37**(4), 626–633.

Sennott, L. I. (1993), 'Constrained average cost Markov decision chains', *Prob. in Eng. and Inf. Sci.* **7**, 69–83.

Sert, E., Sönmez, C., Baghaee, S. & Uysal-Biyikoglu, E. (2018), Optimizing age of information on real-life TCP/IP connections through reinforcement learning, *in* '2018 26th Signal Processing and Communications Applications Conference (SIU)', pp. 1–4.

Silver et al., D. (2016), 'Mastering the game of Go with deep neural networks and tree search', *Nature* **529**(7587), 484–489.

Somuyiwa, S. O., György, A. & Gündüz, D. (2018), 'A reinforcement-learning approach to proactive caching in wireless networks', *IEEE Journal on Selected Areas in Communications* **36**(6), 1331–1344.

Spall, J. C. (2003), *Introduction to Stochastic Search and Optimization*, John Wiley & Sons, Inc., Hoboken, NJ, USA.

Stamatakis, G., Pappas, N. & Traganitis, A. (2019), Control of status updates for energy harvesting devices that monitor processes with alarms, *in* '2019 IEEE Globecom Workshops (GC Wkshps)', pp. 1–6.

Sutton, R. S. & Barto, A. G. (1998), *Introduction to Reinforcement Learning*, 1st edn, MIT Press, Cambridge, MA, USA.

Whittle, P. (1988), 'Restless bandits: activity allocation in a changing world', *Journal of App. Prob.* **25**, 287–298.

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