

Status Updating with an Energy Harvesting Sensor under Partial Battery Knowledge

Mohammad Hatami*, Markus Leinonen*, and Marian Codreanu†

Abstract—We consider status updating under inexact knowledge of the battery level of an energy harvesting (EH) sensor that sends status updates about a random process to users via a cache-enabled edge node. More precisely, the control decisions are performed by relying only on the battery level knowledge captured from the last received status update packet. Upon receiving on-demand requests for fresh information from the users, the edge node uses the available information to decide whether to command the sensor to send a status update or to retrieve the most recently received measurement from the cache. We seek for the best actions of the edge node to minimize the average AoI of the served measurements, i.e., *average on-demand AoI*. Accounting for the partial battery knowledge, we model the problem as a partially observable Markov decision process (POMDP), and, through characterizing its key structures, develop a dynamic programming algorithm to obtain an optimal policy. Simulation results illustrate the threshold-based structure of an optimal policy and show the gains obtained by the proposed optimal POMDP-based policy compared to a request-aware greedy (myopic) policy.

I. INTRODUCTION

In future Internet of things (IoT) systems in 5G and 6G wireless generations, timely delivery of status updates about a remotely monitored random process to a destination is the key enabler for the emerging time-critical applications, e.g., drone control and smart home systems. Such destination-centric information freshness is quantified by the *age of information* (AoI) [2], [3]. IoT networks with low-power sensors are subject to stringent energy limitations, which is often counteracted by *energy harvesting* (EH) technology. Thus, there is a need for designing *AoI-aware status updating* procedures that provide the end users with timely status of remotely observed processes while account for the limited energy resources of EH sensors.

We consider a status update system consisting of an EH sensor, an edge node, and users. The users are interested in time-sensitive information about a random process measured by the sensor. The users send requests to the edge node which has a cache storage to store the most recently received measurements from the sensor. To serve a user's request, the edge node either commands the sensor to send a fresh measurement, i.e., a status update packet, or uses the aged data in the cache. In contrast to the existing works (e.g., [4]–[7]), we consider a practical scenario where the edge node is informed of the sensor's battery level only via received status update packets, leading to *partial* battery knowledge. Particularly, our objective is to

find the best actions of the edge node to minimize the average AoI of the served measurements, i.e., *average on-demand AoI*. Accounting for the partial battery knowledge, we model this as a partially observable Markov decision process (POMDP) problem. We convert the POMDP into a belief-state MDP and, via characterizing its key structures, develop an iterative algorithm to obtain an optimal policy. Numerical experiments illustrate the threshold-based structure of an optimal policy and show the superiority of the proposed optimal POMDP-based policy compared to a request-aware greedy policy.

Only a few works have applied POMDP formulation in AoI-aware design [8]–[11]. In [8], the authors proposed POMDP-based AoI-optimal transmission scheduling in a status update system under an average energy constraint and uncertain channel state information. In [9], the authors proposed a POMDP-based AoI-optimal scheduling policy for a multiuser uplink system under partial knowledge of the status update arrivals at the monitor node. In [10], the authors investigated AoI-optimal scheduling in a wireless sensor network where the AoI values related to the sensors are not directly observable to the access point. In [11], the authors derived an efficient policy for sensor probing in an IoT network with intermittent faults and inexact knowledge about the status (healthy or faulty) of the system.

To the best of our knowledge, this is the first work that proposes an optimal policy for (on-demand) AoI minimization with an EH sensor, where the decision-making relies only on partial battery knowledge about the sensor's battery level.

II. SYSTEM MODEL AND PROBLEM FORMULATION

A. Network Model

We consider a status update system, where an energy harvesting (EH) sensor sends status updates about the monitored random process to users via a cache-enabled gateway, as depicted in Fig. 1. This models, e.g., an IoT sensing network, where the gateway represents an edge node; we refer to the gateway as the edge node henceforth. A time-slotted system with slots $t \in \mathbb{N}$ is considered.

We consider request-based status updating, where, at the beginning of slot t , users request for the status of the sensor (i.e., a new measurement) from the edge node. The edge node, which has a cache that stores the most recently received *status update* from the sensor, handles the arriving requests during the same slot t by the following procedure. Let $r(t) \in \{0, 1\}$, $t = 1, 2, \dots$ denote the random process of requesting the status of the sensor at slot t ; $r(t) = 1$ if the status is requested (by at least one user) and $r(t) = 0$ otherwise. The

*Centre for Wireless Communications, University of Oulu, Finland. e-mail: {mohammad.hatami, markus.leinonen}@oulu.fi. †Department of Science and Technology, Linköping University, Sweden. e-mail: marian.codreanu@liu.se. An extended version of the work including all the proofs can be found in [1].

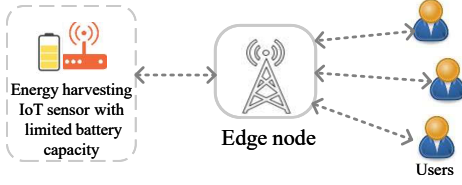


Fig. 1: A status update system, where an EH sensor sends status updates about the monitored random process to users via a cache-enabled edge node (i.e., the gateway).

requests are independent across time slots and the probability of having a request at each time slot is $p = \Pr\{r(t) = 1\}$. Upon receiving a request at slot t , the edge node serves the requesting user(s) by 1) commanding the sensor to send a fresh status update or 2) using the stored measurement from the cache. Let $a(t) \in \mathcal{A} = \{0, 1\}$ be the *command action* of the edge node at slot t ; $a(t) = 1$ if the edge node commands the sensor to send an update and $a(t) = 0$ otherwise.

B. Energy Harvesting Sensor

The sensor relies on the energy harvested from the environment. We model the energy arrivals $e(t) \in \{0, 1\}$, $t = 1, 2, \dots$, as a Bernoulli process with rate $\lambda = \Pr\{e(t) = 1\}$, $\forall t$. The sensor stores the harvested energy into a battery of finite size B (units of energy). Let $b(t)$ denote the battery level of the sensor at the beginning of slot t , where $b(t) \in \{0, \dots, B\}$.

We assume that transmitting a status update from the sensor to the edge node consumes one unit of energy (see, e.g., [5]). Once the sensor is commanded (i.e., $a(t) = 1$), the sensor sends an update if its battery is non-empty (i.e., $b(t) \geq 1$). Let $d(t) \in \{0, 1\}$ denote the *action of the sensor* at slot t ; $d(t) = 1$ if the sensor sends a status update and $d(t) = 0$ otherwise. Thus, $d(t) = a(t) \mathbb{1}_{\{b(t) \geq 1\}}$, where $\mathbb{1}_{\{\cdot\}}$ is the indicator function.

The evolution of the battery level is given by

$$b(t+1) = \min\{b(t) + e(t) - d(t), B\}. \quad (1)$$

C. Status Updating with Partial Battery Knowledge

We consider that the edge node is informed about the sensor's battery level (only) via the received *status update packets*. Considering this realistic setting is in stark contrast to the previous works on AoI-aware design which all assume perfect battery knowledge available at each time slot.

Each status update packet consists of the measured value of the sensor, a time stamp representing the time when the sample was generated, and the current battery level of the sensor. Consequently, the edge node has only *partial* knowledge about the battery level at each time slot, i.e., *outdated* knowledge based on the sensor's last update. Formally, let $\tilde{b}(t) \in \{1, 2, \dots, B\}$ denote the *knowledge* about the battery level of the sensor at the edge node at slot t . At slot t , let $u(t)$ denote the most recent slot in which the edge node received a status update packet, i.e., $u(t) = \max\{t' | t' < t, d(t') = 1\}$. Thus, $\tilde{b}(t) = b(u(t))$.

D. On-demand Age of Information

To account for the request-based status updating, we use the notion of age of information (AoI) [2] and measure the

freshness of information seen by the users via *on-demand AoI* [5], [6]. Let $\Delta(t)$ be the AoI about the monitored random process at the edge node at the beginning of slot t , i.e., the number of slots elapsed since the generation of the most recently received status update at the sensor. Thus, the AoI is defined as $\Delta(t) = t - u(t)$. We make a common assumption that $\Delta(t)$ is upper-bounded by a finite value Δ^{\max} , i.e., $\Delta(t) \in \{1, 2, \dots, \Delta^{\max}\}$. Besides tractability, this accounts for the fact that once the available measurement becomes excessively stale, further counting would be irrelevant. The evolution of $\Delta(t)$ is expressed as

$$\Delta(t+1) = \begin{cases} 1, & \text{if } d(t) = 1, \\ \min\{\Delta(t) + 1, \Delta^{\max}\}, & \text{if } d(t) = 0. \end{cases} \quad (2)$$

We define on-demand AoI at slot t as

$$\begin{aligned} \Delta^{\text{OD}}(t) &\triangleq r(t)\Delta(t+1) \\ &= r(t) \min\{(1-d(t))\Delta(t) + 1, \Delta^{\max}\}. \end{aligned} \quad (3)$$

In (3), since the requests come at the beginning of slot t and the edge node sends measurements to the users at the end of the same slot, $\Delta(t+1)$ is the AoI seen by the users.

E. Problem Formulation

We aim to find the best action of the edge node at each time slot, i.e., $a(t)$, $t = 1, 2, \dots$, called an *optimal policy*, that minimizes the average cost (i.e., on-demand AoI), defined as

$$\bar{C} = \lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T \mathbb{E}[\Delta^{\text{OD}}(t)], \quad (4)$$

where the expectation is taken over all system dynamics.

III. POMDP MODELING

We model the problem of finding an optimal policy as a POMDP and propose an iterative algorithm that finds such an optimal policy. The POMDP is defined by a tuple $(\mathcal{S}, \mathcal{O}, \mathcal{A}, \Pr(s(t+1)|s(t), a(t)), \Pr(o(t)|s(t), a(t-1)), c(s(t), a(t)))$ [12, Chap. 7], with the following elements.

- **State Space \mathcal{S} :** Let $s(t) \in \mathcal{S}$ denote the system state at slot t , which is defined as $s(t) = (b(t), r(t), \Delta(t), \tilde{b}(t))$. The state space \mathcal{S} has a finite dimension $|\mathcal{S}| = 2B(B+1)\Delta^{\max}$. We denote the *observable* (visible by the edge node) part of the state by $s^v(t) = (r(t), \Delta(t), \tilde{b}(t))$, thus $s(t) = (b(t), s^v(t))$.

- **Observation Space \mathcal{O} :** Let $o(t) \in \mathcal{O}$ be the edge node's observation about the system state at slot t , which is defined as the visible part of the state; $o(t) = s^v(t)$. The observation space \mathcal{O} has a finite dimension $|\mathcal{O}| = 2B\Delta^{\max}$.

- **Action Space \mathcal{A} :** At each time slot, the edge node decides whether to command the sensor or not, i.e., $a(t) \in \mathcal{A} = \{0, 1\}$.

- **State Transition Function $\Pr(s(t+1)|s(t), a(t))$:** The state transition function maps a state-action pair at slot t onto a distribution of states at slot $t+1$; the probability of transition from current state $(s(t))$ $s = (b, r, \Delta, \tilde{b})$ to next state $(s(t+1))$ $s' = (b', r', \Delta', \tilde{b}')$ under action $a(t) = a$ is given by

$$\begin{aligned} &\Pr(b', r', \Delta', \tilde{b}' | b, r, \Delta, \tilde{b}, a) = \\ &\Pr(r') \Pr(b' | b, a) \Pr(\Delta', \tilde{b}' | b, \tilde{b}, \Delta, a), \end{aligned} \quad (5)$$

where

$$\Pr(r') = \begin{cases} p, & r' = 1, \\ 1 - p, & r' = 0. \end{cases}$$

$$\begin{aligned} \Pr(b' | b = B, a = 0) &= \mathbb{1}_{\{b'=B\}}, \\ \Pr(b' | b < B, a = 0) &= \begin{cases} \lambda, & b' = b + 1, \\ 1 - \lambda, & b' = b, \\ 0, & \text{otherwise.} \end{cases} \\ \Pr(b' | b = 0, a = 1) &= \begin{cases} \lambda, & b' = 1, \\ 1 - \lambda, & b' = 0, \\ 0, & \text{otherwise.} \end{cases} \\ \Pr(b' | b \geq 1, a = 1) &= \begin{cases} \lambda, & b' = b, \\ 1 - \lambda, & b' = b - 1, \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

$$\begin{aligned} \Pr(\Delta', \tilde{b}' | b, \tilde{b}, \Delta, a = 0) &= \mathbb{1}_{\{\Delta' = \min\{\Delta+1, \Delta^{\max}\}, \tilde{b}' = \tilde{b}\}}, \\ \Pr(\Delta', \tilde{b}' | b = 0, \tilde{b}, \Delta, a = 1) &= \mathbb{1}_{\{\Delta' = \min\{\Delta+1, \Delta^{\max}\}, \tilde{b}' = \tilde{b}\}}, \\ \Pr(\Delta', \tilde{b}' | b \geq 1, \tilde{b}, \Delta, a = 1) &= \mathbb{1}_{\{\Delta' = 1, \tilde{b}' = b\}}. \end{aligned}$$

• **Observation Function:** The observation function is given by $\Pr(o(t)|s(t), a(t-1)) = \mathbb{1}_{\{o(t)=s^v(t)\}}$.

• **Immediate Cost Function** $c(s, a)$: This is the expected one-step cost of taking action a in state $s = (b, r, \Delta, \tilde{b})$, which is calculated using (3), i.e., $c(s, a) = r[(1 - a\mathbb{1}_{\{b \geq 1\}})\Delta + 1]$.

A. Belief-State

In the POMDP formulation above, the system state $s(t)$ is not fully observable for the edge node – the decision maker – at slot t . In particular, the state consists of two parts as $s(t) = \{b(t), s^v(t)\}$. Accordingly, at slot t , the exact battery $b(t)$ is unknown to the edge node, whereas the request, AoI, and partial battery knowledge captured by $s^v(t)$ are observable. This incomplete state information in a POMDP causes challenges in the decision making, because the edge node can make decisions only based on the available/observed information or on the quantities derived from that information.

To counteract the insufficiency in the state information, we need to define state-like quantities that preserve the Markov property and summarize all the necessary information for the edge node – called *sufficient information states* – in respect to searching for an optimal policy [12, Chapter 7]. One sufficient information state is a *belief-state*. We define the belief-state at slot t as $z(t) = \{\beta(t), s^v(t)\} \in \mathcal{Z}(t)$, where $\beta(t)$ is *belief*¹ and $\mathcal{Z}(t)$ is the belief-state space. The belief at slot t is a $(B+1)$ -dimensional vector $\beta(t) = (\beta_0(t), \dots, \beta_B(t))^T \in \mathcal{B}$, representing the probability distribution on the possible values of battery levels, where $\mathcal{B} \subset \mathbb{R}^{(B+1) \times 1}$ is the belief space. Let $\phi^c(t)$ be the so-called *complete information state* at slot t , which consists of an initial probability distribution over the states, and the complete history of observations and actions up to slot t , i.e., $\{o(1), \dots, o(t), a(1), \dots, a(t-1)\}$. Formally, the belief $\beta(t)$ represents the conditional probability that the battery level

is at a specific level given the complete information state $\phi^c(t)$; thus, the entries of $\beta(t)$ are defined as

$$\beta_j(t) = \Pr(b(t) = j | \phi^c(t)), \quad j \in \{0, 1, \dots, B\}. \quad (6)$$

The belief is updated at each time slot based on the previous belief, the current observation, and the previous action, i.e., $\beta(t+1) = \tau(\beta(t), o(t+1), a(t))$, where $\tau(\cdot)$ is the belief update function, given by the following proposition.

Proposition 1. *The belief update function $\tau(\cdot)$ is given by*

$$\begin{aligned} \beta(t+1) &= \tau(\beta(t), o(t+1), a(t)) = \\ &\begin{cases} \Lambda\beta(t), & \begin{cases} a(t) = 0, \\ o(t+1) = \{r(t+1), \min\{\Delta(t) + 1, \Delta^{\max}\}, \tilde{b}(t)\}, \end{cases} \\ \rho^0, & \begin{cases} a(t) = 1, \\ o(t+1) = \{r(t+1), \min\{\Delta(t) + 1, \Delta^{\max}\}, \tilde{b}(t)\}, \end{cases} \\ \rho^1, & \begin{cases} a(t) = 1, \\ o(t+1) = \{r(t+1), 1, 1\}, \end{cases} \\ \dots & \\ \rho^B, & \begin{cases} a(t) = 1, \\ o(t+1) = \{r(t+1), 1, B\}, \end{cases} \end{cases} \end{aligned} \quad (7)$$

where the matrix $\Lambda \in \mathbb{R}^{(B+1) \times (B+1)}$ is a left stochastic matrix, having a banded form as

$$\Lambda = \begin{pmatrix} 1 - \lambda & 0 & \dots & 0 & 0 \\ \lambda & 1 - \lambda & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 - \lambda & 0 \\ 0 & 0 & \dots & \lambda & 1 \end{pmatrix}, \quad (8)$$

and the vectors $\rho^0, \rho^1, \rho^2, \dots, \rho^B$ are given by

$$\begin{aligned} \rho^0 &= \rho^1 = (1 - \lambda, \lambda, 0, 0, \dots, 0, 0)^T \\ \rho^2 &= (0, 1 - \lambda, \lambda, 0, \dots, 0, 0)^T \\ &\dots \\ \rho^B &= (0, 0, 0, 0, \dots, 1 - \lambda, \lambda)^T. \end{aligned}$$

IV. OPTIMAL POLICY AND PROPOSED ALGORITHM

In this section, we find an optimal policy for the POMDP formulation. Formally, a policy π is a mapping from the belief-state space to the action space, i.e., $\pi : \mathcal{Z} \rightarrow \mathcal{A}$; the action taken in a belief-state $z(t) = z$ under policy π is denoted by $\pi(z)$. Under a policy π , the average cost is given by (see (4))

$$\bar{C}_\pi = \lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T \mathbb{E}_\pi[\Delta^{\text{OD}}(t) | z(0)], \quad (9)$$

where $\mathbb{E}_\pi[\cdot]$ denotes the expected value of $c(t)$ given that the edge node follows π , and $z(0)$ is the initial belief-state. In accordance with Section II-E, our objective is to find an optimal policy π^* that minimizes (9), i.e., $\pi^* = \arg \min_\pi \bar{C}_\pi$.

Theorem 1. *An optimal policy π^* is obtained by solving the following equations*

$$\bar{C}^* + h(z) = \min_{a \in \mathcal{A}} Q(z, a), \quad z \in \mathcal{Z}, \quad (10)$$

where $h(z)$ is a relative value function, \bar{C}^* is the optimal

¹Note that, in general, the belief of a POMDP is a probability distribution over the entire state space \mathcal{S} . However, because $s^v(t)$ is fully observable in our problem, it has no uncertainty to be modelled via a belief.

TABLE I: The belief space \mathcal{B} . The row and column numbers are used to represent each belief, e.g., $\Lambda^2\beta$ is represented by (0, 2).

	0	1	2	3	...	M	...
0	$\beta(1)$	$\Lambda\beta(1)$	$\Lambda^2\beta(1)$	$\Lambda^3\beta(1)$...	$\Lambda^M\beta(1)$...
1	ρ^1	$\Lambda\rho^1$	$\Lambda^2\rho^1$	$\Lambda^3\rho^1$...	$\Lambda^M\rho^1$...
...
B	ρ^B	$\Lambda\rho^B$	$\Lambda^2\rho^B$	$\Lambda^3\rho^B$...	$\Lambda^M\rho^B$...
	\mathcal{B}						

average cost achieved by π^* which is independent of the initial state $z(0)$, and $Q(z, a)$ is an action-value function, which, for actions $a = 0$ and $a = 1$, is given by

$$Q(z, 0) = r \min\{\Delta + 1, \Delta^{\max}\} + \sum_{r'=0}^1 [r'p + (1 - r')(1 - p)]h(\Lambda\beta, r', \min\{\Delta + 1, \Delta^{\max}\}, \tilde{b}), \quad (11a)$$

$$Q(z, 1) = [r\beta_0 \min\{\Delta + 1, \Delta^{\max}\} + r(1 - \beta_0)] + \beta_0 \sum_{r'=0}^1 [r'p + (1 - r')(1 - p)]h(\rho^0, r', \min\{\Delta + 1, \Delta^{\max}\}, \tilde{b}) + \sum_{j=1}^B \beta_j [ph(\rho^j, 1, 1, j) + (1 - p)h(\rho^j, 0, 1, j)]. \quad (11b)$$

Then, an optimal action taken in belief-state z is obtained as

$$\pi^*(z) = \arg \min_{a \in \mathcal{A}} Q(z, a), \quad z \in \mathcal{Z}. \quad (12)$$

An optimal policy π^* can be found by turning the Bellman's optimality equation (11) into an iterative procedure, called relative value iteration algorithm (RVIA) [13, Section 8.5.5]. Particularly, at each iteration $i = 0, 1, \dots$, we have

$$V^{(i+1)}(z) = \min_{a \in \mathcal{A}} Q^{(i+1)}(z, a), \quad (13)$$

$$h^{(i+1)}(z) = V^{(i+1)}(z) - V^{(i+1)}(z_{\text{ref}}),$$

where $z_{\text{ref}} \in \mathcal{S}$ is an arbitrary reference state and $Q^{(i+1)}(z, a)$, $a \in \{0, 1\}$, is given by

$$Q^{(i+1)}(z, 0) = r \min\{\Delta + 1, \Delta^{\max}\} + \sum_{r'=0}^1 [r'p + (1 - r')(1 - p)]h^{(i)}(\Lambda\beta, r', \min\{\Delta + 1, \Delta^{\max}\}, \tilde{b}),$$

$$Q^{(i+1)}(z, 1) = [r\beta_0 \min\{\Delta + 1, \Delta^{\max}\} + r(1 - \beta_0)] + \beta_0 \sum_{r'=0}^1 [r'p + (1 - r')(1 - p)]h^{(i)}(\rho^0, r', \min\{\Delta + 1, \Delta^{\max}\}, \tilde{b}) + \sum_{j=1}^B \beta_j [ph^{(i)}(\rho^j, 1, 1, j) + (1 - p)h^{(i)}(\rho^j, 0, 1, j)]. \quad (14)$$

For any initialization $V^{(0)}(z)$, the sequences $\{Q^{(i)}(z, a)\}_{i=1,2,\dots}$, $\{h^{(i)}(z)\}_{i=1,2,\dots}$ and $\{V^{(i)}(z)\}_{i=1,2,\dots}$ converge, i.e., $\lim_{i \rightarrow \infty} Q^{(i)}(z, a) = Q(z, a)$, $\lim_{i \rightarrow \infty} h^{(i)}(z) = h(z)$ and $\lim_{i \rightarrow \infty} V^{(i)}(z) = V(z)$, $\forall z$. Thus, $h(z) = V(z) - V(z_{\text{ref}})$ satisfies (10) and $\bar{C}^* = V(z_{\text{ref}})$.

Theorem 2. $V(z)$ is fixed with respect to \tilde{b} .

Corollary 1. According to Theorem 2, $V(z)$, $z = \{\beta, r, \Delta, \tilde{b}\}$, and consequently $h(z)$ and $Q(z, a)$, do not depend on \tilde{b} . Thus, \tilde{b} does not have any impact on calculating π^* in (12). Therefore, we remove² \tilde{b} from the belief-state z and redefine the belief-state hereinafter as $z = \{\beta, r, \Delta\} \in \mathcal{Z}$. This will be exploited to reduce the computational complexity of the proposed algorithm. Note that (11) can easily be rewritten based on the new belief-state definition.

²Note that while \tilde{b} is removed from the belief-state, it is still needed to calculate the belief $\beta(t)$.

Algorithm 1 Proposed algorithm that obtains π^* .

- 1: **Initialize** $V(z) = h(z) = 0$, $\forall z = \{\beta, r, \Delta\}$, $\beta \in \hat{\mathcal{B}}$, $r \in \{0, 1\}$, $\Delta \in \{1, \dots, \Delta^{\max}\}$, determine an arbitrary $z_{\text{ref}} \in \mathcal{Z}$, a small threshold $\theta > 0$, and large M such that $\Lambda^M\beta \approx \Lambda^{(M+1)}\beta$
- 2: **repeat**
- 3: **for** z **do**
- 4: calculate $Q(z, 0)$ and $Q(z, 1)$ by using (14)
- 5: $V_{\text{tmp}}(z) \leftarrow \min_{a \in \mathcal{A}} Q(z, a)$
- 6: **end for**
- 7: $\delta \leftarrow \max_z (V_{\text{tmp}}(z) - V(z)) - \min_z (V_{\text{tmp}}(z) - V(z))$
- 8: $V(z) \leftarrow V_{\text{tmp}}(z)$ and $h(z) \leftarrow V(z) - V(z_{\text{ref}})$, for all z
- 9: **until** $\delta < \theta$
- 10: $\pi^*(z) = \arg \min_{a \in \mathcal{A}} Q(z, a)$, for all z

Although the sequences in (13) converges, finding such $V(z)$ (and $h(z)$) iteratively via (13) is intractable, because the belief space \mathcal{B} has infinite dimension. Fortunately, there is a certain pattern in the evolution of the belief-states $\{z(t)\}$, which allows us to perform a truncation of belief space \mathcal{B} and subsequently develop a practical iterative algorithm relying on (13). Particularly, given the initial belief $\beta(1)$, the belief at slot $t = 2$ is chosen from the set $\{\Lambda\beta(1), \{\rho^j\}_{j=1}^B\}$, the belief at $t = 3$ is a member of the set $\{\Lambda^2\beta(1), \{\Lambda\rho^j\}_{j=1}^B, \{\rho^j\}_{j=1}^B\}$, the belief at $t = 4$ is selected from the set $\{\Lambda^3\beta(1), \{\Lambda^2\rho^j\}_{j=1}^B, \{\Lambda\rho^j\}_{j=1}^B, \{\rho^j\}_{j=1}^B\}$, and etc. Thus, the belief space \mathcal{B} , which contains all the possible beliefs $\beta(t)$, $\forall t$, is countable but infinite, as shown in Table I.

The following lemma expresses an important property of the matrix Λ in (8), which is used to truncate the belief space \mathcal{B} into a finite space $\hat{\mathcal{B}}$.

Lemma 1. The m th power of matrix Λ is given by

$$\Lambda_{j,l}^m = \begin{cases} 0, & j < l, \\ (1 - \lambda)^m, & j = l, \\ \lambda^{(j-l)}(1 - \lambda)^{(m-j+l)} \prod_{v=0}^{j-l-1} \frac{(m-v)}{(v+1)}, & l < j \leq B, \\ 1 - \sum_{j'=1}^B \Lambda_{j',l}^m, & j = B + 1, \end{cases}$$

where $\Lambda_{j,l}^m$ is the entry of matrix Λ^m at its j th row and l th column. Thus, for any $\lambda > 0$, we have

$$\lim_{m \rightarrow \infty} \Lambda_{j,l}^m = \begin{cases} 0, & j \leq B, \\ 1, & j = B + 1, \end{cases} \quad (15)$$

Corollary 2. By Lemma 1, for a sufficiently large integer M , we have $\Lambda^M \approx \Lambda^{M+1}$. Thus, we construct a truncated belief space $\hat{\mathcal{B}}$ of finite dimension $|\hat{\mathcal{B}}| = (B+1)M$, as shown in Table I. Intuitively, the value M represents the maximum number of consecutive no-command actions ($a = 0$) for which the belief is updated; from the $(M+1)$ th no-command onward, the belief is no longer updated. This is reasonable because after M consecutive $a = 0$ actions, the battery is almost full, i.e., $\Lambda^M\beta \rightarrow (0, 0, \dots, 0, 1)^T$, $\forall \beta \in \mathcal{B}$, and thus, for sufficiently large M , the space $\hat{\mathcal{B}}$ contains (almost) all the possible beliefs.

Finally, considering the truncated belief space $\hat{\mathcal{B}}$, we use (13)–(14) to find $V(z)$, $h(z)$, $Q(z, a)$, and consequently an optimal policy π^* iteratively, as presented in Algorithm 1.

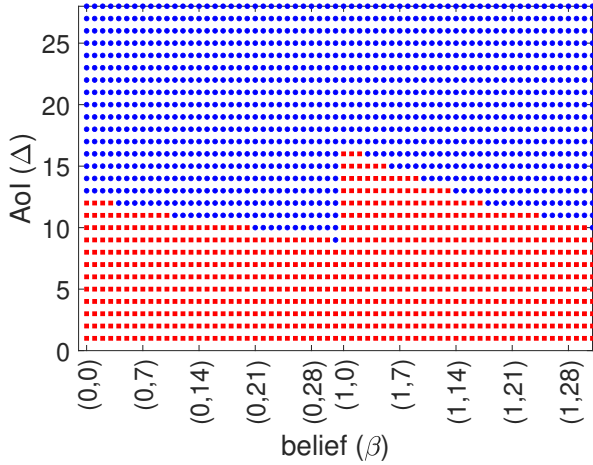


Fig. 2: Structure of an optimal policy $\pi^*(z)$ for each belief-state $z = \{\beta, 1, \Delta\}$, where $p = 0.8$, $\lambda = 0.08$, and initial belief $\beta(1) = (1/3, 1/3, 1/3)$. Red: no command $a = 0$; blue: command $a = 1$.

V. SIMULATION RESULTS

We consider a scenario with $\lambda = 0.08$, $p = 0.8$, $\Delta^{\max} = 64$, and $B = 2$. Fig. 2 illustrates the structure of an optimal policy π^* , where each point represents a potential belief-state as a three-tuple $z = (\beta, 1, \Delta)$. For each such z , a blue point indicates that the optimal action is to command the sensor (i.e., $\pi^*(z) = 1$), whereas a red point means not to command. Fig. 2 illustrates that π^* has a *threshold-based* structure with respect to the AoI. To exemplify, consider the belief-state $z = \{(1, 7), 1, 20\}$ in which $\pi^*(z) = 1$; then, by the threshold-based structure, $\pi^*(z) = 1$ for all $z = \{(1, 10), 1, \Delta\}$, $\Delta \geq 20$.

Fig. 3 depicts the performance of the proposed algorithm. In the *request-aware greedy (myopic)* policy, the edge node commands the sensor whenever there is a request (i.e., $r(t) = 1$). As benchmark, we consider a case that the edge node knows the exact battery level at each time slot. In this case, an optimal policy, denoted by $\pi_{\text{MDP}}(s), \forall s$, can be found by the value iteration algorithm [6]. Clearly, this policy serves as a lower bound to the proposed POMDP-based algorithm. As shown in Fig. 3, for sufficiently large M (say, $M \geq 32$), the proposed algorithm obtains optimal performance. Furthermore, the proposed method reduces the average cost by 28 % compared to the greedy policy.

VI. CONCLUSIONS

We considered status updating under inexact knowledge of the battery level of an EH sensor that sends updates to users via a cache-enabled edge node. Accounting for the partial battery knowledge at the edge node, we derived a POMDP model for the on-demand AoI minimization problem. We converted the POMDP into a belief-state MDP and, via characterizing its key structures, developed an iterative algorithm that obtains an optimal policy. We numerically depicted that an optimal POMDP-based policy has a threshold-based structure and demonstrated the performance gains obtained by the proposed algorithm compared to a request-aware greedy policy.

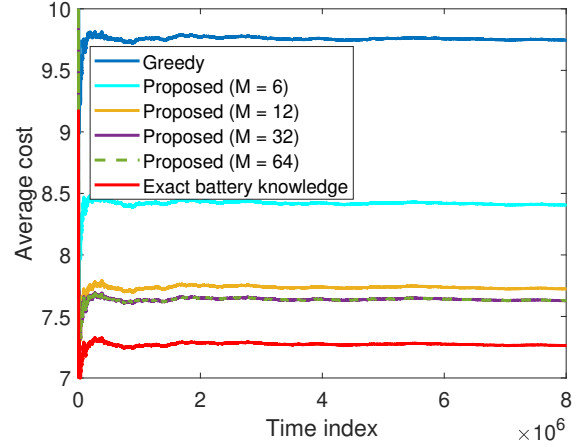


Fig. 3: Performance of the proposed algorithm.

VII. ACKNOWLEDGMENTS

This research has been financially supported by the Infotech Oulu, the Academy of Finland (grant 323698), and Academy of Finland 6G Flagship program (grant 346208). M. Hatami would like to acknowledge the support of Nokia Foundation. The work of M. Leinonen has also been financially supported in part by the Academy of Finland (grant 340171).

REFERENCES

- [1] M. Hatami, M. Leinonen, and M. Codreanu, "Status updating with an energy harvesting sensor under partial battery knowledge," *arXiv preprint*, 2022.
- [2] S. Kaul, R. Yates, and M. Gruteser, "Real-time status: How often should one update?" in *Proc. IEEE Int. Conf. on Computer. Commun. (INFOCOM)*, Orlando, FL, USA, Mar. 25–30, 2012, pp. 2731–2735.
- [3] Y. Sun, I. Kadota, R. Talak, E. Modiano, and R. Srikant, "Age of information: A new metric for information freshness," *Synthesis Lectures on Communication Networks*, vol. 12, no. 2, pp. 1–224, 2019.
- [4] M. Hatami, M. Jahandideh, M. Leinonen, and M. Codreanu, "Age-aware status update control for energy harvesting IoT sensors via reinforcement learning," in *Proc. IEEE Int. Symp. Pers., Indoor, Mobile Radio Commun.*, London, UK, Aug. 31–Sep. 3 2020.
- [5] M. Hatami, M. Leinonen, and M. Codreanu, "AoI minimization in status update control with energy harvesting sensors," *IEEE Trans. Commun.*, vol. 69, no. 12, pp. 8335–8351, Dec. 2021.
- [6] —, "Minimizing average on-demand AoI in an IoT network with energy harvesting sensors," in *Proc. IEEE Works. on Sign. Proc. Adv. in Wirel. Comms.*, Lucca, Italy, Sep. 27–30, 2021, pp. 1–6.
- [7] M. Hatami, M. Leinonen, Z. Chen, N. Pappas, and M. Codreanu, "On-demand AoI minimization in resource-constrained cache-enabled IoT networks with energy harvesting sensors," *arXiv preprint arXiv:2201.12277*, 2022.
- [8] G. Yao, A. M. Bedewy, and N. B. Shroff, "Age-optimal low-power status update over time-correlated fading channel," in *Proc. IEEE Int. Symp. Inform. Theory*, Melbourne, Australia, Jul. 12–20, 2021, pp. 2972–2977.
- [9] A. Gong, T. Zhang, H. Chen, and Y. Zhang, "Age-of-information-based scheduling in multiuser uplinks with stochastic arrivals: A POMDP approach," in *Proc. IEEE Global Telecommun. Conf.*, Taipei, Taiwan, Dec. 7–11, 2020, pp. 1–6.
- [10] Y. Shao, Q. Cao, S. C. Liew, and H. Chen, "Partially observable minimum-age scheduling: The greedy policy," *IEEE Trans. Commun.*, vol. 70, no. 1, pp. 404–418, 2022.
- [11] G. Stamatakis, N. Pappas, A. Fragkiadakis, and A. Traganitis, "Semantics-aware active fault detection in status updating systems," *arXiv preprint arXiv:2202.00923*, 2022.
- [12] O. Sigaud and O. Buffet, *Markov decision processes in artificial intelligence*. John Wiley & Sons, 2013.
- [13] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.