Query Age of Information: Freshness in Pull-Based Communication

Federico Chiariotti, Josefine Holm, Anders E. Kalør, Beatriz Soret,
Søren K. Jensen, Torben B. Pedersen, and Petar Popovski

***Abstract*—Age of Information (AoI) has become an important concept in communications, as it allows system designers to measure the freshness of the information available to remote monitoring or control processes. However, its definition tacitly assumes that new information is used at any time, which is not always the case: the instants at which information is collected and used may be dependent on a certain query process, and resource-constrained environments such as most Internet of Things (IoT) use cases require precise timing to fully exploit the limited available transmissions. In this work, we consider a *pull-based communication model* in which the freshness of information is only important when the receiver generates a query: if the monitoring process is not using the value, the age of the last update is irrelevant. We optimize the Age of Information at Query (QAoI), a metric that samples the AoI at relevant instants, better fitting the pull-based resource-constrained scenario, and show how this can lead to very different choices. Our results show that QAoI-aware optimization can significantly reduce the average and worst-case perceived age for both periodic and stochastic queries.**

***Index Terms*—Age of Information, networked control systems, pull-based communication, Markov Decision Processes**

I. Introduction

Over the past few years, the concept of information freshness has received a significant attention in relation to cyber-physical systems that rely on communication of various updates in real time. This has led to the introduction of *Age of Information (AoI)* [1] as a measure that reflects the freshness at the receiver, and denotes the difference between the current time and the time when the most recently received update was generated at the sender.

In a common model for AoI-sensitive systems, a wireless-equipped sensor measures a physical process and transmits its readings using a wireless link to a destination, where a monitor processes the received information. On the other hand, we study the effect of the existence of a *query process* at the receiver in a resource-limited scenario in which a single sensor is constrained in how often it can transmit, either due to energy considerations or duty cycle limitations. In most works in the literature, the tacit assumption behind AoI has been that the monitor at the receiver is interested in having fresh information at any time. In other words, the model assumes a *push-based communication*, in which a hypothetical application residing at the monitor has a *permanent query* to the updates that arrive at the receiver. Our work considers a *pull-based communication* model, in which the query arrival process can guide the communication strategy by, for example, reading the sensor and transmitting the updates immediately before the query times. To design for pull-based communications, we introduce an age metric named Age of Information at Query (QAoI), which represents the age of the information available to the receiver in the instant when it actually needs it. This new metric is similar to the Effective Age of Information (EAoI), presented in [2], which studied the effect of queries in a different scenario, with multi-user scheduling and a constant channel. The QAoI metric and the system optimization based on it is tested in a scenario with *limited link availability* at the source, as our system model considers both when it is possible to transmit and when it is convenient to do so.

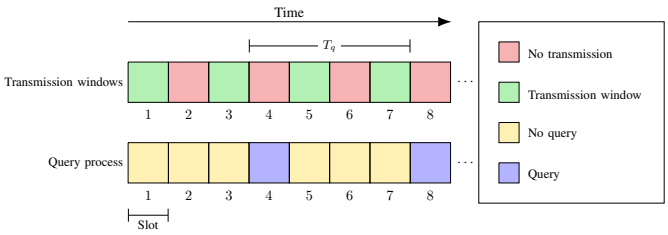
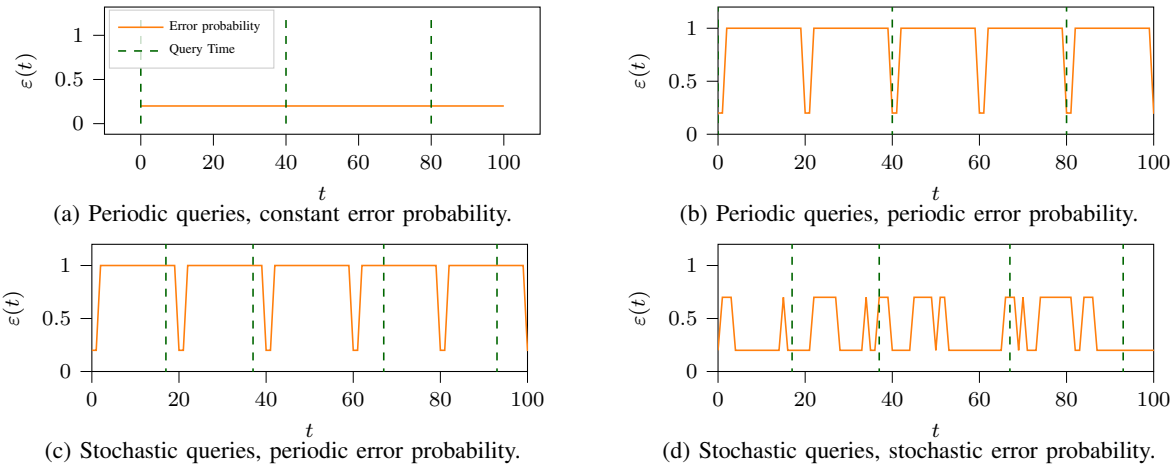


Fig. 1: Diagram of a satellite IoT-based pull-based communication model.

One can think of several scenarios that would fit this pull-based model, as most monitoring and control applications operate over discrete time intervals, or only activate to react to some external trigger or user input. An interesting practical use case for this concept is represented by the Satellite Internet of Things (IoT), which connects sensors in remote areas to the wider Internet through the use of Geosynchronous Earth Orbit (GEO) or Low Earth Orbit (LEO) satellites, and is still a mostly unexplored setting for the AoI literature [3], [4]. Fig. 1 shows an example of the transmission and query timing with periodic intervals: the transmitter can only send a packet in the green slots, and the application queries the received result in the blue slots. In the push-based communication model, the transmitter should optimize for freshness at any time, while in the pull-based model, if there is a successful transmission in, e.g., slot 7, any transmission in slot 5 will then be useless to the application, as it will never see it. The transmitter will try to send packets as close to the query instant as possible, even if this results in a larger age in between queries.

In our model, the channel between the sensor and the

F. Chiariotti, J. Holm, A. E. Kalør, B. Soret, and P. Popovski are with the Department of Electronic Systems, Aalborg University (email: {fchi,jho,aek,bsa,petarp}@es.aau.dk). S K. Jensen and T. B. Pedersen are with the Department of Computer Science, Aalborg University (email: {skj,tbp}@cs.aau.dk). This work has been partly supported the Danish Council for Independent Research (Grant No. 8022-00284B SEMIOTIC).



(a) Periodic queries, constant error probability.

(b) Periodic queries, periodic error probability.

(c) Stochastic queries, periodic error probability.

(d) Stochastic queries, stochastic error probability.

Fig. 2: Scenarios considered in the simulations.

intermediate cache, i.e., the satellite uplink and downlink in the Satellite IoT scenario, is abstracted as a Packet Erasure Channel (PEC), whose erasure probability can be either constant or time-varying. The query arrival process is, in general, a stochastic process. In our simulations, we consider four scenarios, which can correspond to three different Satellite IoT use cases:

- *Periodic queries and constant channel*: this is the simplest case, in which queries are deterministic and periodic and the channel error probability is constant over time. This model can represent a GEO monitoring application, in which the satellite is always in the same position relative to the ground and the remote monitoring application simply logs the data by querying the ground station at predetermined intervals. This corresponds to the scenario in Fig. 2a, in which $\varepsilon(t)$ represents the channel error probability at time t.
- *Periodic queries with a predictable channel*: in this case, we introduce some complexity in the channel behavior by having a time-varying error probability, while maintaining a deterministic and periodic query arrival process. This scenario can represent a LEO remote monitoring application, in which the sensor is not served by a GEO satellite, but by a constellation of LEO satellites, whose orbits bring them outside the coverage range of the sensor: when there are no visible satellites, the packet error probability is 1. The sensors must then transmit its data when at least one satellite is passing over it. The orbits of the satellites can be computed in advance, so these periods of availability are known, but the sensor will be constrained in its scheduling decisions. This corresponds to the scenario in Fig. 2b. Another possible example of this model would be a wireless scenario with recurring outages, during which the error probability for updates sharply increases.
- *Stochastic queries with a predictable channel*: in this case, the channel error probability can change over time and queries arrive according to a stochastic process, so the sensor will need to optimize the expected QAoI, considering the probability of queries arriving in the near future. This scenario can represent a human-in-the-loop LEO monitoring application, in which queries are driven by the behavior of the user, and are then only partially predictable. This corresponds to the scenario in Fig. 2c.
- *Stochastic queries with a stochastic channel*: this is the most general case, in which queries are stochastic, and the channel error probability is not predictable, but follows a stochastic process such as a Markov chain. This case can represent a general wireless channel, which does not have pre-computed satellite passes, but depends on the propagation conditions, and is shown in Fig. 2d. It corresponds to a general wireless edge system getting queries from the Cloud.

These four examples are described in more detail and adapted to the communication model in Sec. VI. Besides the satellite IoT application of Fig. 1, our model can fit several other monitoring applications, and the formulation is fully general for relaying scenarios with an intermediate cache node fulfilling requests from the end user. A generic example includes queries that are periodically/regularly sent from a central cloud to an edge node.

In this work, we model a scheduling problem for a resource-constrained sensor as a Markov Decision Process (MDP), showing the difference between a query-aware scheduler and a legacy one that tries to minimize AoI at any instant: in the most general case, we consider the query arrival process and the channel state to be driven by two independent Markov chains. The model in this paper extends the framework we presented in [5], which only considered a simple scenario with a constant channel and periodic queries. While previous works on AoI often dealt with limitations on the link availability, because of congestion, energy, or propagation constraints, but to the best of our knowledge, this is the first work to combine limitations on the channel availability with AoI in pull-based communication.

Our simulation results show that awareness of the query process can give significant gains in terms of both the average QAoI and its higher percentiles, which represent a worst-case result and are critical for reliability-oriented applications, improving the performance of monitoring systems even in the most general case. The query-aware optimization often incurs a cost in terms of AoI, as the scheduling strategy that optimizes

freshness when a query arrives will often let the age increase when the probability of a query arriving is low,

The rest of this paper is divided as follows: first, Sec. II presents the state of the art on scheduling and AoI minimization. We then present the QAoI metric and our communication system model in Sec. III, formulating it as an MDP and finding the optimal policies in Sec. V. We then present our simulation and its results in Sec. VI, considering a simple system first and gradually increasing its complexity. Finally, we conclude the paper and describe some possible avenues of future work in Sec. VII.

II. Related work

Over the last few years, AoI [6] has attracted a significant amount of interest from the research community, as it represents a more relevant metric than latency for the ongoing monitoring and control of processes over a network. Most works in the literature deal with AoI in queuing systems, examining different scheduling policies. Some recent works have proven that preemption, i.e., removing packets already in the queue in favor of newer ones with more up to date information, can give significant advantages in terms of average age [7], even when multiple $M/M/1$ queuing systems in tandem are involved [8]. However, preemption can be suboptimal for different service time distributions, as the decision over whether to preempt or to continue the ongoing transmission becomes more complex [9].

Other works addressed AoI in specific wireless scenarios with errors [10] and retransmissions [11], or basing their analysis on live experiments [12]. The addition of more sources in the queuing system leads to an interesting scheduling problem, which aims at finding the packet generation rate that minimizes the age for the whole system [13]. Optimizing the access method and senders' updating policies to minimize AoI in complex wireless communication system has been proven to be an NP-hard problem, but heuristics can achieve near-optimal solutions [14] by having sources decide whether an update is valuable enough to be sent (i.e., whether it would significantly reduce the AoI) [15]. The average AoI has been derived in slotted [16] and unslotted ALOHA [17], as well as in scheduled access [18], and the performance of scheduling policies has been combined with these access methods in [19].

The scheduling problem can be formulated both for multiple sources, in which case the scheduling problem involves balancing the ages of the different sources while avoiding interference [20], or for a single source with resource constraints: usually, these constraints are in the form of limited energy availability or enforced duty cycles. Energy harvesting nodes are considered in [21], which derives a near-optimal threshold-based heuristic that can work without knowledge of future energy generation, and by [22], which derives the optimal policy for nodes with infinite as well as finite batteries. The trade-off between energy and freshness is examined explicitly in [23], while [24] considers a noisy channel as well, in which updates can be randomly erased. A more complex scenario, which includes a Hybrid Automated Repeat Request (HARQ) channel as well as an energy harvesting node with a finite battery, is considered in [25], which models the problem as an MDP and finds the optimal policy with reinforcement learning. Another interesting case for the scheduling problem is AoI minimization in drone networks, in which drones have to move back and forth between the sensing location and the base station [26]: finding the correct balance to minimize AoI and energy expenditure is a non-trivial problem in this scenario.

To the best of our knowledge, most of the literature so far has adopted a push-based model, in which updates are always relevant to the monitoring process. We are aware of only a few other works that consider a pull-based model: the one most similar to this work [2] considers a server updating multiple users, using a metric called EAoI, which is similar to QAoI, although in their case the system is not constrained by energy considerations, but by the presence of multiple sensors that need to be scheduled appropriately. The effect of queries is also modeled slightly differently: in our case, there is a strict ordering between transmissions and queries, and the response to a query is always sent immediately. On the other hand, EAoI considers the possibility of a *delayed response* if a transmission from the sensor is scheduled but not yet received. In our work, we focus on the optimization of the connection between the sensor and the intermediate cache, considering significant communication constraints and a more challenging IoT scenario. The difference between our metric and EAoI is shown in Sec. IV. Another work, first presented by Li *et al.* in [27] and later expanded in [28], considers age not to matter unless and until a request for the information is generated. However, Li *et al.*'s work does not consider the effects of scheduling in a regime with limited transmission opportunities, but rather tries to provide freshness to the user by combining multiple replications of the sensor value over multiple servers. The innovation from [28] can be combined with ours with limited adaptations to the two models, resulting in a joint optimization of the whole end-to-end scheduling. Another work also models requests in the optimization function [29], but it only deals with memoryless request processes, which (as we will describe in the introduction) lead to a solution that is equivalent to standard AoI minimization. The extended version of that paper [30] considers more complex scenarios with partial battery knowledge, but still uses the same memoryless request model. Finally, a recent work by Xu *et al.* [31] also considers a memoryless request process, but considers a mix between traditional AoI and query-aware metrics. This is a significant difference from QAoI and related metrics such as EAoI, and it leads to a highly different optimization, which will prioritize users with a less active request process.

III. System model

We now define a simple system model and consider the QAoI metric. The notation in the following sections is summarized in Table I.

We consider a time-slotted system indexed by $t = 1, 2, \ldots$, and denote the time instants at which updates are successfully delivered to the edge node as $t_{u,1}, t_{u,2}, \ldots$. The source can be sampled at any time, and fresh information is always available, a condition known as zero-wait sampling. Following

Symbol	Description	Symbol	Description
$t_{u,i}$	Delivery time of the i-th update	T_q	Query arrival period
$\Delta(t)$	AoI at time t	$\mathcal{S}$	State space of the scheduling MDP
Δ_∞	Long-term expected AoI	$\mathcal{A}$	Action space of the scheduling MDP
$t_{q,i}$	Time of the i-th query	$p_a(s,s')$	Probability to go from s to s' for action a
$\mathcal{S}_q$	State space of the query process	$r(s,a,s')$	Instantaneous reward
$\mathcal{Q}$	Set of states in which a query arrives	$b(t)$	Number of available tokens at time t
P_q	Transition matrix of the query process	$c(s_t,a_t)$	Long-term expected cost
$\mathcal{S}_e$	State space of the error probability process	λ	Cost discount factor
P_e	Transition matrix of the error probability process	π	Action policy
$\varepsilon(s_e)$	Packet error probability for state s_e	$v_\pi(s_t)$	Expected state value with policy π
T_e	Packet error probability period	ε_0	Error probability during the satellite pass

TABLE I: Notation definitions.

the common definition of AoI considered in the literature, e.g. [6], [13] we denote the AoI in time slot t by $\Delta(t)$, and define it as the difference between t and the time at which the last successfully received packet was generated:

$$\Delta(t) = t - \max_{i:t_{u,i}\leq t} t_{u,i}. \tag{1}$$

We will assume that $t_{u,1} = 0$ so that $\Delta(t)$ is well defined. An alternative, but equivalent definition can be obtained by defining the time-varying variable u_t that takes the value 1 if a new update is received at the edge node in time slot t, and 0 otherwise:

$$\Delta(t) = \begin{cases} \Delta(t-1)+1 & \text{if } u_t = 0 \\ 1 & \text{if } u_t = 1 \end{cases} \tag{2}$$

where $\Delta(0) = 0$. This definition of AoI, as given in [6], considers the freshness of information at any given point in time. The long-term expected AoI Δ_∞ is given by:

$$\Delta_\infty = \limsup_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\Delta(t)\right]. \tag{3}$$

This formulation does not include any weighting, assuming that all time steps are equally important. This is reasonable if the monitoring system is either continuous or much faster than the update generating process and communication system, i.e., can be considered as essentially continuous. However, this is only one possibility in real monitoring and control systems: discrete-time systems involve queries in which the monitoring process samples the available information. To capture such applications, we introduce the QAoI metric, which samples $\Delta(t)$ according to an arbitrary querying process, thereby considering only the instants at which a query arrives. In this case, we can consider long-term AoI as a special case of QAoI in which queries arrive at every time instant.

Naturally, in order for an update to be received successfully in slot t, we need to transmit it: the *policy* to transmit an update is a function $\pi : \mathcal{S} \to \{0,1\}$, where $\mathcal{S}$ is a state space and an update is transmitted if the policy outputs 1.

A. The QAoI metric

If the query arrival process is known in advance, we denote the query arrival times at the edge node by $t_{q,1}, t_{q,2}, \ldots$. We can then define the QAoI for the i-th query, denoted as $\tau(i)$ and given by:

$$\tau(i) = \Delta(t_{q,i}). \tag{4}$$

The EAoI metric proposed in [2] shares many similarities with QAoI, and indeed it also represents a pull-based system; however, it deals with concurrent queries and updates differently, allowing the server to wait until the update is over to respond to the query, while our formulation is stricter and enforces an order, with updates always arriving before queries. We then define the overall objective as minimizing the long-term expected QAoI, defined as

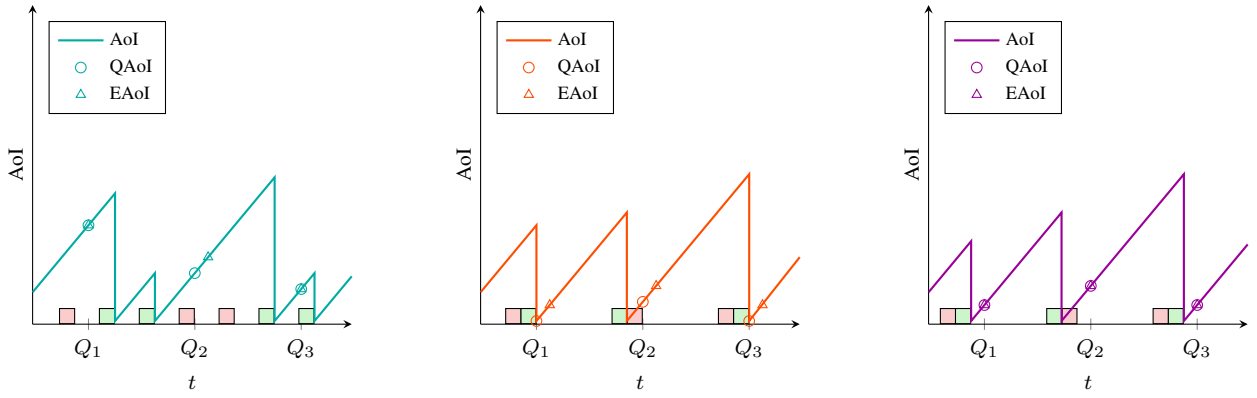
$$\tau_\infty = \limsup_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\sum_{i:t_{q,i}\leq T}\Delta(t_{q,i})\right]. \tag{5}$$

It is also possible to optimize QAoI without full knowledge of future query arrival times, as long as there is some information on the statistics of the process: in our model, the query process is represented by a finite Markov chain with a state space $\mathcal{S}_q$ and a transition matrix P_q. The query process is then in a (known) state at any time instant, and queries are generated if the state is in a predetermined subset $\mathcal{Q} \subseteq \mathcal{S}_q$.

Relating this to the use case example, the Markov chain represents the monitoring application: in the simplest case, it requests the sensor reading to the ground station periodically, but in general queries can have complex periodicities that can be modeled by a Markovian process. In most of the simple cases, we have $|\mathcal{Q}| = 1$, and the interval between two consecutive queries is Independent and Identically Distributed (IID). We can then rewrite the long-term QAoI in the more general case with stochastic queries as the following:

$$\tau_\infty(s_q) = \limsup_{T\to\infty} \mathbb{E}\left[\sum_{t=0}^{T-1}\frac{\Delta(t+1)}{T}\sum_{s'_q(t+1)\in\mathcal{Q}} P_q(s_q, s'_q; t+1)\right], \tag{6}$$

where $P_q(s_q, s'_q; t+1)$ is the probability of transitioning from state s_q to state s'_q after $t+1$ steps, specified by the (s_q, s'_q)-th entry in $(P_q)^{t+1}$. In this way, the QAoI is considered only in the instants in which a query is happening, i.e., when the Markov chain representing the query process is in a state in which a query arrives. It is also possible to consider normalizing the QAoI over the number of queries instead of the number of steps, but this simply leads to a constant multiplying factor as T tends to infinity.



(a) AoI optimization. (b) QAoI optimization. (c) EAoI optimization.

Fig. 3: Example of the difference between a system assuming a permanent query and one that is aware of the query arrival process. The same packets are lost in all systems (correctly received packets are depicted in green, lost ones in red), and the markers indicate the age at the query arrival instants.

If the query arrival process is memoryless, e.g., queries follow a Bernoulli process, the strategies to minimize AoI and QAoI are the same, as well as their distributions, as the query process transition probabilities are independent of the current state. The same is true when the query arrival process is much faster than the sensor, i.e., when there is a query in each time slot. The opposite extreme is the case with deterministic query arrivals, in which the transition matrix is deterministic: the query arrival instants are then known *a priori*, and the sensor can optimize its transmissions to minimize QAoI directly. The most obvious example of this is given by periodic queries, but the same holds for any deterministic process that is known to the sensor.

In general, the QAoI can be very different from the AoI in a given system, as well as the strategies for optimizing it. In this work, we present such a class of systems, arguing that an approach that takes the nature and possible periodicity of the monitoring process into account can fit more situations and result in better performance than standard AoI minimization. This difference is shown in a simple example in Fig. 3, in which the three metrics lead to different solutions: while minimizing AoI leads to periodic transmissions, QAoI and EAoI lead the sensor to transmit just before queries. However, EAoI-oriented systems would transmit one slot earlier, so as to avoid delaying the response to the query.

B. Communication system model

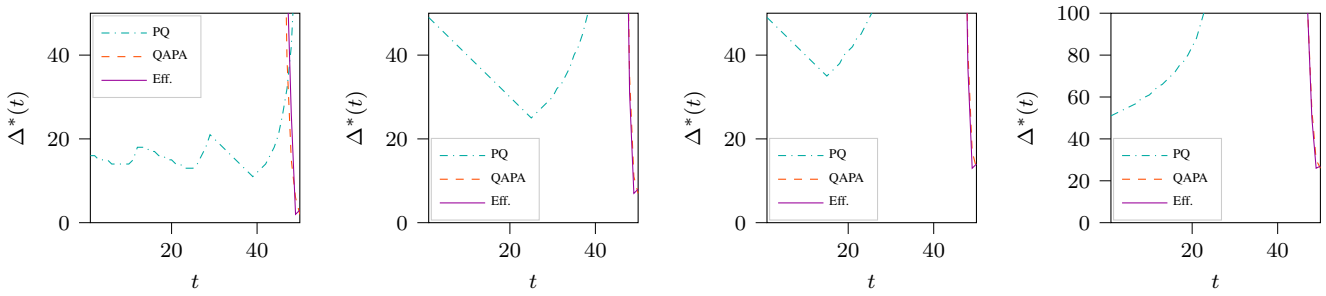
We now define the model of our pull-based communication scenario, in which a sensor needs to schedule transmissions over a link with limited availability. Independently, queries about the state of the sensor are generated, e.g., as part of a monitoring or control process. The objective of this work is to maximize the freshness of the information *at the query time*, while considering that the sensor is energy-constrained and needs to limit the number of transmissions to prolong its lifetime.

We assume that each update has a fixed size and is transmitted over a PEC with slotted time. The sensor and receiver are assumed to be synchronized, i.e., the sensor is informed when the last query arrived by the time of the next transmission window. In order to be as general as possible, we model the channel as a Markov chain with state space $\mathcal{S}_e$ and transition matrix P_e, as we did for the query arrival process. An error probability $\varepsilon(s)$ is associated to each state $s \in \mathcal{S}_e$, and packets are instantaneously acknowledged by the receiver, so the sensor knows if the last transmitted packet was erased or correctly received. The Markovian model can fit several practical scenarios, including periodic communications or a channel with random outages, and is often used in wireless communications. We also make the simplifying assumption that the sensor can know the state of the channel through beacon messages. This is a simplifying hypothesis, and is not always true in sensor networks. However, having to rely on older estimates of the state of the channel would not change the fundamental nature of the model, and would only increase the uncertainty of the transmission.

The simplest case we can examine is the constant and always available channel, in which $\varepsilon(t) = \varepsilon$. A slightly more complex example is a deterministic and periodic error process. This models links that are available in a cyclical manner with period T_e, like the orbital passes of LEO communication satellites. In this case, the error probability is 1 when no satellite is visible and constant during a pass. In our simulations, we limit ourselves to deterministic and periodic error probability processes, whose value is known by the transmitter, but the formulation and solution are general. In our simulations, we consider the four use cases presented in the introduction, which can be simply mapped to different Markov chains for the query and channel error processes: deterministic queries follow a Markov chain with T_q states and forced transmissions, as do predictable channels (with the constant channel as a special case with $T_e = 1$). Stochastic query processes and channels follow general finite Markov chains, and in this case, any foresighted action relies on knowledge of the transition probabilities of the Markov model.

IV. Analytical example

We can now consider a simple example in which the difference between AoI and QAoI is clear, and in which we



(a) $\gamma = 1$ (100 for PQ). (b) $\gamma = 5$ (500 for PQ). (c) $\gamma = 10$ (1000 for PQ). (d) $\gamma = 20$ (2000 for PQ).

Fig. 4: Thresholds Δ^* for $T = 50$, $\varepsilon = 0.2$.

can derive the optimal strategies analytically. We assume that transmitting has a constant cost γ, that the channel has a constant error probability ε, and that the optimization is over a finite horizon T. The objective of the Permanent Query (PQ) system is then to minimize the average AoI during an episode, while the Query Arrival Process Aware (QAPA) system will try to minimize the AoI at the query instant, i.e., in slot T. We also consider the EAoI metric from [2], which results in a system similar to QAPA.

We then define a cost function for the two systems, which we call c:

$$c_{\text{PQ}}(t) = \gamma a_t + \Delta(t); \tag{7}$$

$$c_{\text{QAPA}}(t) = \gamma a_t + \delta(t-T)\Delta(T), \tag{8}$$

where $\delta(x)$ is the Dirac delta function. We can also consider a QAPA system which tries to minimize EAoI instead of QAoI, to show the difference between the two metrics:

$$c_{\text{EAoI}}(t) = \gamma a_t + \delta(t-T)\left[\delta(a_t)\Delta(T-1) + (1-\delta(a_t)\Delta(T)\right]. \tag{9}$$

We then define the long-term cost $C(\tau) = \sum_{t=\tau}^{T} c(t), \tau \leq T$. The optimal policy π^*_{PQ} is then the one that minimizes $C_{\text{PQ}}(\tau)$, and the same is true for the QAPA system. In the last step, with $t = T$, $c(T) = C(T)$, and both systems have the same expected cost if they start from the same conditions and take the same action:

$$\mathbb{E}[c(T)|a_t = a, \Delta(t-1) = d] = \begin{cases} \gamma + \varepsilon d + 1, & \text{if } a = 1; \\ d + 1, & \text{if } a = 0. \end{cases} \tag{10}$$

It is easy to see that transmitting in the last slot, i.e., setting $a_T = 1$, reduces the long-term cost if:

$$\Delta(T-1) > \frac{\gamma}{1-\varepsilon}. \tag{11}$$

We can then derive the optimal policies. In this case, any policy $\pi(d,t)$ depends only on the current age $\Delta(\tau-1) = d$ and the time slot index $\tau = t$. We only consider cases in which $\varepsilon > 0$, as the $\varepsilon = 0$ case is trivially optimized by transmitting only in the last slot before a query.

Theorem 1. *The optimal policies π^*_{PQ} and π^*_{QAPA} are threshold policies, i.e., $\pi^*_{PQ}(d,t) = 1 \Rightarrow \pi^*_{PQ}(d+1,t) = 1$ and $\pi^*_{QAPA}(d,t) = 1 \Rightarrow \pi^*_{QAPA}(d+1,t) = 1$.*

The proof of the theorem is given in the Appendix. The PQ policy is then defined by threshold values $\Delta^*_{\text{PQ}}(t)$, defined as:

$$\Delta^*_{\text{PQ}}(t) = \min\{d \in \mathbb{N} : \pi^*_{\text{PQ}}(d,t) = 1\}. \tag{12}$$

The same holds for the QAPA policy. The threshold for transmission can be computed recursively: if we know π^*_{PQ} from time $\tau+1$ to the end of the episode, we can compute $\Delta^*_{\text{PQ}}(\tau)$ using the following formula:

$$\Delta^*_{\text{PQ}}(\tau) = \inf\Big\{d \in \mathbb{N} : d + \mathbb{E}[C(\tau+1)|\pi^*, \Delta(\tau) = d] > \frac{\gamma}{1-\varepsilon} + \mathbb{E}[C(\tau+1)|\pi^*, \Delta(\tau) = 0]\Big\}, \tag{13}$$

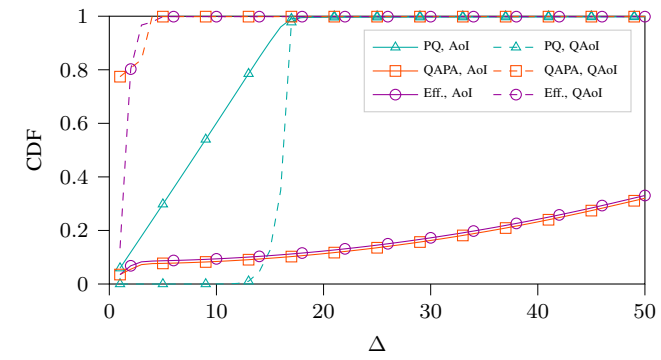
where $\mathbb{E}[C(\tau+1)|\pi^*, \Delta(\tau) = d]$ is the expected long-term cost while following the optimal policy. On the other hand, $\Delta^*_{\text{QAPA}}(\tau)$ is given by:

$$\Delta^*_{\text{QAPA}}(\tau) = \inf\Big\{d \in \mathbb{N} : \mathbb{E}[C(\tau+1)|\pi^*, \Delta(\tau) = d] > \frac{\gamma}{1-\varepsilon} + \mathbb{E}[C(\tau+1)|\pi^*, \Delta(\tau) = 0]\Big\}. \tag{14}$$

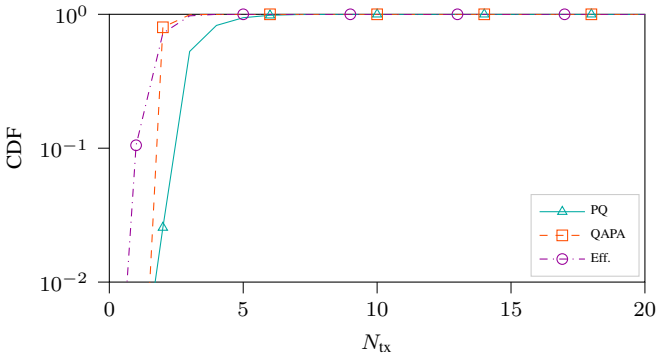
Naturally, we have $\Delta^*_{\text{PQ}}(T) = \Delta^*_{\text{QAPA}}(T) = \frac{\gamma}{1-\varepsilon}$, as given by (11).

Fig. 4 shows how the threshold strategies work in a system with $\varepsilon = 0.2$: while the PQ system exhibits some periodic behavior, mostly due to the effect of the finite horizon, it tends to transmit less if the episode is close to the end, as the future reduction of the AoI is limited in time. On the other hand, the QAPA sensor transmits only in the last steps before the query, but increasing the value of γ makes it less convenient to transmit, and the sensor will only do so for a progressively higher expected QAoI, and only closer to the actual query instant. If we use EAoI as a metric for the QAPA system, the threshold is not monotonic any more: the best moment to transmit is actually one step before the query, as transmitting in the same slot increases the delay in the query response. The transmission cost γ for PQ is higher than for both QAPA settings, because the overall cost is much higher, taking into account the AoI at every step and not just at the last step. The monotonicity of the strategy for QAoI, which holds over all our simulations in more complex systems, can make optimization easier, and a formulation without delayed responses can be simpler to implement for IoT gateways.

The performance of the systems is shown in Fig. 5, which shows the Cumulative Distribution Function (CDF) of the



(a) CDF of the AoI and QAoI.



(b) CDF of N_{tx}.

Fig. 5: Behavior of the PQ, QAPA, and EAoI systems with $T = 50$ and $\varepsilon = 0.2$, starting from a random age Δ_0 between 1 and 100.

AoI and QAoI, along with the number of transmissions in each episode N_{tx}: as we can see, the QAPA strategy has a much better QAoI, even though it transmits slightly less. The EAoI-based system has a slightly higher QAoI than the QAoI-oriented system, as it tends to transmit slightly earlier: however, it also has slightly fewer transmission attempts on average. For both systems, this comes at the cost of the AoI, which is significantly higher than for the PQ system. As we will see in the next sections, this basic pattern holds even for much more complex systems.

V. MDP Formulation and Problem Solution

In this section, we will consider the full problem, with a query process and energy dynamics. To understand the impact of the query process in the performance of a communication system, we will model the PQ and QAPA systems as MDPs, which we will then proceed to solve. A MDP is defined by a state space $\mathcal{S}$, an action space $\mathcal{A}$, a set of transition probabilities $p_a(s, s') = P(s_{t+1} = s' | a_t = a, s_t = s)$, and an instantaneous reward function $r(s, a, s')$, which represents the immediate reward when taking action a and transitioning from state s to state s'. The two systems, PQ and QAPA, can use the same state and action spaces, and only differ in the reward function that they use. This problem formulation is significantly more complex than the example from Sec. IV, making the direct derivation of strategies complex, but it can represent a more general class of communication systems.

To model the energy-constrained nature of the node, we use a *leaky bucket* model, as commonly done in the literature [32]: we consider a bucket of tokens, which is replenished by a process which can generate tokens independently at each step with probability μ_b. The node can only transmit a packet if there are tokens in the bucket, and each transmission consumes one token. This model can fit a general power consumption constraint on a battery-powered node, which should limit its number of transmissions in order to prolong its lifetime. Furthermore, it allows us to easily include the constraint in the MDP formulation as elaborated further in the next section.

Decisions are made by the sensor at every slot, as it can either keep silent or send a packet. Consequently, the action space is $\mathcal{A} = \{0, 1\}$. Both the PQ and the QAPA agents (i.e., sensors with two different objectives) need to know the current age $\Delta(t)$, as well as the state $s_e(t) \in \mathcal{S}_e$ of the error probability Markov process. Additionally, the agent should know the number of available tokens, $b(t)$, as it will influence its decision whether to transmit. If the number of tokens is 0, the sensor is blocked from transmitting until a token is generated. The tuple $(\Delta(t), s_e(t), b(t))$ is sufficient to represent the state in the PQ system, which does not require any knowledge of the query arrival process, while the QAPA system adds a fourth element to the state, i.e., the state of the query arrival process $s_q(t) \in \mathcal{S}_q$. As the PQ system can be studied as a special case of the QAPA system (with a single-state query arrival process), we adopt the wider definition for both systems to simplify the notation. We then define the state space as $\mathcal{S} = \mathbb{N}^2 \times \mathcal{S}_e \times \mathcal{S}_q$ and assume that the query arrival, token generation, and error probability processes are independent, examining each element of the state separately. The AoI increases by one between each slot unless the node decides to transmit and the packet is successfully received, with probability $p_s(t) = 1 - \varepsilon(s_e(t))$, in which case the AoI is reduced to one in the subsequent slot. The transition probabilities are thus described by

$$P(\Delta(t+1) = d | s_t, a_t) = \begin{cases} a_t p_s(t) & d = 1; \\ 1 - a_t p_s(t) & d = \Delta(t) + 1; \\ 0 & \text{otherwise}, \end{cases} \tag{15}$$

where a_t is the action at time t, which equals zero if the sensor is silent and one if it transmits. Secondly, the number of tokens in the next slot depends on whether a new token is generated and whether the sensor transmits, in which case it uses one token. The transition probability from $b(t)$ to $b(t+1)$ is:

$$P(b(t+1) = b + i | b(t), a_t) = \begin{cases} \mu_b & \text{if } i = 1 - a_t; \\ 1 - \mu_b & \text{if } i = -a_t; \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

The transition probabilities for the error probability and query arrival processes are defined by the matrices P_e and P_q, respectively. We assume that the query process is error-free, i.e., that the link between the ground station and the monitor is error-free.

We define two cost functions; one for the PQ system, which does not depend on the query instant and will be used as

baseline, and one for the QAPA system, in which the cost is only considered when a query arrives. In the baseline PQ model, the cost is given by the AoI in any slot:

$$c_{\text{PQ}}(s_t, a_t, s_{t+1}) = \Delta(t+1). \tag{17}$$

However, in the QAPA system, the cost is the AoI when a query arrives:

$$c_{\text{QAPA}}(s_t, a_t, s_{t+1}) = \begin{cases} \Delta(t+1) & \text{if } s_q(t+1) \in \mathcal{Q}; \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

In both cases, the objective is to find a policy π^* that minimizes the long-term cost. In this initial work, we limit ourselves to consider the discounted case, which benefits from strong convergence guarantees, and defer the case with undiscounted costs to future work. In this case, at least one optimal policy is guaranteed to exist as a stationary deterministic decision rule [33], i.e. $\pi^* : \mathcal{S} \to \mathcal{A}$. Specifically, we solve

$$\pi^* = \arg\min_{\pi} \left[\sum_{t=0}^{\infty} \lambda^t \sum_{s_t \in \mathcal{S}} P(s_t|s_0, \pi) c(s_t, \pi(s_t)) \right], \tag{19}$$

where $\lambda < 1$ is the discount factor, and $c(s_t, a_t) = \mathbb{E}_{s_{t+1}}[c(s_t, a_t, s_{t+1})|s_t, a_t]$ is the expected cost of taking action a_t in state s_t under either the PQ or QAPA model. Naturally, the energy constraint has a major impact on the cost, but it is implicit: as the sensor cannot transmit an update if it has no energy tokens, its age will increase, consequently increasing the long-term cost. In fact, we explicitly excluded a cost of transmission from the model, as the policy can already account for energy limitations by tuning its behavior and avoiding short-sighted choices that maximize the short-term reward while leading to long-term harm. The factor λ is crucial in this, as a higher discount value leads to a more foresighted policy.

A. Problem solution

We can now proceed to solve the MDP for the two systems we have defined using policy iteration, as described in [34, Ch. 4]. In order to apply the algorithm, we need to truncate the problem to a finite MDP. We do so by defining a maximum age $\Delta_{\max}$, a maximum query interval $T_{q,\max}$, and a token bucket size B: once the age, the query interval, or the number of tokens in the bucket reach the maximum, they cannot increase further. As long as the maximum values are sufficiently large, they are not reached during normal operation and this simplification does not affect the optimal policies or their performance.

The policy iteration algorithm has two steps, policy evaluation and policy improvement, which are repeated until convergence. The algorithm is initialized with a policy function π^0 and a value function v_π^0, which are both set to all zeros. At each step n, we can use the current estimate of the value of a state to update the next estimate, getting the value $u^n(s, a, s')$, defined as:

$$u^n(s, a, s') = c(s, \pi^n(s), s') + \lambda v_\pi^n(s'). \tag{20}$$

The iterative steps are then:

1) The policy is evaluated using
$$v_\pi^{n+1}(s) = \sum_{s' \in \mathcal{S}} p(s'|s, \pi^n(s)) u^n(s, a, s'), \tag{21}$$
for all s, where s is the current state, s' is the new state, a is the action, and c is the cost from either (17) or (18). The value function is an estimate of the long-term value that can be achieved in a given state using the policy.
2) The policy is improved by choosing the action that maximizes the long-term value, i.e., minimizes the long-term cost:
$$\pi^{n+1}(s) = \arg\min_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} p(s'|s, a) u^{n+1}(s, a, s'). \tag{22}$$

Policy iteration is guaranteed to converge to the optimal policy [35] in finite-state MDPs with finite reward. The complexity of policy iteration in general is exponential in the number of states, making it particularly impractical for realistic problems. However, if the correct pivoting rule is adopted and the discount factor is constant in time, policy iteration was shown to be strongly polynomial in [36]:

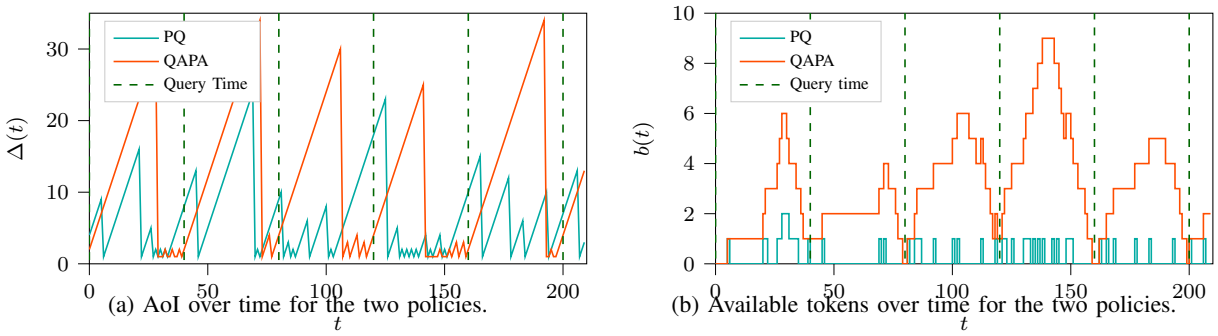
$$N_{\text{it}} \leq \frac{|\mathcal{S}|^2(|\mathcal{A}| - 1)}{1 - \lambda} \log\left(\frac{|\mathcal{S}|^2}{1 - \lambda}\right). \tag{23}$$

Each iteration uses at most $O(|\mathcal{A}||\mathcal{S}|^2)$ operations, making the resulting bound polynomial in the size of the MDP. As mentioned above, we truncated the age and token bucket size to make the MDP finite, so the conditions to use the algorithm apply. The notation in the past two sections is summarized in Table I. In our case, in which there are only two actions, policy iteration then converges in $O\left(\frac{|\mathcal{S}|^2}{1-\lambda} \log\left(\frac{|\mathcal{S}|^2}{1-\lambda}\right)\right)$ iterations, which correspond to $O\left(\frac{2|\mathcal{S}|^4}{1-\lambda} \log\left(\frac{|\mathcal{S}|^2}{1-\lambda}\right)\right)$ steps.

Naturally, policy iteration can suffer from the curse of dimensionality in more complex scenarios: as the number of states increases exponentially with the number of considered dimensions of the state, and policy iteration has itself an exponentially increasing complexity as the number of states increases, this solution is only practical for relatively small problems. In larger problems, learning-based techniques such as reinforcement learning can provide a far faster convergence, particularly when using deep reinforcement methods that can generalize experience [37]. These solutions have been successfully applied in the AoI optimization literature [25], and they can be applied directly to our MDP formulation, but we consider them as part of future work on this subject, solving some simpler examples with policy iteration.

VI. Simulation settings and results

This section presents Monte Carlo evaluations of the policies obtained using the MDP described in Section V. Although, the methods in Section V can be applied to any query process, throughout the evaluation we will consider queries that occur periodically, at a fixed time interval T_q. Furthermore, we truncate the MDP at a maximum age of $\Delta_{\max} = 100 \times T_q$ and a maximum token bucket size of $B = 10$, and we use a discount factor $\lambda = 0.95$. We use the term AoI to refer to the age at any time and QAoI for the age sampled at the query instants.



(a) AoI over time for the two policies.

(b) Available tokens over time for the two policies.

Fig. 6: AoI dynamics of the PQ and QAPA policies for $T_q = 40, \mu_b = 0.2, \varepsilon = 0.2$. The PQ policy generally has a lower AoI, but the QAPA policy minimizes the AoI at the query instants.

A. *Periodic queries with constant error probability*

We first consider the simplest scenario, in which the error probability is constant and the query arrival process is deterministic with period T_q. In this scenario, the error probability process only has one state, i.e., $|\mathcal{S}_e| = 1$, and the error probability is a constant value ε. The query arrival process is a deterministic Markov chain with T_q states, with $\mathcal{S}_q = \{1, \dots, T_q\}$. The transition probabilities are given by:

$$P_q(s_q, s_q') = \begin{cases} 1, & s_q < T_q \wedge s_q' = s_q + 1; \\ 1, & s_q = T_q \wedge s_q' = 1; \\ 0, & \text{otherwise.} \end{cases} \tag{24}$$

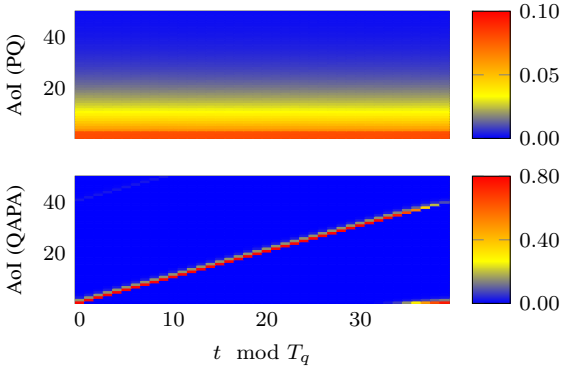
The subset of query states is given by $\mathcal{Q} = T_q$, i.e., the server sends a query only when the state reaches T_q, after which the state is reset to 1 and the counter starts increasing again. This Markov chain is equivalent to a periodic deterministic query process.

We start by exploring the temporal dynamics of the AoI process obtained using the PQ and the QAPA policies. Recall that PQ is optimized to achieve a low AoI independent of the query process, while QAPA minimizes the AoI at the query times, using cost functions (17) and (18), respectively. Fig. 6a shows the AoI for queries occurring periodically every $T_q = 40$ time slots as indicated by the vertical lines, a packet error probability of $\varepsilon = 0.2$, and a token rate $\mu_b = 0.2$. It is seen that the PQ policy reduces the AoI approximately uniformly across time, while the QAPA policy consistently tries to reduce the AoI in the slots immediately prior to a query, so that the AoI is minimized when the query arrives. This is reflected in Fig. 6b, which shows that the QAPA policy accumulates energy when the next query is far in the future, unlike PQ. A consequence of this is that the QAPA policy generally has a slightly higher average AoI than the PQ policy, but the QAoI of the QAPA is significantly lower than that of the PQ policy.

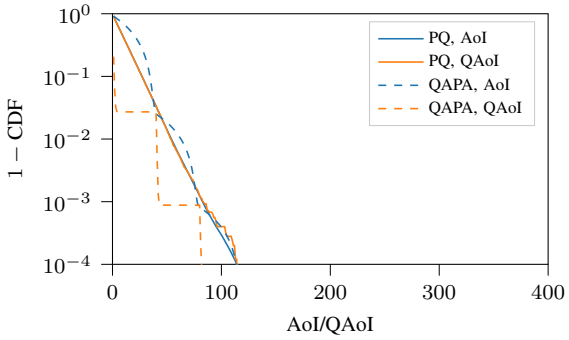
The initial observations from Fig. 6 can be confirmed by the distribution of the AoI as a function of the time since the last query, as illustrated in Fig. 7. Figs. 7a and 7c show the probability mass function (pmf) of the AoI conditioned on various time instants $t \mod T_q$, while Figs. 7b and 7d show the CDF of the overall AoI and QAoI. We can immediately see the difference between the two policies from Fig. 7a: the horizontal axis represents the time since the last query, while the colors represent the pmf of the AoI, whose domain is on the vertical axis. The AoI for the PQ policy does not depend on the time since the last query: the distribution is the same for all time instants, as can be seen by the fact that each horizontal line in the plot has exactly the same color. On the other hand, the QAPA policy shows a very different pattern: the AoI increases linearly as time passes, which indicates that the sensor does not send any packets in the first half of the interval, then sharply drops and stays very close to 0 in the final part of the interval. This behavior is consistent with what we would expect from a QAoI-oriented system, as sending packets long before the next query is basically a waste of energy, and transmissions are clustered just before the query instant. The resulting CDF in Fig. 7b reveals, as expected, that the AoI and the QAoI are equivalent for the PQ policy, as the distribution is the same at any time instant. However, for the QAPA policy, the QAoI is significantly lower than the AoI, while the AoI is often larger than the PQ policy's. This is due to the fact that the QAoI is only measured at the query instants, at which the age of the QAPA policy is minimized. Due to the energy constraint, this comes at the cost of a generally higher age, causing a higher AoI measured at each time instant. Finally, the staircase appearance in the CDF is due to the fact that the queries happen periodically. If the queries were arriving at variable (but known in advance) intervals, then the QAPA would still have lower QAoI than the PQ query, but its CDF would have a different shape.

The same observations apply for the scenario with high error probability, $\varepsilon = 0.7$, shown in Figs. 7c and 7d. Although the AoI and QAoI are higher due to the high packet error rate, the applied policies are similar. It is interesting to note that there is a significant probability of having a QAoI higher than 40 (which corresponds to T_q) in Fig. 7c, and that transmissions tend to be even more clustered towards the end of the query interval for the QAPA policy, indicating that the sensor will tend to save energy for future queries when it gets several tokens. Although there is a significant probability that the packet immediately prior to the query is lost, the AoI distribution at $t \mod T_q = 0$ is still concentrated close to one.

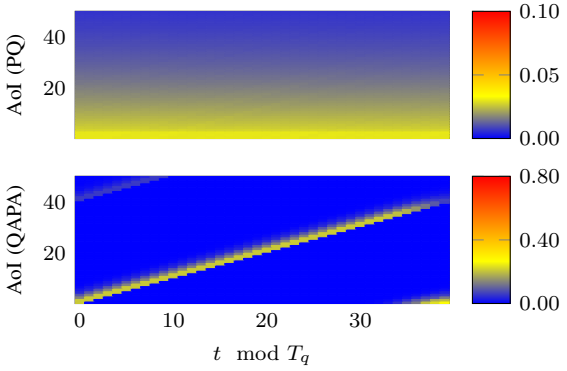
We now study how the average AoI and QAoI changes with the packet error probability ε for various choices of the parameters, shown in Fig. 8. For all cases, the QAPA policy achieves the lowest QAoI, while the PQ policy achieves the lowest AoI. When the query period, T_q, is low, the difference



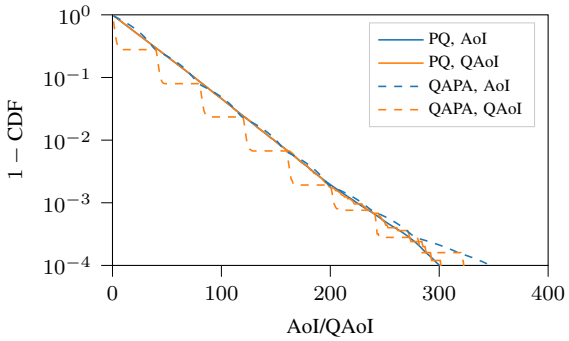
(a) PQ and QAPA AoI pmf over t_q, $\varepsilon = 0.2$.



(b) Complementary CDF, $\varepsilon = 0.2$.

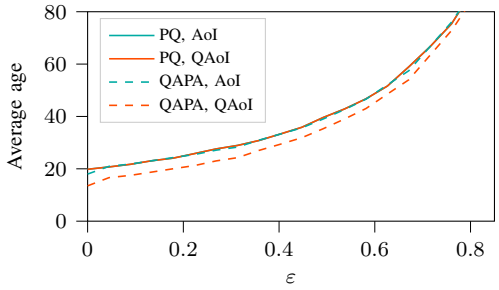


(c) PQ and QAPA AoI pmf over t_q, $\varepsilon = 0.7$.

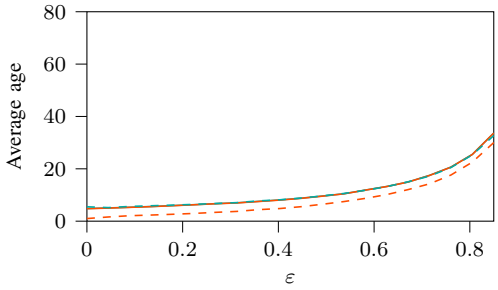


(d) Complementary CDF, $\varepsilon = 0.7$.

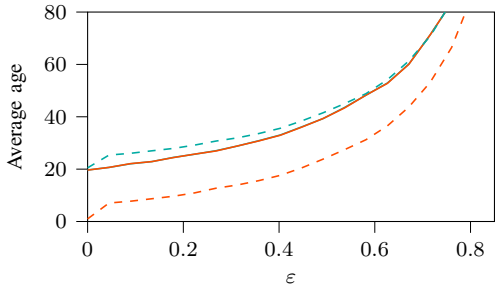
Fig. 7: AoI distributions and CCDFs for PQ and QAPA for $T_q = 40, \mu_b = 0.1$, and $\varepsilon = \{0.2, 0.7\}$.



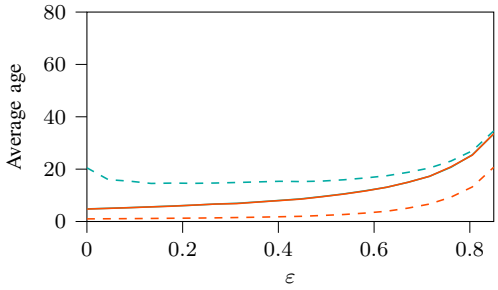
(a) Average age for $T_q = 10, \mu_b = 0.05$.



(b) Average age for $T_q = 10, \mu_b = 0.2$.



(c) Average age for $T_q = 40, \mu_b = 0.05$.

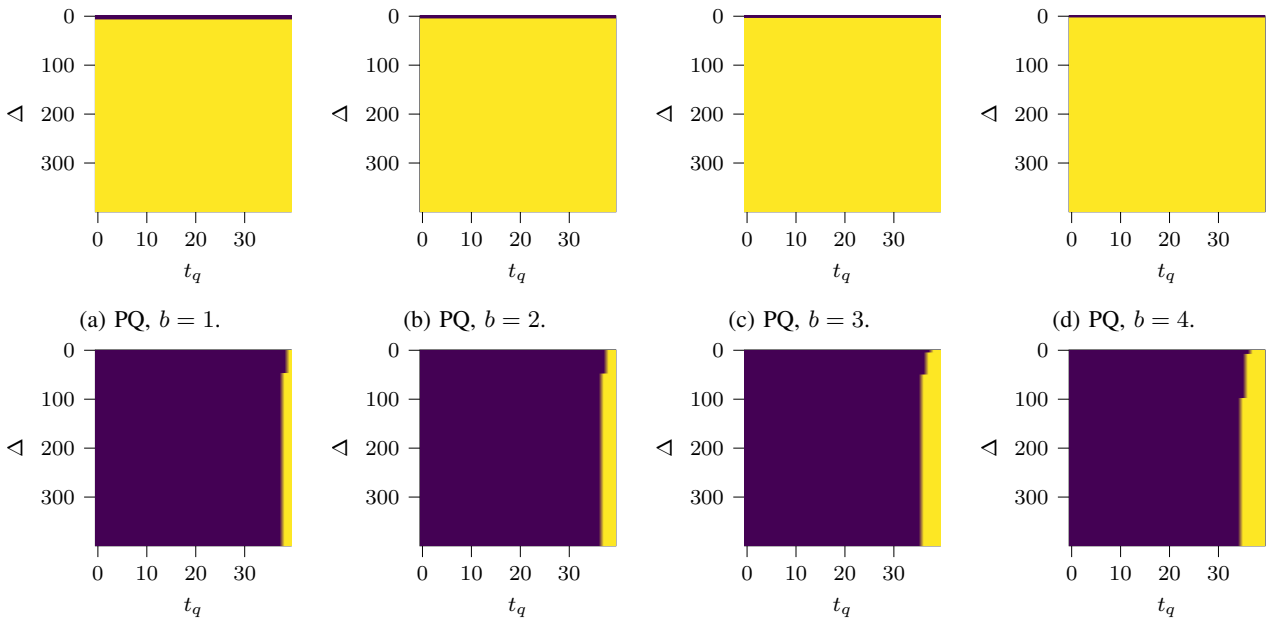


(d) Average age for $T_q = 40, \mu_b = 0.2$.

Fig. 8: Average AoI and QAoI for the two systems for different values of T_q and μ_b.

between AoI and QAoI is relatively small, as is the difference between the two policies. Intuitively, this is because the query instants, which are prioritized by the QAPA policy, are more frequent, making the two problems more similar.

As a result, awareness of the query arrival process becomes more important when queries are rare, i.e., when T_q is large: this is clear from the large gap between the average QAoI achieved by QAPA and by PQ in Fig. 8c and Fig. 8d. The plots on the left show the results for $\mu_b = 0.05$, i.e., when a new token is generated on average every 20 time slots. When $T_q = 10$ (see Fig. 8a), the token period becomes a limiting factor, and both the AoI and QAoI are relatively



(a) PQ, $b = 1$. (b) PQ, $b = 2$. (c) PQ, $b = 3$. (d) PQ, $b = 4$.

(e) QAPA, $b = 1$. (f) QAPA, $b = 2$. (g) QAPA, $b = 3$. (h) QAPA, $b = 4$.

Fig. 9: Strategies for the two systems with $\varepsilon = 0.2$. Yellow indicates a transmission, dark blue indicates no transmission.

high even for low values of ε. In particular, in the error-free case when $\varepsilon = 0$, the average QAoI cannot be lower than $(1 + 11)/2 = 6$, which is achieved by transmitting an update prior to every second query. Interestingly, the impact of the energy limit becomes less significant for the QAPA policy as the time between queries increases: by saving up tokens until right before the query, this policy can significantly reduce the QAoI, at the cost of a higher AoI. On the other hand, the PQ policy does not benefit from this increase, as it is oblivious of the query arrival frequency. When tokens are generated faster, at rate $\mu_b = 0.2$, as shown in the plots on the right side of the figure, the AoI and the QAoI are generally lower, since more frequent transmissions are allowed.

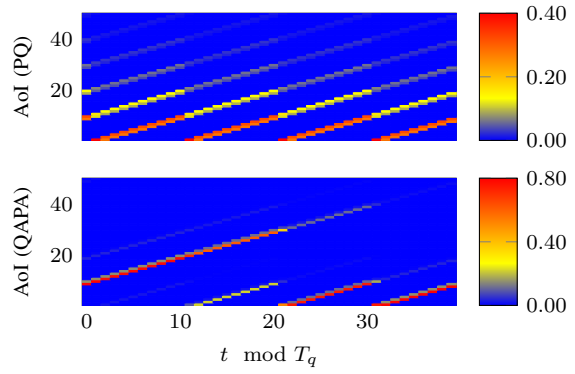
Finally, it is interesting to look at the strategies for the two systems, which are plotted in Fig. 9. As expected, the optimal strategy for the PQ system does not depend on the queries, only on the current AoI. If the AoI is very low, the PQ system waits for a while to transmit, but this is limited to a few slots (8 if $b = 1$, and just 4 if $b = 4$). On the other hand, while the PQ strategy plots are horizontal, the QAPA plots are almost vertical, i.e., the importance of the AoI is very low, and the system decides almost only based on the time until the next query. Naturally, if b is low, it only tries to transmit in the very last possible slot, to reduce the QAoI, while more tokens allow it to start earlier and protect itself from errors. If the AoI is high enough, it might be worth it to start transmitting a little earlier and hope to get a new token in the next slot, as the small increase in the QAoI if only the packet transmission in the earlier slot is successful is offset by the large decrease if the new token arrives and only that transmission is successful. If we compare the QAPA strategies with the threshold-based ones derived for the simplified system in Fig. 4, we see a significant similarity: the basic structure is still the same, although the precise values of the thresholds change. This result is encouraging for future attempts at deriving analytical threshold-based strategies for the full problem.

It is possible to imagine a simple threshold-based strategy that performs almost as well as the optimal strategy in this scenario: sending packets only if $b \geq T_q - t_q$ is a good approximation of the optimal strategy. Computing the threshold AoI over which it is convenient to send the first packet earlier is more complex, but still relatively easy in this case. More complex cases, such as the ones we will examine below, have a much wider state space with a higher dimensionality, making strategies difficult to visualize and hand-design.

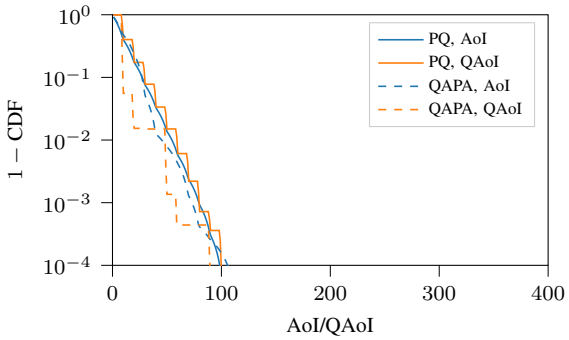
B. Periodic queries and error probability

We now analyze what happens when the error probability is not constant, but follows a periodic function. We consider the case of a LEO satellite connection which has limited availability due to constraints on the available constellation: connectivity is only available sporadically, depending on the periodic passes of the satellite over the transmitter node. In order to show the main trade-offs and represent a realistic case in which a LEO satellite passes over the transmitter at regular intervals, we consider a periodic error process with period T_e, in which the state space $\mathcal{S}_e = \{1, \dots, T_e\}$. The first two slots of each period are the only ones during which a transmission is possible, with an error probability ε_0, and correspond to the satellite pass. In all other slots, the transmission fails, as the transmitter is outside the satellite's coverage area. We then have $\varepsilon(1) = \varepsilon(2) = \varepsilon_0$, and $\varepsilon(s_e) = 1 \, \forall s_e \notin \{1, 2\}$. The transition probabilities for the error probability process are given by:

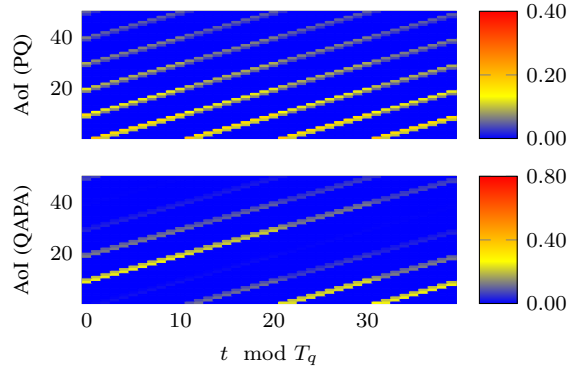
$$P_e(s_e, s'_e) = \begin{cases} 1, & s_e < T_e \wedge s'_e = s_e + 1; \\ 1, & s_e = T_e \wedge s'_e = 1; \\ 0, & \text{otherwise.} \end{cases} \tag{25}$$



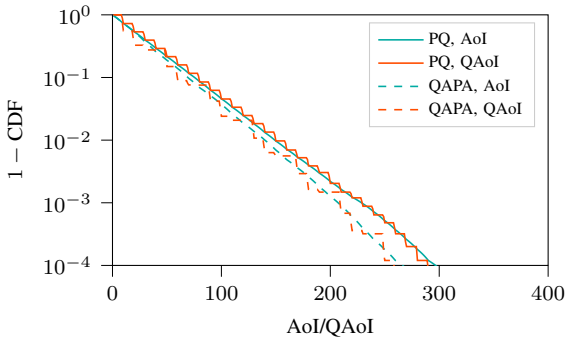
(a) PQ and QAPA AoI pmf over t_q, $\varepsilon = 0.2$.



(b) Complementary CDF, $\varepsilon = 0.2$.



(c) PQ and QAPA AoI pmf over t_q, $\varepsilon = 0.7$.



(d) Complementary CDF, $\varepsilon = 0.7$.

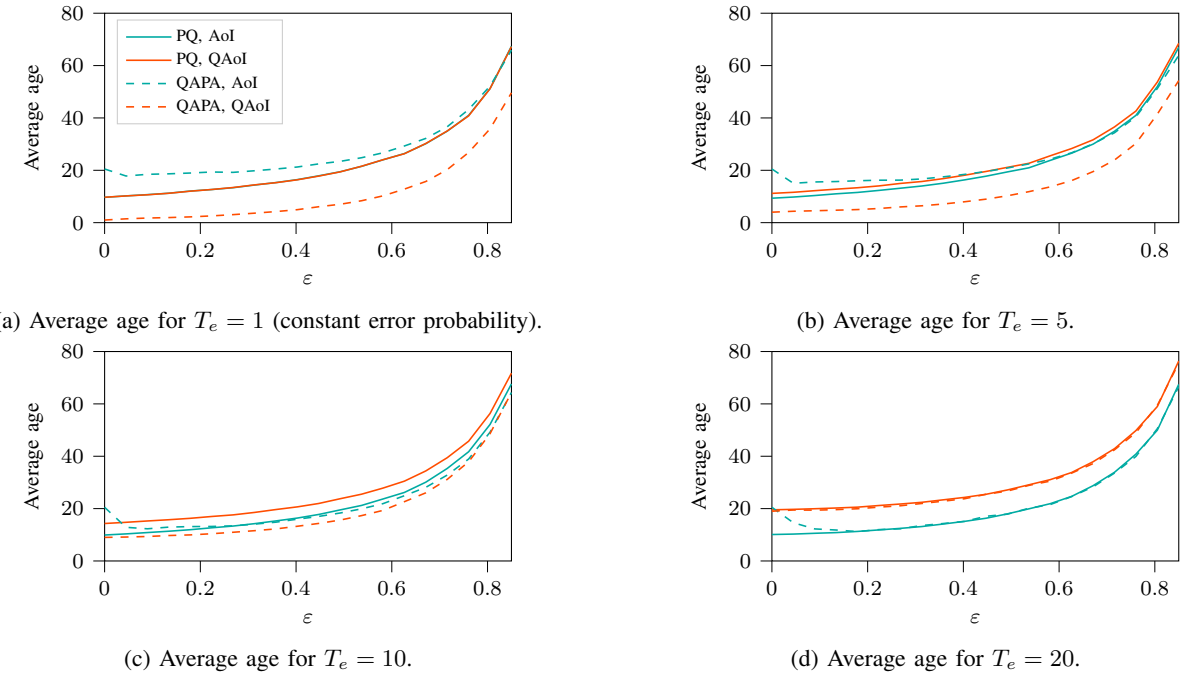
Fig. 10: AoI distributions and CCDFs for PQ and QAPA for $T_q = 40, \mu_b = 0.1$ and $\varepsilon = \{0.2, 0.7\}$.

We can analyze the behavior of the system as a function of the period T_e and the basic error probability ε_0. The query arrival process is defined as above, with transition probabilities given by (24). We first consider what happens in a simple case, setting $\mu_b = 0.1$, $T_e = 10$, and $T_q = 40$. Fig. 10b and Fig. 10d show how the PQ and QAPA system are restricted to the available transmission slots: outside of those slots, the AoI can only grow, resulting in the striped pattern on the plots. As expected, the QAPA system concentrates its effort in the transmission opportunities closer to a query, while the PQ system uses all available slots indiscriminately. This generates the difference in the QAoI seen in Fig. 10b and Fig. 10d: the QAPA system can maintain a lower QAoI, and the higher percentiles of the AoI, i.e., the tail of its distribution, are lower as well if the error probability is high.

We can then look at the effect of increasing the period of the satellite on the interplay between AoI and QAoI. We note here that we consider the worst possible scenario for the QAoI, i.e., the one in which the queries are synchronized with the satellite passes and each query arrives at the instant immediately before a satellite's pass. Fig. 11 shows the difference in the average age for $T_e = 1$ (i.e., the previous scenario with constant error probability), $T_e = 5$, $T_e = 10$, and $T_e = 20$, considering $\mu_b = 0.05$ and $T_q = 40$. In all cases, the average AoI of the PQ process is similar: since the most important limiting factor is the energy constraint, the average age is about 20 slots in the error-free case and follows a similar trend for all subfigures. By comparing Fig. 11a and Fig. 11d, it is clear that this is not true for QAoI: the effect of having transmissions only at the beginning of the period, at least $T_e - 2$ slots from the query, increases the average QAoI for the PQ process by approximately $T_e/2 - 1$ slots. As in the previous case, the QAPA system can improve the QAoI by paying a small cost in terms of AoI, but the difference between its QAoI and the PQ system's reduces as transmission opportunities become scarcer: by constraining the possible transmissions of the QAPA system to a few slots, we reduce the optimality gap of the traditional AoI maximization strategy. However, the difference between the two is still significant even for $T_e = 20$, as shown in Fig. 11d. It is also interesting to see that AoI and QAoI are not the same in this case, even for the PQ policy: this is due to the fact that, as T_q is a multiple of T_e, queries always come just before a transmission opportunity, so the QAoI is always at least $T_e - 1$, while there is no such offset for AoI, which is measured in all slots.

C. *Stochastic queries with periodic error probability*

We now examine a more general case, in which queries arrive at stochastic IID intervals with a known distribution and transmission opportunities are limited by satellite passes. The error probability process is then defined as above, with transition probabilities given by (25). On the other hand, the queries are modeled to arrive with uniformly distributed inter-query times, i.e., $t_{q,i+1} - t_{q,i} \sim \mathcal{U}\left(\frac{T_q}{2} + 1, T_q\right)$. This is a worst-case scenario for the QAPA system: as queries can arrive at a random instant over a wide range of values, the transmitter



(a) Average age for $T_e = 1$ (constant error probability).

(b) Average age for $T_e = 5$.

(c) Average age for $T_e = 10$.

(d) Average age for $T_e = 20$.

Fig. 11: Average AoI and QAoI for the two systems for different values of T_e, with $\mu_b = 0.1$ and $T_q = 40$.

needs to keep the AoI low almost at all times. We have $\mathcal{S}_q = \{1, \dots, T_q\}$ with these non-zero transition probabilities:

$$P_q(s_q, s_q') = \begin{cases} 1, & s_q \leq \frac{T_q}{2} \wedge s_q' = s_q + 1; \\ 1 - \frac{1}{T_q - s_q + 1}, & \frac{T_q}{2} < s_q < T_q \wedge s_q' = s_q + 1; \\ \frac{1}{T_q - s_q + 1}, & s_q > \frac{T_q}{2} \wedge s_q' = 1. \end{cases} \tag{26}$$

Fig. 12 shows that this is indeed a case in which knowledge of the query process is not critical: the age pmfs in Fig. 12a and Fig. 12c are almost the same for PQ and QAPA. In this case, the time since the last query has a limited value to the QAPA system, as it does not help much in predicting when a query will arrive. Consequently, the behavior of the QAPA system is much more similar to the PQ system's: the knowledge of the query arrival process statistics results in a very small gain, as the uniform interval distribution gives a limited amount of information on future steps. However, setting a uniform distribution for the interval still implies some memory in the process, since, e.g., knowing that there has been no query for $T_q - 1$ steps implies that a query will arrive in the next step with probability 1. The PQ and QAPA strategies would be absolutely identical if the query process was Poisson, i.e., memoryless: if queries were to follow a memoryless process, any instant would be as valuable as any other in terms of future QAoI. This is evident in Fig. 12b, which shows a negligible gain for the QAPA system in terms of QAoI, and even more in Fig. 12d.

The analysis of the average AoI and QAoI as a function of the error probability ε_0, shown in Fig. 13, shows that freedom of action and the precision of knowledge about the query arrival times are two factors that increase the gap between a naive PQ system and a query-aware QAPA one. This is intuitive, as limits to the available strategies can reduce gains, as can uncertainty about query arrival times. The more randomness is included in the system, and the more constrained the possible strategies become, the more QAoI looks exactly like AoI.

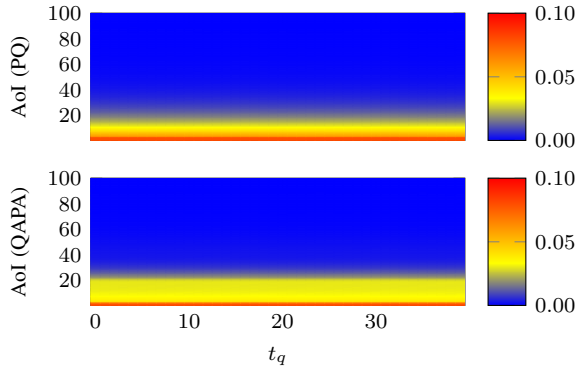
D. Stochastic queries with a Gilbert-Elliott channel

Finally, we consider another case, in which the channel does not have predefined transmission opportunities, but follows a stochastic Gilbert-Elliott [38], [39] model. In this model, we have two states, with $\varepsilon_1 = 0.2$ and $\varepsilon_2 = 0.7$. The transition probability matrix is given by:

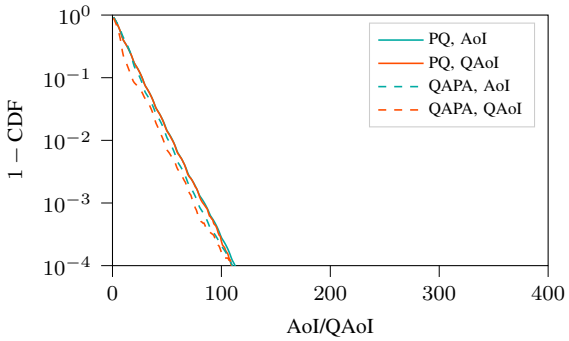
$$P_e(s_e, s_e') = \begin{pmatrix} 0.9 & 0.1 \\ 0.6 & 0.4 \end{pmatrix}. \tag{27}$$

On the other hand, the query process is given by the distribution in (26). In this case, both the channel and the query process are stochastic, and the amount of information available to the sensor is limited to the current state of each process and the transition probabilities.

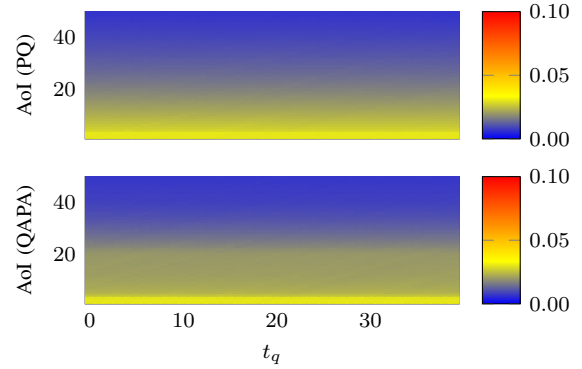
Fig. 14a shows the complementary CDF of the AoI and QAoI in this scenario: as this case is less restrictive than the one with periodic transmissions, in which a successful update was only possible once every 5 or more slots, the QAPA strategy performs significantly better in terms of QAoI. Surprisingly, it also performs better in terms of the tail of the AoI, although not on average: this might be because of the choice to save energy by not transmitting in the first 20 slots after each query, which have a probability 0 of having another query. Fig. 14b shows this behavior over time: while the PQ policy tends to uniformly transmit frequently, the QAPA policy lets the AoI grow when queries cannot arrive, then transmits more often to maintain the lowest possible QAoI for a potential query.



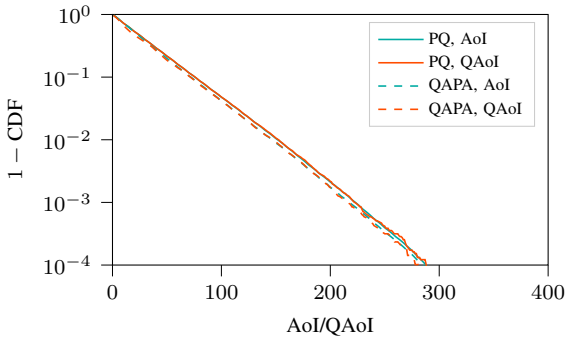
(a) PQ and QAPA AoI pmf over t_q, $\varepsilon = 0.2$.



(b) Complementary CDF, $\varepsilon = 0.2$.

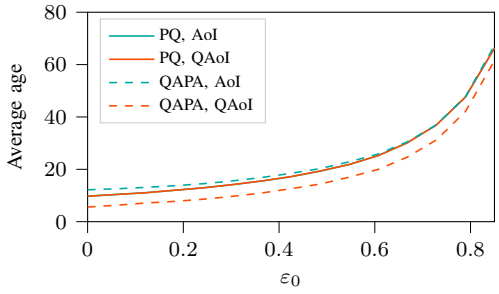


(c) PQ and QAPA AoI pmf over t_q, $\varepsilon = 0.7$.

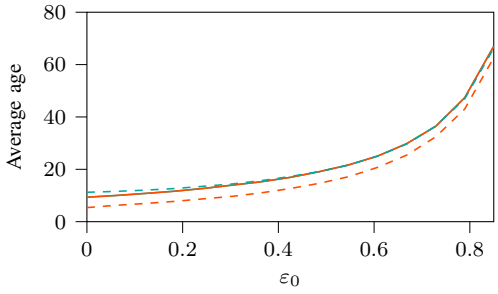


(d) Complementary CDF, $\varepsilon = 0.7$.

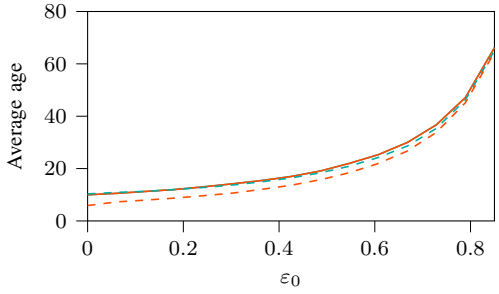
Fig. 12: AoI distributions and CCDFs for PQ and QAPA for $T_q = 40, \mu_b = 0.1, T_e = 10$, and $\varepsilon = \{0.2, 0.7\}$, with uniformly distributed query intervals over $\{21, 22, \ldots, 40\}$.



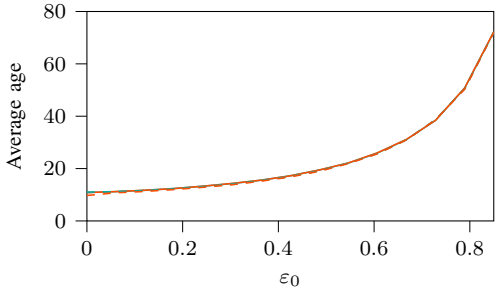
(a) Average age for $T_e = 1$ (constant error probability).



(b) Average age for $T_e = 5$.



(c) Average age for $T_e = 10$.

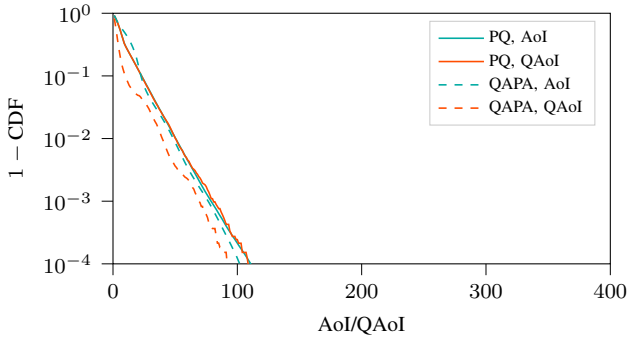


(d) Average age for $T_e = 20$.

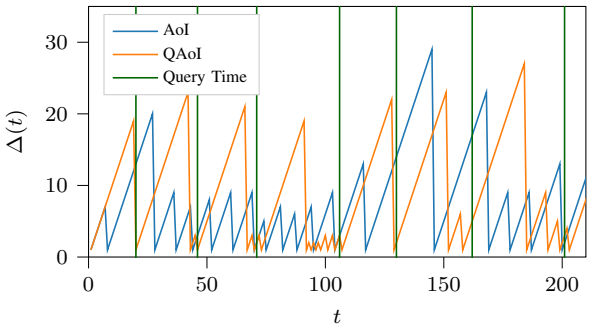
Fig. 13: Average AoI and QAoI for the two systems for different values of T_e, with $\mu_b = 0.1$, $T_q = 40$, and uniformly distributed query intervals over $\{21, 22, \ldots, 40\}$.

VII. Conclusions and Future Work

In this work, we have presented an optimization of QAoI, which takes the query arrival process and resource constraints on the communication into account. We showed that awareness of the query process can improve average and worst-case freshness in a variety of systems, modeling the single-source



(a) Complementary CDF of the AoI and QAoI.



(b) AoI over time for the two policies.

Fig. 14: Behavior of the two systems with stochastic queries and a Gilbert-Elliott channel.

scheduling problem as an MDP and finding the analytical solution. As AoI does not consider the specific features of applications, but reduces the age of any packet at any time, it cannot incorporate this additional information. The awareness of the query process can significantly improve the freshness as perceived by several monitoring application, adapting the scheduling to only transmit when it is most useful and avoid useless updates.

This work is a first step in considering the requirements of time-sensitive monitoring applications in resource-constrained communication scenarios: we see several avenues of possible future work, such as including the value of updates in the scheduling problem as well as their timing. Furthermore, the extension of the problem to more complex systems with multiple sources and realistic channel access can be an interesting direction of research, as there are several scenarios with one or more monitoring applications that need information from multiple sensors. In these more complex scenarios, policy iteration would be too complex due to the curse of dimensionality, and we foresee the application of reinforcement learning methods to find the optimal strategy for minimizing QAoI.

Appendix

In this Appendix, we prove that the threshold strategies defined in Sec. IV are optimal. In order to prove Theorem 1, we first introduce the expected long-term cost $\Gamma(t,a,d)$:

$$\Gamma(t,a,d) = \mathbb{E}[C(t)|a_t = a, \Delta(t-1) = d]. \tag{28}$$

We can similarly define $\Gamma(t,\pi,d)$:

$$\Gamma(t,\pi,d) = \mathbb{E}[C(t)|\pi, \Delta(t-1) = d]. \tag{29}$$

We then introduce the following lemma.

Lemma 1. *The expected long-term cost is a monotonically increasing function of the age Δ:*

$$\Gamma(t,a,d) \leq \Gamma(t,a,d+1), \ \forall a,d,t. \tag{30}$$

Proof. We prove this lemma by backward induction in t, working from $t=T$ and going backward. If we consider the base case with $t=T$, we can see from (10) that the condition holds. We now assume that the condition is from $t+1$ to $t=T$. First, we consider the case with $a_t = 0$. We have:

$$\Gamma_{\text{PQ}}(t,0,d) = d + \Gamma_{\text{PQ}}(t+1,\pi^*,d+1). \tag{31}$$

Since we know that $\Gamma_{\text{PQ}}(t+1,\pi^*,d+2) \geq \Gamma_{\text{PQ}}(t+1,\pi^*,d+1)$ due to the assumption in the inductive step, and $d+1 > d$, the lemma is proven for $a_t = 0$. If we consider the case in which $a_t = 1$, we get:

$$\begin{aligned}\Gamma_{\text{PQ}}(t,1,d) =& 1 + (1-\varepsilon)\Gamma_{\text{PQ}}(t+1,\pi^*,1) \\ &+ \varepsilon(d + \Gamma_{\text{PQ}}(t+1,\pi^*,d+1)).\end{aligned} \tag{32}$$

It is easy to see how this case also respects the condition, as d appears twice. All elements of the cost are then the same or higher for $\Delta(t-1) = d+1$ and the lemma is proven by induction. The same procedure can be repeated for QAPA, using either QAoI or EAoI as a metric. □

As the expected cost is a monotonically increasing function of the current age, we can immediately prove Theorem 1.

Proof of Theorem 1. As for Lemma 1, we can use backward induction starting from $t=T$ to prove the theorem. The base case is trivially true, as the decision is given by (11). We can now attempt to perform the inductive step *ad absurdum*, by assuming that the theorem is true from $t+1$ to T. Suppose that the optimal policy is not a threshold policy, i.e., $\exists d : \pi^*(d,t) = 1 \wedge \pi^*(d+1,t) = 0$. In this case, we know that, since $\pi^*(d,t) = 1$, we get:

$$\Gamma_{\text{PQ}}(t,1,d) < \Gamma_{\text{PQ}}(t,0,d). \tag{33}$$

We can then take the components of the reward:

$$\begin{aligned}1 + d + \Gamma_{\text{PQ}}(t+1,\pi^*,d+1) >& (1-\varepsilon)\Gamma_{\text{PQ}}(t+1,\pi^*,1) \\ &+1 + \gamma + \varepsilon(d + \Gamma_{\text{PQ}}(t+1,\pi^*,d+1)),\end{aligned} \tag{34}$$

which yields:

$$d + \Gamma_{\text{PQ}}(t+1,\pi^*,d+1) - \Gamma_{\text{PQ}}(t+1,\pi^*,1) > \frac{\gamma}{1-\varepsilon}. \tag{35}$$

On the other hand, since we assumed that $\pi^*(d+1,t) = 0$, we can follow the same procedure to get:

$$d+1+\Gamma_{\text{PQ}}(t+1,\pi^*,d+2) - \Gamma_{\text{PQ}}(t+1,\pi^*,1) \leq \frac{\gamma}{1-\varepsilon}. \tag{36}$$

Since $C(t)$ is non-negative, and we know from Lemma 1 that $\Gamma_{\text{PQ}}(t+1,\pi^*,d+1) \leq \Gamma_{\text{PQ}}(t+1,\pi^*,d+2)$, the results in (35) and (36) are in contradiction, and $\pi^*(d,t) = 1 \Rightarrow \pi^*(d+n,t) = 1, \forall n \in \mathbb{N}$. As before, the same procedure can be repeated for QAPA, which does not have the term d in the long-term cost, but still yields the same inequality. □

REFERENCES

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

[2] B. Yin, S. Zhang, Y. Cheng, L. X. Cai, Z. Jiang, S. Zhou, and Z. Niu, "Only those requested count: Proactive scheduling policies for minimizing effective age-of-information," in *Conf. on Computer Communications (INFOCOM)*. IEEE, Apr. 2019, pp. 109–117.

[3] B. Soret, S. Ravikanti, and P. Popovski, "Latency and timeliness in multi-hop satellite networks," in *Int. Conf. on Communications (ICC)*. IEEE, Jun. 2020.

[4] D. Li, S. Wu, Y. Wang, J. Jiao, and Q. Zhang, "Age-optimal HARQ design for freshness-critical satellite-IoT systems," *IEEE Internet of Things Journ.*, vol. 7, no. 3, pp. 2066–2076, Dec. 2019.

[5] J. Holm, A. E. Kalør, F. Chiariotti, B. Soret, S. K. Jensen, T. B. Pedersen, and P. Popovski, "Freshness on demand: Optimizing Age of Information for the query process," in *Int. Communications Conf. (ICC)*. IEEE, Jun. 2021.

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

[7] A. M. Bedewy, Y. Sun, and N. B. Shroff, "Age-optimal information updates in multihop networks," in *Int. Symp. on Information Theory (ISIT)*. IEEE, Jun. 2017, pp. 576–580.

[8] A. M. Bedewy, Y. Sun, and N. B. Shroff, "The age of information in multihop networks," *IEEE/ACM Trans. on Networking*, vol. 27, no. 3, pp. 1248–1257, Jun. 2019.

[9] B. Wang, S. Feng, and J. Yang, "To skip or to switch? minimizing age of information under link capacity constraint," in *19th Int. Worksh. on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE, Jun. 2018.

[10] K. Chen and L. Huang, "Age-of-information in the presence of error," in *Int. Symp. on Information Theory (ISIT)*. IEEE, Jul. 2016, pp. 2579–2583.

[11] R. Devassy, G. Durisi, G. C. Ferrante, O. Simeone, and E. Uysal, "Reliable transmission of short packets through queues and noisy channels under latency and peak-age violation guarantees," *IEEE Journ. on Selected Areas in Communications*, vol. 37, no. 4, pp. 721–734, Feb. 2019.

[12] H. B. Beytur, S. Baghaee, and E. Uysal, "Measuring age of information on real-life connections," in *27th Signal Processing and Communications Applications Conf. (SIU)*. IEEE, Apr. 2019.

[13] I. Kadota and E. Modiano, "Minimizing the age of information in wireless networks with stochastic arrivals," *IEEE Trans. on Mobile Computing*, Dec. 2019.

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

[15] R. D. Yates, "Lazy is timely: Status updates by an energy harvesting source," in *Int. Symp. on Information Theory (ISIT)*. IEEE, Jun. 2015, pp. 3008–3012.

[16] R. D. Yates and S. K. Kaul, "Status updates over unreliable multiaccess channels," in *Int. Symp. on Information Theory (ISIT)*. IEEE, Jun. 2017, pp. 331–335.

[17] R. D. Yates and S. K. Kaul, "Age of information in uncoordinated unslotted updating," in *Int. Symp. on Information Theory (ISIT)*. IEEE, Jun. 2020, pp. 1759–1764.

[18] R. Talak, S. Karaman, and E. Modiano, "Distributed scheduling algorithms for optimizing information freshness in wireless networks," in *19th Int. Worksh. on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE, Jun. 2018.

[19] X. Chen, K. Gatsis, H. Hassani, and S. S. Bidokhti, "Age of information in random access channels," in *Int. Symp. on Information Theory (ISIT)*. IEEE, Jun. 2020, pp. 1770–1775.

[20] R. Talak, S. Karaman, and E. Modiano, "Optimizing information freshness in wireless networks under general interference constraints," *IEEE/ACM Trans. on Networking*, vol. 28, no. 1, pp. 15–28, Dec. 2019.

[21] B. T. Bacinoglu, Y. Sun, E. Uysal-Bivikoglu, and V. Mutlu, "Achieving the age-energy tradeoff with a finite-battery energy harvesting source," in *Int. Symp. on Information Theory (ISIT)*. IEEE, Jun. 2018, pp. 876–880.

[22] X. Wu, J. Yang, and J. Wu, "Optimal status update for age of information minimization with an energy harvesting source," *IEEE Trans. on Green Communications and Networking*, vol. 2, no. 1, pp. 193–204, Nov. 2017.

[23] Y. Gu, H. Chen, Y. Zhou, Y. Li, and B. Vucetic, "Timely status update in Internet of Things monitoring systems: An age-energy tradeoff," *IEEE Internet of Things Journ.*, vol. 6, no. 3, pp. 5324–5335, Feb. 2019.

[24] S. Feng and J. Yang, "Optimal status updating for an energy harvesting sensor with a noisy channel," in *Conf. on Computer Communications Worksh. (INFOCOM)*. IEEE, Apr. 2018, pp. 348–353.

[25] E. T. Ceran, D. Gündüz, and A. György, "Reinforcement learning to minimize age of information with an energy harvesting sensor with HARQ and sensing cost," in *Conf. on Computer Communications Worksh. (INFOCOM)*. IEEE, Apr. 2019, pp. 656–661.

[26] S. Zhang, H. Zhang, Z. Han, H. V. Poor, and L. Song, "Age of information in a cellular internet of UAVs: Sensing and communication trade-off design," *IEEE Trans. on Wireless Communications*, vol. 19, no. 10, pp. 6578–6592, Jun. 2020.

[27] Y. Sang, B. Li, and B. Ji, "The power of waiting for more than one response in minimizing the age-of-information," in *Global Communications Conf. (GLOBECOM)*. IEEE, Dec. 2017.

[28] F. Li, Y. Sang, Z. Liu, B. Li, H. Wu, and B. Ji, "Waiting but not aging: Optimizing information freshness under the pull model," *IEEE/ACM Trans. on Networking*, pp. 1–14, Dec. 2020.

[29] M. Hatami, M. Jahandideh, M. Leinonen, and M. Codreanu, "Age-aware status update control for energy harvesting IoT sensors via reinforcement learning," in *31st Ann. Int. Symp. on Personal, Indoor and Mobile Radio Communications (PIMRC)*. IEEE, Aug. 2020.

[30] M. Hatami, M. Leinonen, and M. Codreanu, "AoI minimization in status update control with energy harvesting sensors," *IEEE Trans. on Communications*, Sep. 2021.

[31] C. Xu, X. Wang, H. H. Yang, H. Sun, and T. Q. Quek, "AoI and energy consumption oriented dynamic status updating in caching enabled IoT networks," in *Conf. on Computer Communications Worksh. (INFOCOM)*. IEEE, Jul. 2020, pp. 710–715.

[32] V. Raghunathan, S. Ganeriwal, M. Srivastava, and C. Schurgers, "Energy efficient wireless packet scheduling and fair queuing," *ACM Trans. on Embedded Computing Systems (TECS)*, vol. 3, no. 1, pp. 3–23, Feb. 2004.

[33] R. Bellman, "A Markovian decision process," *Journ. of Mathematics and Mechanics*, vol. 6, no. 5, pp. 679–684, Jan. 1957.

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

[35] R. A. Howard, *Dynamic programming and Markov processes*. John Wiley, 1960.

[36] Y. Ye, "The simplex and policy-iteration methods are strongly polynomial for the markov decision problem with a fixed discount rate," *Mathematics of Operations Research*, vol. 36, no. 4, pp. 593–603, Nov. 2011.

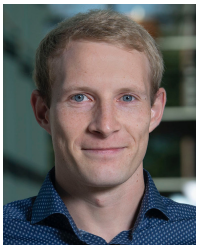
[37] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[38] E. N. Gilbert, "Capacity of a burst-noise channel," *Bell System Technical Journ.*, vol. 39, no. 5, pp. 1253–1265, Sep. 1960.

[39] E. O. Elliott, "Estimates of error rates for codes on burst-noise channels," *Bell System Technical Journ.*, vol. 42, no. 5, pp. 1977–1997, Sep. 1963.

Federico Chiariotti (S'15–M'19) is currently a post-doctoral researcher at the Department of Electronic Systems, Aalborg University, Denmark. He received his Ph.D. in information engineering in 2019 from the University of Padova, Italy. He received the bachelor's and master's degrees in telecommunication engineering (both *cum laude*) from the University of Padova, in 2013 and 2015, respectively. He has authored over 40 published papers on wireless networks and the use of artificial intelligence techniques to improve their performance. He was a recipient of the Best Paper Award at several conferences, including the IEEE INFOCOM 2020 WCNEE Workshop. His current research interests include network applications of machine learning, transport layer protocols, Smart Cities, bike sharing system optimization, and adaptive video streaming.

Josefine Holm received her B.Sc and M.Sc. degrees in mathematical engineering from Aalborg University in 2016 and 2018, respectively. She is currently pursuing her Ph.D. degree at the Connectivity Section at Aalborg University. Her research interests include wireless communication and IoT networks.



Anders E. Kalør (S'17) received the B.Sc. degree in computer engineering and the M.Sc. degree in networks and distributed systems from Aalborg University, Denmark, in 2015 and 2017, respectively. He is currently pursuing a Ph.D. degree in the area of wireless communications and networking at the Connectivity section at Aalborg University. In 2017, he was a visiting student at Robert Bosch, Germany, and in 2020 at King's College London, UK. His research interests include communication theory, MAC layer design for wireless systems, and networking.

Beatriz Soret (M'11) received her M.Sc. and Ph.D. degrees in Telecommunications from the University of Malaga, Spain, in 2002 and 2010, respectively. She is currently an associate professor at the Department of Electronic Systems, Aalborg University, and a Senior Research Fellow at the Communications Engineering Department, University of Malaga. Her current research interests include semantic communications and AoI, LEO satellite communications, and intelligent IoT environments.

Søren K. Jensen is a postdoctoral researcher at the Center for Data-Intensive Systems (Daisy) at the Department of Computer Science, Aalborg University, Denmark. His research interests span multiple areas of Computer Science, including programming language design, compiler design and implementation, developer tooling, parallel and distributed computing, big data, database management systems, data warehousing, and extract-transform-load processes.

Torben B. Pedersen is a professor with the Center for Data-Intensive Systems (Daisy), Aalborg University, Denmark. His research concerns Predictive, Prescriptive, and Extreme-Scale Data Analytics with Digital Energy as the main application area. He is an ACM Distinguished Scientist, a Senior Member of the IEEE, a member of the Danish Academy of Technical Sciences, and holds an honorary doctorate from TU Dresden.

Petar Popovski (S'97–A'98–M'04–SM'10–F'16) is a Professor at Aalborg University, where he heads the section on Connectivity and a Visiting Excellence Chair at the University of Bremen. He received his Dipl.-Ing and M. Sc. degrees in communication engineering from the University of Sts. Cyril and Methodius in Skopje and the Ph.D. degree from Aalborg University in 2005. He is a Fellow of the IEEE. He received an ERC Consolidator Grant (2015), the Danish Elite Researcher award (2016), IEEE Fred W. Ellersick prize (2016), IEEE Stephen O. Rice prize (2018), Technical Achievement Award from the IEEE Technical Committee on Smart Grid Communications (2019), the Danish Telecommunication Prize (2020) and Villum Investigator Grant (2021). He is a Member at Large at the Board of Governors in IEEE Communication Society, Vice-Chair of the IEEE Communication Theory Technical Committee and IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING. He is currently an Editor-in-Chief of IEEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS. Prof. Popovski was the General Chair for IEEE SmartGridComm 2018 and IEEE Communication Theory Workshop 2019. His research interests are in the area of wireless communication and communication theory. He authored the book "Wireless Connectivity: An Intuitive and Fundamental Guide", published by Wiley in 2020.