Scheduling "Last Minute" Updates for Timely Decision-Making

Jean Abou Rahal[*]
Chandra Family Department of Electrical and Computer Engineering, The University of Texas at Austin
Austin, TX, USA
jeanabourahal@utexas.edu

Gustavo de Veciana
Chandra Family Department of Electrical and Computer Engineering, The University of Texas at Austin
Austin, TX, USA
deveciana@utexas.edu

ABSTRACT

We consider a setting where requests for updates regarding time-varying processes are required prior to making a sequence of decisions. Each request has a finite length time window during which the update should be received. The end of the window reflects the time at which a decision is to be made, while the start of the window models the earliest possible time at which a useful update could be sent. An update scheduled as near to the end of the window as possible is deemed the best, i.e., reflects the most timely information about the process' state. This is modelled by a reward depending on the time difference between the decision point and the last scheduled update. Requests arrive arbitrarily and share a limited communication resource, e.g., a single request can be scheduled per time slot, hence not all decisions can be based on the latest possible update. We consider update scheduling policies which maximize the overall reward rate. In particular we consider an adversarial request model and evaluate proposed algorithms via their Competitive Ratio (CR). Specifically, we first derive a lower bound on the CR of any causal policy. We then propose two scheduling policies, denoted adversarial and greedy, and provide further analysis and insights on regimes where one might be superior to the other. We validate these observations via simulation for a setting with stochastic arrivals.

CCS CONCEPTS

• **Networks** → **Network algorithms**.

KEYWORDS

Age-of-Information, Adversarial Scheduling, Competitive-Ratio

ACM Reference Format:
Jean Abou Rahal and Gustavo de Veciana. 2023. Scheduling "Last Minute" Updates for Timely Decision-Making. In *Proceedings of the Int'l ACM Conference on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM '23), October 30-November 3, 2023, Montreal, QC, Canada.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3616388.3617518

[*]Contact person for correspondence

MSWiM '23, October 30-November 3, 2023, Montreal, QC, Canada
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0366-9/23/10.
https://doi.org/10.1145/3616388.3617518

1 INTRODUCTION

The emergence of applications relying on networked systems has revolutionized the sensing industry and led the way towards modeling systems that rely on the timely sharing of information to support real-time decision making. Among these, a challenging set of examples is tied to automated vehicles, robots, UAVs, etc., that are constantly traveling through complex environments and requiring updates to smoothly navigate with a high degree of situational awareness. Such applications are often best supported when updates are delivered right on time. In the vehicular setting for example, cars driving at different speeds and heading towards an obstructed intersection may express interest in accurate information on the state of the intersection right before they reach it and thus generate requests for timely updates about such intersections. Sending an update to a vehicle early on may not accurately represent the state of the intersection by the time it gets there and will lead to poor decisions. On the other hand, scheduling an update transmission to the vehicle when it is close to the intersection is likely to be advantageous and results in better decisions.

A major challenge in such systems is the dynamic aspect of requests for timely updates. Requests may not only arrive arbitrarily but may also be short-lived, i.e., such that one can receive updates only for a finite length time window before making a critical decision at the end of the window preferably based on the freshest information update.

A key step in this direction is to define appropriate metrics that capture the freshness and timeliness of the last received update to ensure that it accurately represents the state of the time-varying process that the request is interested in. To that end, the Age of Information (AoI) has been proposed as a metric that measures the freshness of the updates at the receiver [6, 7, 17]. In contrast to prior work on AoI, we propose a novel setting where a request can only receive updates within a finite time window, which we refer to as the request's active window. The importance of scheduling an update transmission to a request close to the end of its active window is modelled through a reward function which depends on the time difference between the end of the request's active window and the time at which the last update was received. Such a reward model is therefore tied to the age of the last received update within the request's active window which reflects the freshness of this update. In this paper, we study scheduling policies that aim to optimize the rewards associated with scheduling such updates.

Related work. There has been substantial work on the scheduling of requests with deadlines. The Earliest Deadline First (EDF) policy [12] is the most well-known policy for scheduling in real-time systems. It was proved to maximize the fraction of customers served prior to their respective deadlines when the service time is equal to a single slot. EDF requires that the customer with the earliest deadline be scheduled first and at most once. Meanwhile [15] considers a real time queuing system where packets have deadlines and the processing time of a packet is known upon its arrival. A predetermined fixed reward is associated with servicing each packet and the goal is hence to design a scheduling policy that maximizes the cumulative reward. Other works, e.g., [11], [5], introduce scheduling policies in systems with strict bounds on the service delays. In contrast to prior work, we allow a request to be scheduled more than once before the end of its active window, and we define a reward function that is tied to the time difference between the request's decision time and its last scheduled update.

More recent work addresses scheduling in collaborative sensing settings in an attempt to achieve real-time situational awareness and can be found in [1, 2, 13, 19]. We extend on the prior works by assuming that requests for timely updates arrive arbitrarily and can only receive updates within a limited period of time before making a critical decision. Other recent work investigate scheduling sensing nodes to update a remote node under communication constraints with requirements on the AoI [3, 4, 8–10]. They consider applications where the AoI has to meet some freshness threshold, i.e., they impose either a hard or soft upper bound on the worst case AoI that can be achieved by any sensing node and devise policies that can schedule at most one node per slot to satisfy the constraints. On the other hand, [16] and [14] consider the setting where packets arrive arbitrarily over time and the algorithms only have access to information about packet arrivals. In particular, [16] devises a policy to minimize the energy consumption under the peak AoI constraint at all times, while [14] introduces a resource allocation problem that captures the trade-off between AoI, quality and energy associated with packet transmission and proposed a policy to minimize the three costs. Finally, [18] develops and implements a scheduling algorithm that enables the customization of WiFi networks to the needs of time-sensitive applications. They propose a scheduling approach which makes use of the most up-to-date data as opposed to all past sampled data points, similarly to the one suggested in our work. We differentiate ourselves from [18] by considering a setting where requests arrive arbitrarily over time, as opposed to having a fixed number of nodes requesting information, and additionally consider that the requests can receive updates only in a short period of time before making decisions based on the last received update.

Contributions. We explore a new class of scheduling problems associated with delivering information updates "just in time". We consider a setting where requests for timely updates arrive arbitrarily and we assume that requests can receive updates within a finite-time window, which we refer to as the request's active window. Our goal is to ensure that requests have updates that are as fresh as possible by the end of their active windows to enable accurate decisions to be made based on the states of the time-varying processes they are interested in. We hence define a reward function that captures the importance of scheduling an update transmission to a request as close to the end of its active window as possible. We propose to maximize the reward rate under the assumption that only a single request can be scheduled at a time. In this setting, we investigate an adversarial setting where the number of new requests' arrivals as well as the length of a request's active window are unknown a-priori and only revealed once requests arrive, and thus use the competitive ratio as our performance metric. We derive a lower bound on the reward rate achieved by any non-idling causal policy. We then propose two causal scheduling policies π^a and π^g, referred to as the adversarial and greedy policies respectively and further derive the competitive ratio of π^g with respect to the optimal genie-based policy. Finally, we validate our theoretical analysis with numerical evaluations.

2 SYSTEM MODEL

2.1 Model for timely information requests

Consider requests for timely updates about time-varying processes that arrive arbitrarily to a time-slotted system. We let $\boldsymbol{\rho} = (\rho_i)_{i\in\mathbb{N}}$ denote the sequence of request arrivals, where the tuple $\rho_i = (a_i, s_i, e_i)$ is the i^{th} request, characterized by,

- a_i: Arrival time of request i,
- $s_i \geq a_i$: Release (or start) time of request i, which reflects the earliest time after which updates about the time-varying process become relevant to i,
- e_i: End time after which request i is no longer active,
- $[s_i, e_i]$: Active window within which updates in response to request i are permissible.

In particular, we let $\boldsymbol{\rho}_T$ denote the truncated sequence of requests that have end times prior to T. Updates scheduled for request i outside of $[s_i, e_i]$ have no value to i. We point out that one or more updates for a request can be scheduled while it is active. However, requests end up using only the most recent update they received within their active windows. It follows that an update scheduled closer to the start to service time of a request may age by the end of the active window and become stale and not accurately reflect the actual status of the time-varying process. Therefore, scheduling an update for active request i in a slot close to e_i is more beneficial than scheduling an update in a slot close to s_i. We further let $w_i = e_i - s_i + 1$ be the length of request i's active window and $w_{\max}$ be an upper bound on the length of the active window, i.e., for all i we have that $1 \leq w_i \leq w_{\max}$.

2.2 Scheduling updates

We consider a time-slotted system where the length of a time slot corresponds to the duration it takes to transmit an update. Further, we consider for simplicity a setting where a policy can schedule a single update per time slot. That said, recall that multiple updates can be scheduled sequentially for the same request within its active window. Below, we formally introduce the notation to be used in this paper.

Definition 1. ***(Servicing a request)*** *We say that a policy π has serviced a request i if i is no longer active and π scheduled one or more updates for i within its active window $[s_i, e_i]$.*

We shall use the following notation.

- $(N_t)_{t\geq 1}$, where $N_t := \{i : i \in \mathbb{N},\ a_i = t\}$, is the set of new requests arriving at the beginning of slot t.
- Q_t is the set of active requests in slot t.
- $x^{\pi}_{i,t}$ is an indicator variable that takes value 1 if an update for an active request i is scheduled in slot t under a policy π and 0 otherwise.

- $T^{\pi}_{i,t} := \{\tau : s_i \le \tau \le t \le e_i \ , \ x^{\pi}_{i,\tau} = 1\}$ is the set of time slots in which updates for active request i are scheduled prior to and including t under a policy π.
- $H_t := \{i : i \in \mathbb{N} \ , \ e_i \le t-1\}$ is the set of requests which are no longer active at t.
- $S^{\pi}_t := \{i : i \in H_t \ , \ T^{\pi}_{i,e_i} \ne \emptyset\}$ is the set of requests in H_t that have been serviced by policy π.

2.3 Reward and age of last update

We let r^{π}_i denote the reward obtained for a request i that was serviced under policy π. It depends on the last slot in which an update for i was scheduled when it was active. Assume the last update for an active request i is scheduled in a slot close to its start time s_i, then the "age" of the update would increase by the end of the active window and the update would not provide timely information to i as would an update scheduled in a slot close to e_i. Therefore scheduling an update for request i in a slot closer to its end time e_i is deemed advantageous. We let d^{π}_i denote the time difference between the end time of active request i and the last slot in which an update is scheduled for i under policy π, i.e., $d^{\pi}_i = e_i - \max_{t \in [s_i, e_i]} x^{\pi}_{i,t} t + 1$.

Definition 2. ***(Reward obtained from servicing*** i***)*** *The reward r^{π}_i obtained from servicing request i under a policy π depends on the last slot in which an update for i was scheduled and is collected when i is no longer active, i.e., at the end of slot e_i. It is modelled as*

$$r^{\pi}_i = \begin{cases} f(d^{\pi}_i), & \text{if } T^{\pi}_{i,e_i} \ne \emptyset, \\ 0, & \text{otherwise}, \end{cases} \tag{1}$$

where $f(\cdot)$ is a non-decreasing upper-bounded function of d^{π}_i.

Scheduling an update for an active request i under a policy π in a slot close to its end time e_i results in a smaller $e_i - \max_{t\in[s_i,e_i]} x^{\pi}_{i,t} t$ and thus in a larger reward r^{π}_i. The cumulative reward at slot t under policy π is denoted by

$$r^{\pi}(S^{\pi}_t) = \sum_{i \in S^{\pi}_t} r^{\pi}_i. \tag{2}$$

In the rest of the paper, we consider both linear and convex reward functions $f(\cdot)$.

Definition 3. ***(Linear reward function)*** *A positive linear reward function $f(x) = \alpha + \beta(w_{\max} - x + c)$ associated with servicing request i under a policy π is given by*

$$r^{\pi}_i = \alpha + \beta(w_{\max} - d^{\pi}_i + c), \tag{3}$$

where $\alpha, \beta, c \ge 0$.

The linear reward associated with servicing a request i under a policy π as defined above consists of two main components, α and $\beta(w_{\max} - d^{\pi}_i + c)$. The second component is bounded below and above as follows, $\forall i \in \mathbb{N}$,

$$\beta(w_{\max} - w_i + c) \le \beta(w_{\max} - d^{\pi}_i + c) \le \beta(w_{\max} + c).$$

Definition 4. ***(Convex reward function)*** *A positive convex function $f(x) = \alpha + h(-\beta(x-c))$ models the reward associated with servicing a request i under a policy π if*

$$r^{\pi}_i = \alpha + h(-\beta(d^{\pi}_i + c)), \tag{4}$$

where $\alpha, \beta, c \ge 0$ and $h(\cdot)$ is a positive convex function.

Paralleling the way we defined the linear reward, the convex reward has two main components, α and $h(-\beta(d^{\pi}_i + c))$, where the second term is bounded above and below as follows, $\forall i \in \mathbb{N}$,

$$h(-\beta(w_i + c)) \le h(-\beta(d^{\pi}_i + c)) \le h(-\beta c).$$

2.4 Characterization of the scheduling problem

Our objective is to maximize the reward rate obtained from servicing a sequence of requests $\boldsymbol{\rho}_T$ in a finite time window $[1, T]$, where without loss of generality, $t = 1$ corresponds to the first slot in which there are any arrival of new requests and T is the last slot after which there are no longer any active requests. For a sequence of requests $\boldsymbol{\rho}_T$, we let $g(\pi, \boldsymbol{\rho}_T) = \frac{r^{\pi}(S^{\pi}_{T+1})}{|H_{T+1}|}$ be the reward rate obtained under policy π, where $|H_{T+1}|$ is the number of requests that were active prior to to time $T+1$. Formally, the problem is defined as follows.

Problem 1.

$$\max_{\pi \in \Pi} g(\pi, \boldsymbol{\rho}_T) = \max_{\pi \in \Pi} \frac{r^{\pi}(S^{\pi}_{T+1})}{|H_{T+1}|} \tag{5}$$

$$\text{s.t.} \sum_{i \in Q_t} x^{\pi}_{i,t} \le 1, \forall t \in [1, T], \tag{6}$$

$$x^{\pi}_{i,t} \in \{0, 1\}, \forall i \in Q_t, \forall t, \tag{7}$$

where Equation (6) limits the number of users that can be scheduled at a time to at most 1, and where Π is the set of causal non-idling policies defined as follows.

Definition 5. ***(Non-idling policy)*** *A policy $\pi \in \Pi$ is said to be non-idling if it only idles when there are no active requests.*

Our goal is to design a causal non-idling scheduling policy that maximizes the reward rate in Problem 1.

Assumption 1. *In the remainder of the paper we consider the regime where $\alpha > \beta(w_{\max} + c)$ and $\alpha > h(-\beta c)$ for both linear and convex reward functions respectively.*

Discussion. An interesting regime for both the linear and convex reward functions in Definitions 3 and 4, is that where $\alpha > \beta(w_{\max} + c)$ and $\alpha > h(-\beta c)$. Then a large reward of value α is obtained if an update to an active request i is scheduled in any slot within its active window and in addition to a smaller reward of value $\beta(w_{\max} - d^{\pi}_i + c)$ which depends on how close to e_i is the last slot in which an update to i was scheduled. Therefore, a policy whose target is to maximize the reward rate defined in Problem 1 is driven to first maximizing the number of scheduled requests then, if possible, to schedule these requests as close as possible to the ends of their associated update windows. In particular, if $\beta = 0$, the linear reward function in Definition 3 reduces to $r^{\pi}_i = \alpha$ and Problem 1 corresponds to maximizing the fraction of updates scheduled within their active windows. Hence, an optimal policy π that solves this problem when $\beta = 0$ needs to schedule at most one update per active request, in any time slot within its active window. An optimal policy for this setting is the Earliest Deadline First (EDF) policy [12]. From Definition 2, the reward obtained from servicing a request i under a policy π depends on both the last slot in which an update for i is scheduled as well as on i's release time

s_i, and is hence independent of its arrival time a_i. Therefore, and without loss of generality, we assume in the rest of the paper that a request's release time is equal to its arrival time, i.e., for all $i \in \mathbb{N}$ we have that $s_i = a_i$.

3 OPTIMAL OFFLINE POLICY

In this section we characterize the optimal offline policy π^* that solves Problem 1.

Definition 6. ***(Optimal genie-based offline policy** π^***)** A policy π^* is optimal for Problem 1 if it maximizes the reward rate.*

Policy π^* has knowledge of the requests' arrivals in the entire timeline and can therefore optimally service requests. Optimal offline policies are useful since they provide an upper bound on the reward rate that can be achieved by causal policies. π^* schedules at most a single update transmission per active request, since an active request requires at most a single update being scheduled as close as possible to its end time.

4 CAUSAL SCHEDULING POLICIES

The limitation of causal policies Π is that they have no knowledge of future request arrivals. Our goal is to devise online causal policies which have provable performance guarantees in terms of Competitive Ratio (CR) with respect to the optimal offline policy, defined as follows.

Definition 7. ***(Competitive ratio of a policy** π**)** The competitive ratio of a policy π is given by*

$$\mathrm{CR}_\pi = \min_{\boldsymbol{\rho}_T \in \mathbf{P}_T} \frac{g(\pi, \boldsymbol{\rho}_T)}{g(\pi^*, \boldsymbol{\rho}_T)}, \tag{8}$$

where $\mathbf{P}_T$ is the set of all possible request arrivals in a time window of length T, π^* is the optimal offline policy that solves Problem 1, and $g(\pi, \boldsymbol{\rho}_T)$ and $g(\pi^*, \boldsymbol{\rho}_T)$ are the reward rate expressions achieved by both π and π^* respectively. An online algorithm is q-competitive for some $q \geq 1$ if it achieves at least $1/q$ of the optimal offline value in the worst case, i.e., for all $\boldsymbol{\rho}_T \in \mathbf{P}_T$, we have that $g(\pi, \boldsymbol{\rho}_T) \geq \frac{1}{q} g(\pi^*, \boldsymbol{\rho}_T)$.

We shall begin by providing a lower bound on the competitive ratio for any causal non-idling scheduling policy in Π.

4.1 Lower bound on the competitive ratio of any policy in Π

The following theorem states that any causal non-idling policy $\pi \in \Pi$ achieves a competitive ratio of at least $\frac{1}{w_{\max}}$.

Theorem 1. *For any causal non-idling policy $\pi \in \Pi$, and with a reward function that satisfies the condition in Assumption 1, the competitive ratio of π satisfies*

$$\mathrm{CR}_\pi \geq \frac{1}{w_{\max}}.$$

Proof. Consider a causal non-idling policy $\pi \in \Pi$. According to Assumption 1, the worst scheduling strategy that π can follow is to schedule the same active request every slot until it is no longer active. Let $[1, T]$ be a time window of length $T \geq w_{\max}$, where T corresponds to the slot after which there are no longer any active requests. A lower bound on the cumulative reward $r^\pi(S^\pi_{T+1})$ obtained under π is $r^\pi(S^\pi_{T+1}) \geq \frac{T}{w_{\max}} f(0)$. The cumulative reward $r^{\pi^*}(S^{\pi^*}_{T+1})$ obtained under the optimal offline policy π^* is upper bounded by $Tf(0)$. Therefore, $\forall\, T \geq w_{\max}$, the competitive ratio of π is lower bounded as follows, $\mathrm{CR}_\pi \geq \frac{r^\pi(S^\pi_{T+1})}{r^{\pi^*}(S^{\pi^*}_{T+1})} = \frac{1}{w_{\max}}$, which concludes the proof.

□

4.2 Greedy policy π^g

The causal greedy policy π^g presented in the Algorithm 1 panel schedules in every slot the request that would maximize the marginal increase in cumulative reward, and if need be, breaks ties arbitrarily (Line 5).

Algorithm 1: Greedy policy π^g.

```
1  $Q_0 \leftarrow \emptyset$; $E_0 \leftarrow \emptyset$; $S^{\pi^g}_1 \leftarrow \emptyset$; $H_1 \leftarrow \emptyset$;
2  for $t = 1, 2, \ldots$ do
3      $N_t :=$ set of new request arrivals;
4      $Q_t \leftarrow (Q_{t-1} \setminus E_{t-1}) \bigcup N_t$;
5      $i^* \in \arg\max_{i \in Q_t} f(e_i - t) - f(e_i - \max_{\tau \in [a_i, t-1]} x^{\pi^g}_{i,\tau} \tau)$; break ties
         arbitrarily;
6      $x^{\pi^g}_{i^*,t} \leftarrow 1$; $T^{\pi^g}_{i^*,t} \leftarrow T^{\pi^g}_{i^*,t-1} \bigcup \{t\}$;
7      $E_t := \{i : i \in Q_t, e_i = t\}$ ;
8      $D^{\pi^g}_t := \{i : i \in E_t, T^{\pi^g}_{i,t} \neq \emptyset\}$ ;
9      $S^{\pi^g}_{t+1} \leftarrow S^{\pi^g}_t \bigcup D^{\pi^g}_t$;
10     $H_{t+1} \leftarrow H_t \bigcup E_t$ ;
```

Theorem 2. ***(**π^g **maximizes the ratio of serviced requests)** Policy π^g maximizes the ratio of serviced requests when the linear and convex reward functions satisfy the conditions in Assumption 1.*

Proof. The proof of this theorem follows from the optimality of the Earliest Deadline First (EDF) policy in maximizing the ratio of serviced requests for the discrete time $G/D/1 - G$ queue where the service time is exactly one unit of time [12]. Following from Assumption 1, π^g prioritizes scheduling requests that have not been scheduled yet. If in a time slot t there are more than one active requests that have not been scheduled prior to t, π^g, similarly to EDF, schedules in slot t an update transmission to the request with the earliest end time. Otherwise in the case where all active requests at time t have already been scheduled prior to t, π^g schedules the request that maximizes the marginal increase in the cumulative reward, which does not affect the ratio of serviced requests. This concludes the proof. □

Theorem 3. ***(Competitive ratio of policy** π^g**)** The competitive ratio of π^g is*

$$\mathrm{CR}_{\pi^g} = \frac{f(w_{\max})}{f(0)}.$$

Proof. According to Theorem 2, π^g maximizes the ratio of serviced requests. Let $N = |S^{\pi^g}_{T+1}|$ be the total number of requests that have been serviced under π^g in a finite time window of length T. It follows that a lower bound on the cumulative reward $r^{\pi^g}(S^{\pi^g}_{T+1})$ obtained under π^g is $r^{\pi^g}(S^{\pi^g}_{T+1}) \geq Nf(w_{\max})$. The cumulative reward $r^{\pi^*}(S^{\pi^*}_{T+1})$ obtained under the optimal offline policy π^* is upper bounded by $Nf(0)$. Therefore, for all $T \geq w_{\max}$, the competitive ratio of π^g is lower bounded as follows, $\mathrm{CR}_{\pi^g} \geq \frac{r^{\pi^g}(S^{\pi^g}_{T+1})}{r^{\pi^*}(S^{\pi^*}_{T+1})} = \frac{f(w_{\max})}{f(0)}$, which concludes the proof. □

4.3 Our proposed causal scheduling heuristic policy π^a

We propose a causal scheduling policy π^a. We shall refer to it as the adversarial policy.

During every slot t, π^a assigns to every slot in the interval $[t, \max_{i \in Q_t} e_i]$ a single request that is active during this slot. We say that at time t, π^a tentatively schedules updates for active requests in slots within the interval $[t, \max_{i \in Q_t} e_i]$. By the end of slot t, an update is sent to the request scheduled in t, while the remaining slots in $(t, \max_{i \in Q_t} e_i]$ are freed from any tentative schedules.

Definition 8. ***(Tentatively scheduling updates for an active request)*** *Policy π^a tentatively schedules updates to an active request $i \in Q_t$ if during slot t it assigns slots within the interval $[t, e_i]$ to potentially transmit updates to request i in those slots.*

We clearly define additional notation specific to π^a.

- $\hat{x}^{\pi^a}_{i,t,t'}$ is an indicator that takes value 1 if at slot t, an update for active request i is tentatively scheduled in slot $t' \in [t, e_i]$.
- $\hat{T}^{\pi^a}_{i,t,t'} := \{\tau : t \leq t' \leq \tau \leq e_i \ , \ \hat{x}^{\pi^a}_{i,t,\tau} = 1\}$ is the set of time slots within the interval $[t', e_i]$ for any $t' \in [t, e_i]$, in which active request i is tentatively scheduled.

Note. As long as a request i is active, no reward is yet collected for request i. A reward associated with servicing a request i is only obtained when i is no longer active. We introduce a specific notion for the reward associated with active request i under policy π^a which we refer to as the tentative reward, defined as follows.

Definition 9. ***(A request's tentative reward under policy π^a)*** *The tentative reward $\hat{r}^{\pi^a}_{i,t,t'}$ of an active request $i \in Q_t$ on slot t is a function of the slots in $T^{\pi^a}_{i,t-1}$ in which i was actually scheduled prior to t under π^a and the slots $\hat{T}^{\pi^a}_{i,t,t'}$ in which i is tentatively scheduled after $t' \in [t, e_i]$ under π^a, and is given by*

$$\hat{r}^{\pi^a}_{i,t,t'} = \begin{cases} f\left(e_i - \max(\max\limits_{\tau \in [a_i, t-1]} x^{\pi^a}_{i,\tau}\tau, \max\limits_{\tau \in [t', e_i]} \hat{x}^{\pi^a}_{i,t,\tau}\tau)\right), \\ \text{if } T^{\pi^a}_{i,t-1} \cup \hat{T}^{\pi^a}_{i,t,t'} \neq \emptyset, \\ 0, \ \text{otherwise.} \end{cases}$$

At any time t, an active request i may have been actually scheduled one or many times prior to but not including t, and may have been tentatively scheduled after t. Hence from Definition 9, the tentative reward depends on the latest time slot in which i is either tentatively scheduled or has been actually scheduled.
We similarly define the aggregate reward associated with a set of active requests.

Definition 10. ***(Requests' aggregate tentative reward under policy π^a)*** *The aggregate tentative reward at slot t associated with the set of active requests Q_t under policy π^a, is the sum of the requests' tentative rewards and is given by $\hat{r}^{\pi^a}_{t,t'}(Q_t) = \sum_{i \in Q_t} \hat{r}^{\pi^a}_{i,t,t'}$.*

We consider the setting where the reward obtained from servicing a request is either linear or convex as introduced in Definitions 3 and 4 respectively. π^a is presented in Algorithm 2, which proceeds as follows.

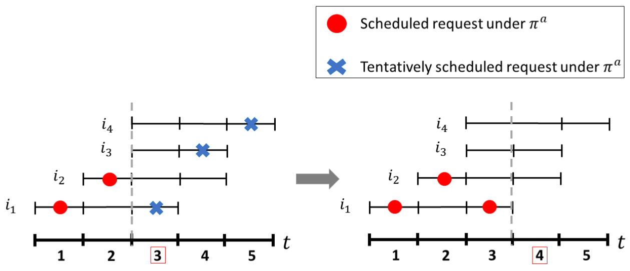
Algorithm 2: Policy π^a.

```
1  $Q_0 \leftarrow \emptyset$; $E_0 \leftarrow \emptyset$; $S^{\pi^a}_1 \leftarrow \emptyset$; $H_1 \leftarrow \emptyset$;
2  for $t = 1, \ldots$ do
3      $N_t :=$ set of new request arrivals;
4      $Q_t \leftarrow (Q_{t-1} \setminus E_{t-1}) \bigcup N_t$;
5      $x^{\pi^a}_{i,t'} \leftarrow 0, \forall t' \in [t, e_i], \forall i \in N_t$;
6      $T^{\pi^a}_{i,t} \leftarrow \emptyset, \forall i \in N_t$;
7      $\hat{x}^{\pi^a}_{i,t,t'} \leftarrow 0, \forall t' \in [t, e_i], \forall i \in Q_t$;
8      $\hat{T}^{\pi^a}_{i,t,t} \leftarrow \emptyset, \forall i \in Q_t$;
9      $\tau \leftarrow \max_{i \in Q_t} e_i$;
10     while $\tau \geq t$ do
11         $A_\tau \leftarrow \{i : i \in Q_t, \tau \in [a_i, e_i]\}$;
12         Go to Tentative schedule ⊳ Get $i^*$;
13         $\hat{x}^{\pi^a}_{i^*,t,\tau} \leftarrow 1$; $\hat{T}^{\pi^a}_{i^*,t,\tau} \leftarrow \hat{T}^{\pi^a}_{i^*,t,\tau+1} \bigcup \{\tau\}$;
14         if $\tau = t$ then
15             $x^{\pi^a}_{i^*,t} \leftarrow 1$; $T^{\pi^a}_{i^*,t} \leftarrow T^{\pi^a}_{i^*,t-1} \bigcup \{t\}$;
16         $\tau \leftarrow \tau - 1$;
17     $E_t := \{i : i \in Q_t, e_i = t\}$ ;
18     $D^{\pi^a}_t := \{i : i \in E_t, T^{\pi^a}_{i,t} \neq \emptyset\}$ ;
19     $S^{\pi^a}_{t+1} \leftarrow S^{\pi^a}_t \bigcup D^{\pi^a}_t$;
20     $H_{t+1} \leftarrow H_t \bigcup E_t$ ;
```

Algorithm 3: Tentative schedule.

```
1  Input: $(\hat{x}^{\pi^a}_{i,t,t'}, \forall t' \in [\tau+1, e_i], \forall i \in Q_t)$, $(\hat{T}^{\pi^a}_{i,t,\tau+1}, \forall i \in Q_t)$,
       $(x^{\pi^a}_{i,t'}, \forall t' \in [a_i, t-1], \forall i \in Q_t)$.
2  Output: $i^*$.
3  for $i \in A_\tau$ do
4      for $j \in A_\tau \setminus \{i\}$ do
5          $\tilde{T}^{\pi^a}_{i,t,\tau} \leftarrow \hat{T}^{\pi^a}_{i,t,\tau+1} \bigcup \{\tau\}$;
6          $\tilde{x}^{\pi^a}_{i,t,t'} \leftarrow \hat{x}^{\pi^a}_{i,t,t'}, \forall t' \in [\tau+1, e_i]$; $\tilde{x}^{\pi^a}_{i,t,\tau} \leftarrow 1$;
7          $m^{\pi^a}_{t,\tau}(i,j) = \frac{f(e_i - \max_{t' \in [\tau, e_i]} \tilde{x}^{\pi^a}_{i,t,t'} t') + f(e_j - \max_{t' \in [a_j, t-1]} x^{\pi^a}_{j,t'} t')}{f(e_j - \tau) + \hat{r}^{\pi^a}_{i,t,\tau+1}}$;
8  $i^* \in \arg\max_{i \in A_\tau} \left[ \min_{j \in A_\tau \setminus \{i\}} m^{\pi^a}_{t,\tau}(i,j) \right]$; break ties by selecting $i^*$
       with smallest $e_{i^*}$; break ties arbitrarily;
```

For every slot t, π^a operates in a backwards manner starting from the last slot in which it can tentatively schedule an update, $\max_{i \in Q_t} e_i$, all the way back to t. For every slot τ between t and $\max_{i \in Q_t} e_i$, π^a first determines A_τ, the subset of active requests



● Scheduled request under π^a
✖ Tentatively scheduled request under π^a
i_4
i_3
i_2
i_1
1
2
3
4
5
t

Figure 1: Scheduling and tentatively scheduling under policy π^a of requests with maximal update window of $w_{\max} = 3$ and receiving a linear reward as defined in Definition 3, with $\alpha = 1, \beta = 0.1, c = 0$.

in Q_t that can be tentatively scheduled in slot τ. Let $i \in A_\tau$ and $j \in A_\tau \setminus \{i\}$ be two requests that will be active in slot τ. In Algorithm 3, for every τ, π^a evaluates $m^{\pi^a}_{t,\tau}(i,j)$ (Line 7). The reasoning behind $m^{\pi^a}_{t,\tau}(i,j)$ is intuitive and described as follows. π^a evaluates through $m^{\pi^a}_{t,\tau}(i,j)$ if it is more advantageous in terms of aggregate tentative reward to tentatively schedule request i in slot τ instead of request j. Slots between $\tau+1$ and e_j in which updates to request j are tentatively scheduled under π^a are ignored when evaluating $m^{\pi^a}_{t,\tau}(i,j)$, in an attempt to characterize the importance of tentatively scheduling an update to request j solely in slot τ. A detailed derivation of $m^{\pi^a}_{t,\tau}(i,j)$ is provided as follows. The numerator of $m^{\pi^a}_{t,\tau}(i,j)$ corresponds to the sum of the following two components:

- Tentative reward of request i if i is tentatively scheduled in any slots in $\hat{T}^{\pi^a}_{i,t,\tau+1} \bigcup \{\tau\}$.
- Tentative reward of request j if j was only scheduled prior to slot t.

The denominator of $m^{\pi^a}_{t,\tau}(i,j)$ corresponds to the sum of the following two components:

- Tentative reward of request j if j is tentatively scheduled in slot τ.
- Tentative reward of request i if i is tentatively scheduled in any slots in $\hat{T}^{\pi^a}_{i,t,\tau+1}$ and/or scheduled in any slots in $T^{\pi^a}_{i,t-1}$.

Therefore, $m^{\pi^a}_{t,\tau}(i,j)$ evaluates the ratio between the aggregate tentative rewards resulting from (1) tentatively scheduling request i at least once after and including slot τ while only considering slots in which request j was actually scheduled prior to t and (2) tentatively scheduling request j only in slot τ while considering all slots strictly greater than τ in which i was tentatively scheduled to receive updates and/or all slots prior to t in which updates to request i were scheduled.

π^a then evaluates $m^{\pi^a}_{t,\tau}(i,j)$ for every request $i \in A_\tau$, and for all $j \in A_\tau \setminus \{i\}$ (Lines 3-7 of Algorithm 3), and determines the request $j \in A_\tau \setminus \{i\}$ for which the ratio $m^{\pi^a}_{t,\tau}(i,j)$ is minimized, i.e., π^a's potential decision to tentatively schedule request i in slot τ instead of request j deemed the least advantageous.

π^a finally tentatively schedules in slot τ the request i^* such that $\min_{q\in A_\tau\setminus\{i^*\}} m^{\pi^a}_{t,\tau}(i^*,q) \geq \max_{j\in A_\tau\setminus\{i^*\}} \min_{q\in A_\tau\setminus\{j\}} m^{\pi^a}_{t,\tau}(j,q)$ (Line 8 of Algorithm 3).

Figure 1 provides an example of requests scheduled and tentatively scheduled to receive updates under policy π^a. Requests i_3 and i_4 became active at the beginning of slot $t = 3$. We observe in the left subfigure of Figure 1 that requests i_1 and i_2 were scheduled to receive updates under π^a in slots $t = 1$ and $t = 2$ respectively. Additionally at $t = 3$, requests i_4, i_3 and i_1 are tentatively scheduled to receive updates under π^a in slots $t = 5$, $t = 4$ and $t = 3$ respectively. Once all slots in the interval $[3,5]$ have been reserved to tentatively schedule updates for active requests, π^a then schedules an update to request i_1 in slot $t = 3$ (right subfigure of Figure 1). The same scheduling process is repeated as long as there are active requests.

4.4 Discussion of π^g and π^a

We provide insights on both proposed causal policies, π^g and π^a. We first define $m^{\pi^g}_{t,t}(i,j)$ for two active requests i and j in Q_t as follows

$$m^{\pi^g}_{t,t}(i,j) = \frac{f(e_i - t) + f(e_j - \max_{t'\in[a_j,t-1]} x^{\pi^g}_{j,t'} t')}{f(e_j - t) + f(e_i - \max_{t'\in[a_i,t-1]} x^{\pi^g}_{i,t'} t')}. \quad (9)$$

If π^g schedules request $i \in Q_t$ in slot t then the following corollary applies.

Corollary 1. *Policy π^g schedules an update transmission in slot t to active request $i \in Q_t$ if and only if $\min_{j\in Q_t\setminus\{i\}} m^{\pi^g}_{t,t}(i,j) \geq 1$.*

Proof. Since π^g schedules an update transmission in slot t to active request $i \in Q_t$, it follows that $\forall j \in Q_t \setminus \{i\}$, $f(e_i - t) - f(e_i - \max_{t'\in[a_i,t-1]} x^{\pi^g}_{i,t'} t') \geq f(e_j - t) - f(e_j - \max_{t'\in[a_j,t-1]} x^{\pi^g}_{j,t'} t')$, which implies that

$$\forall j \in Q_t \setminus \{i\}, \ \frac{f(e_i - t) + f(e_j - \max_{t'\in[a_j,t-1]} x^{\pi^g}_{j,t'} t')}{f(e_j - t) + f(e_i - \max_{t'\in[a_i,t-1]} x^{\pi^g}_{i,t'} t')} \geq 1.$$

This is equivalent to saying that

$$\min_{j\in Q_t\setminus\{i\}} \frac{f(e_i - t) + f(e_j - \max_{t'\in[a_j,t-1]} x^{\pi^g}_{j,t'} t')}{f(e_j - t) + f(e_i - \max_{t'\in[a_i,t-1]} x^{\pi^g}_{i,t'} t')} \geq 1.$$

We can similarly prove the other direction of the condition, which concludes the proof. □

It follows from Corollary 1 that π^g schedules in every slot the request $i^* \in \arg\max_{i\in Q_t} \left[\min_{j\in Q_t\setminus\{i\}} m^{\pi^g}_{t,t}(i,j) \right]$. In comparison with π^a, if we set $\forall t, \forall i \in Q_t$, $\hat{T}^{\pi^a}_{i,t,t+1} = \emptyset$, then $\forall i,j \in Q_t$, $m^{\pi^a}_{t,t}(i,j)$ is equal to $m^{\pi^g}_{t,t}(i,j)$ and π^a becomes equivalent to π^g. In other words, if π^a does not tentatively schedule requests, then it is equivalent to the greedy request scheduling policy π^g.

According to the above derivation, π^g does not tentatively schedule active requests at time t and takes a restrictive approach by prioritizing the scheduling of any active requests that have not been scheduled prior to t. On the other hand, π^a allows for more flexibility in rescheduling active requests, where an active request that has already been scheduled prior to some slot t, can be rescheduled in t, even if this might be at the expense of not scheduling other active requests that have not been scheduled yet prior to t.

It follows that π^a does not make any assumptions about the future load of requests' arrivals and its implications on the tentative schedules. We therefore expect π^a to achieve a higher reward rate than π^g in systems with small loads and requests with large active window's lengths. On the other hand, we expect that both π^a and π^g would achieve a similar reward rate in systems with high-loads and with requests that have small active windows, balanced on the one hand by π^g's urgency to schedule new requests as soon as they become active, and on the other hand by π^a attempting to schedule/reschedule requests as close as possible to the end of their active windows.

5 NUMERICAL EVALUATIONS

We conducted numerical evaluations to explore the performance of our proposed policies π^a and π^g in maximizing the reward rate introduced in Problem 1 and evaluate their results with respect to other baseline policies which we introduce below.

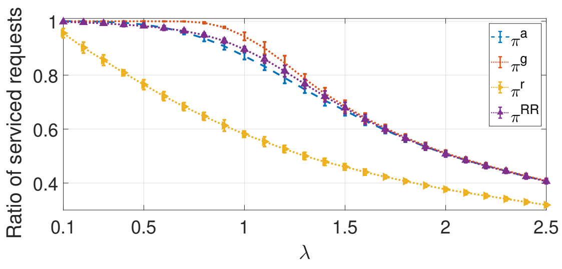
5.1 Model

We shall present results for a convex reward function that satisfies the conditions in Assumption 1. We let $h(x) = e^x$ in Definition 4 and set $\alpha = 1$, $\beta = 2$ and $c = -\frac{1}{2}\ln(0.9)$. It follows that the convex exponential reward obtained after servicing request i under policy π is $r_i^\pi = 1 + 0.9e^{-2(e_i - \max_{t\in[s_i,e_i]} x_{i,t}^\pi t)}$. The maximal achievable reward in this setting is equal to 1.9 whereas the smallest reward obtained after servicing a request is $1+0.9e^{-2w_{\max}}$. We consider the setting where the number of new request arrivals at the beginning of every slot is drawn from a Poisson distribution with intensity λ, whereas a request's active window length is generated from a discrete uniform distribution $\sim U[1, w_{\max}]$. We run simulations over a finite-time of length $T = 1000$, where T is the last slot after which there are no longer any active requests in the system. Our simulation results represent averages over randomly generated requests' arrivals as well as requests' active windows lengths. We ran 100 Monte-Carlo (MC) simulations and plotted both the mean ratio of serviced requests for any policy π, i.e., $\frac{S_{T+1}^\pi}{|H_{T+1}|}$, and the reward rate as well as the confidence intervals corresponding to the standard deviation of the estimator resulting from the MC simulations.

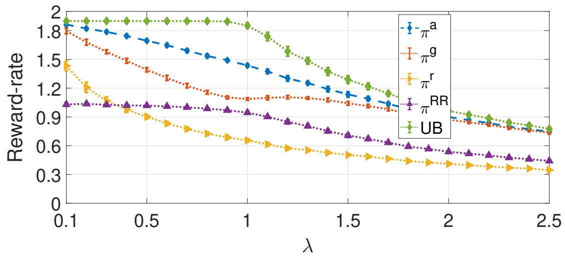
5.2 Scheduling policies

In addition to presenting the results for all of π^a and π^g, we consider two baseline policies. π^r is a causal policy that randomly schedules an active request in a slot. π^{RR} schedules active requests in a round-robin-like fashion which we describe as follows. At the beginning of every slot, the set of new requests is considered for scheduling right after the set of requests that arrived prior to this slot and have not been scheduled yet.

Due to the brute-force nature of the optimal offline policy π^* we provide instead an upper bound (UB) on the maximal achievable reward-rate, which is equal to the product between the maximal ratio of serviced requests (achieved by π^g) and the maximal achievable reward which is equivalent to 1.9 in this setting, normalized by the total number of requests H_T.



(a) Ratio of serviced requests vs. λ.



(b) Reward-rate vs. λ.

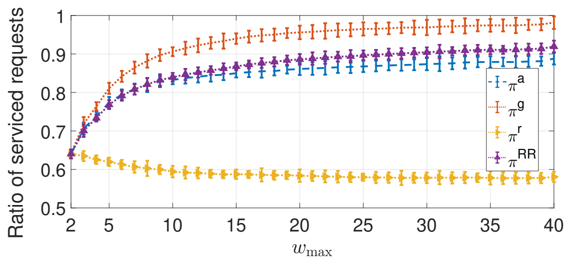
Figure 2: Ratio of serviced requests and reward-rate when the reward is convex, $w_{\max} = 30$ and λ is increasing.

5.3 On the impact of the load of requests with fixed maximal window length

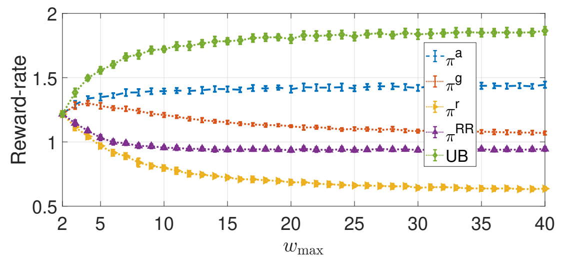
We fix $w_{\max} = 30$ and increase λ from 0.1 to 2.5. A first interesting observation is that all of π^a, π^g and π^{RR} maximize the ratio of serviced requests in both regimes where $\lambda \leq 0.4$ and $\lambda \geq 1.6$. Whereas for $0.4 \leq \lambda \leq 1.6$, π^g maximizes the ratio of serviced requests. Another interesting observation is that for $\lambda \leq 0.7$, the ratio of serviced requests achieved by π^g is constant and equal to 1, whereas the reward-rate achieved by π^g is significantly decreasing in that range. The following behavior is due to the fact that π^g maximizes the number of serviced requests at the expense of scheduling requests closer to the end of their active windows. Therefore, this justifies that π^g does not maximize the reward rate in low-load systems.

On the other hand, π^a achieves the largest reward-rate among all proposed causal policies for $\lambda \leq 2.1$ even if it is linearly decreasing as λ increases. The linear decrease in the reward-rate is justified since π^a may reschedule active requests in slots closer to their end times in an attempt to maximize the cumulative reward, which may come at the expense of servicing requests that have not been scheduled yet.

Finally, for $\lambda \geq 2.1$, both π^a and π^g achieve the largest reward-rate, close to the maximal achievable reward-rate, which is aligned with our previous analysis suggesting that while π^a is superior in systems with low-loads, both π^a and π^g achieve similar performance in systems with high requests' arrival rate.



(a) Ratio of serviced requests vs. $w_{\max}$.



(b) Reward-rate vs. $w_{\max}$.

Figure 3: Ratio of serviced requests and reward-rate when the reward is convex, $\lambda = 1$ and $w_{\max}$ is increasing.

5.4 On the impact of increased update window flexibility

We fix $\lambda = 1$ and increase $w_{\max}$ from 2 to 40. The results are shown in Figure 3. A first observation is that as $w_{\max}$ increases, the ratio of serviced requests under π^a, π^g and π^{RR} increases because of the additional flexibility in scheduling requests in more slots. For $w_{\max} \leq 3$, we observe that both π^a and π^g achieve the same reward-rate. For $w_{\max} \geq 4$, π^a achieves in a higher reward-rate than any of the other proposed causal policies. An interesting observation is that the reward-rate achieved by π^g is clearly decreasing as $w_{\max}$ increases. There are two factors that concurrently lead to the following phenomenon, the first one being that for $w_{\max} \geq 4$, the ratio of serviced requests under π^g slowly increases as $w_{\max}$ increases, and the second one being that the minimal reward obtained from servicing a request is decreasing as $w_{\max}$ increases.

That said, and as aligned with our previous discussions, π^a is superior to π^g in such settings with fixed arrival rate but requests with large active windows, since it allows for more flexibility in scheduling updates as close as possible to the requests' end times, whereas π^g is driven towards maximizing the ratio of scheduled requests with less priority to scheduling those requests closer to their end times.

6 CONCLUSION

In this paper we have developed a model where updates are required by requests prior to making timely decisions regarding time-varying processes they're interested in. Requests can receive updates only within time windows of finite length. A key aspect is the design of a reward function that captures the importance of scheduling the freshest update transmission to a request as close as possible to the decision time as well as scheduling policies that achieve high rewards in adversarial settings. A key part of our future work is to extend our model to a real-time information market that includes multiple servers and allows for multiple updates' transmissions per slot by matching requests to servers through efficient algorithms.

7 ACKNOWLEDGMENTS

We would like to thank the support of the National Science Foundation Award CNS-2212202.

REFERENCES

[1] Jean Abou Rahal, Gustavo de Veciana, Takayuki Shimizu, and Hongsheng Lu. 2020. Optimizing timely coverage in communication constrained collaborative sensing systems. In *2020 18th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT)*. IEEE, 1–8.

[2] Jean Abou Rahal, Gustavo de Veciana, Takayuki Shimizu, and Hongsheng Lu. 2022. Optimizing timely coverage in communication constrained collaborative sensing systems. *IEEE Transactions on Control of Network Systems* (2022).

[3] Antonio Franco, Björn Landfeldt, and Ulf Körner. 2019. Analysis of Age of Information threshold violations. In *Proceedings of the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. 163–172.

[4] Julian Heinovski, Jorge Torres Gómez, and Falko Dressler. 2022. A Spatial Model for Using the Age of Information in Cooperative Driving Applications. In *Proceedings of the 25th International ACM Conference on Modeling Analysis and Simulation of Wireless and Mobile Systems*. 85–94.

[5] Hoai Hoang, Magnus Jonsson, Ulrik Hagstrom, and Anders Kallerdahl. 2002. Switched real-time ethernet with earliest deadline first scheduling protocols and traffic handling. In *Proceedings 16th International Parallel and Distributed Processing Symposium*. IEEE, 6–pp.

[6] Sanjit Kaul, Roy Yates, and Marco Gruteser. 2012. Real-time status: How often should one update?. In *2012 Proceedings IEEE INFOCOM*. IEEE, 2731–2735.

[7] Antzela Kosta, Nikolaos Pappas, and Vangelis Angelakis. 2017. Age of information: A new concept, metric, and tool. *Foundations and Trends in Networking* 12, 3 (2017), 162–259.

[8] Chengzhang Li, Shaoran Li, Yongce Chen, Y Thomas Hou, and Wenjing Lou. 2020. AoI scheduling with maximum thresholds. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 436–445.

[9] Chengzhang Li, Qingyu Liu, Shaoran Li, Yongce Chen, Y Thomas Hou, and Wenjing Lou. 2021. On scheduling with AoI violation tolerance. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 1–9.

[10] Chengzhang Li, Qingyu Liu, Shaoran Li, Yongce Chen, Y Thomas Hou, Wenjing Lou, and Sastry Kompella. 2022. Scheduling With Age of Information Guarantee. *IEEE/ACM Transactions on Networking* (2022).

[11] Jörg Liebeherr, Dallas E Wrege, and Domenico Ferrari. 1996. Exact admission control for networks with a bounded delay service. *IEEE/ACM transactions on networking* 4, 6 (1996), 885–901.

[12] Shivendra S Panwar, Don Towsley, and Jack K Wolf. 1988. Optimal scheduling policies for a class of queues with customer deadlines to the beginning of service. *Journal of the ACM (JACM)* 35, 4 (1988), 832–844.

[13] Xiaoqi Qin, Yangyang Xia, Hang Li, Zhiyong Feng, and Ping Zhang. 2021. Distributed data collection in age-aware vehicular participatory sensing networks. *IEEE Internet of Things Journal* 8, 19 (2021), 14501–14513.

[14] Nived Rajaraman, Rahul Vaze, and Goonwanth Reddy. 2021. Not just age but age and quality of information. *IEEE Journal on Selected Areas in Communications* 39, 5 (2021), 1325–1338.

[15] Li-On Raviv and Amir Leshem. 2018. Maximizing service reward for queues with deadlines. *IEEE/ACM Transactions on Networking* 26, 5 (2018), 2296–2308.

[16] Kumar Saurav and Rahul Vaze. 2021. Online energy minimization under a peak age of information constraint. In *2021 19th International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt)*. IEEE, 1–8.

[17] Yin Sun, Elif Uysal-Biyikoglu, Roy D Yates, C Emre Koksal, and Ness B Shroff. 2017. Update or wait: How to keep your data fresh. *IEEE Transactions on Information Theory* 63, 11 (2017), 7492–7508.

[18] Vishrant Tripathi, Igor Kadota, Ezra Tal, Muhammad Shahir Rahman, Alexander Warren, Sertac Karaman, and Eytan Modiano. 2022. WiSwarm: Age-of-Information-based Wireless Networking for Collaborative Teams of UAVs. *arXiv preprint arXiv:2212.03298* (2022).

[19] Bingkun Yao, Hong Gao, Yang Zhang, Jinbao Wang, and Jianzhong Li. 2022. Maximum AoI Minimization for Target Monitoring in Battery-free Wireless Sensor Networks. *IEEE Transactions on Mobile Computing* (2022).