Minimizing Age of Processed Information in Wireless Networks

Chanikarn Nikunram*†, Wasin Meesena*‡, Stephen John Turner†, Sucha Supittayapornpong†

Abstract—The freshness of real-time status processing of time-sensitive information is crucial for several applications, including patient monitoring and autonomous driving. This freshness is considered in this paper for the system where unprocessed information is sent from sensors to a base station over a shared wireless network. The base station has a dedicated non-preemptive processor with a constant processing time to process information from each sensor. The age of processed information is the time elapsed since the generation of the packet that was most recently processed by a processor. Our objective is to minimize the average age of processed information over an infinite time-horizon. We first show that a drop-free policy simplifies the system without sacrificing optimality. From this simplification, we propose three transmission-scheduling policies with 2-optimal guarantees for different requirements. A distributed Power-2 policy can be implemented without a central scheduler. With a central scheduler, both Back-Off and Max-Weight policies are near optimal with different advantages. The Back-Off policy guarantees a bound on the maximum age of processed information, while the Max-Weight policy achieves the lowest average age in simulation without the guarantee of bound. Simulation results confirm our theoretical findings.

Index Terms—Age of information, Transmission scheduling, Max-weight, Scheduling policy, Wireless network, Optimization

I. INTRODUCTION

Real-time status processing finds various applications, including patient monitoring [1], video surveillance [2] and autonomous driving [3]. These applications collect real-time information from sensors over wireless communication, process the obtained information at a base station, and utilize the processed information for decision making. For example, an autonomous vehicle’s processors receive data wirelessly from sensors, and the data is processed by a dedicated processor to drive safely.

To ensure that systems always make informed decisions based on up-to-date data, the freshness of processed information becomes one of the key factors. The freshness of information can be rigorously modeled by the age of information (AoI) ([4], [5] and references therein), which is the time elapsed since the generation of the packet that the base station most recently received. AoI has been extensively studied from queueing analysis [6]–[9], and scheduling problems, such as throughput constraints [10], power constraints [11], single-hop [12], and multi-hop [13]. Moreover, AoI from the perspective of joint sampling and scheduling has been investigated in [14]. However, those studies focus on the age of information unprocessed (AoIU) of packets that have just arrived at the base station. In some circumstances, the newly received information must be processed before the systems can use it. Thus, we must incorporate this processing time into scheduling procedures.

In this paper, the age of information processed (AoIP), which is the time elapsed since the generation of the packet that was most recently processed by a processor, is modeled to quantify the freshness of information for real-time status processing. The concept of AoIP was originally proposed in [15], for an edge-computing real-time IoT application. As the IoT device has limited resources, it can choose to process the sampled data locally or offload wirelessly for processing at a nearby edge server. The status sampling frequency and processing offloading policy are jointly optimized with the aim of minimizing the average AoIP. In their model, only a single IoT device is considered, together with its local processor and edge server. AoIP minimization with local and edge processing was also considered in [16] and [17], but with different models and processing constraints.

This paper considers a different system model, in which there are multiple sensors and processors, where different processors have different processing times. Time-sensitive and unprocessed information is transmitted from sensors to a base station over a shared wireless channel. To process information from each sensor, the base station possesses a dedicated non-preemptive processor with a constant processing time. Each processor may process one packet at a time and is said to have fresher information when a new packet has been fully processed—rather than when a new packet arrives at the base station.

Our goal is to develop scheduling policies for a resource-constrained wireless channel that minimizes the average age of processed information over an infinite time-horizon. While the AoIP can better capture the aspect of processed information, it is challenging to derive, compared to the AoU. Thus, we concentrate on a subclass of scheduling policies, namely drop-free policies, that make the analysis of the average age of processed information more tractable. We show that this subclass of policies does not sacrifice any optimality and derives a lower performance bound to the optimal average sum of AoIP. Afterward, we present three drop-free scheduling policies, called the Power-2 policy, the Back-Off policy, and the Max-Weight policy. They all achieve 2-optimal performance guarantees with different advantages.

To elaborate, the Power-2 policy is a low-complexity and fully distributed. On the other hand, with a central scheduler employing a greedy approach, the Back-Off policy can significantly outperform the Power-2 policy in terms of the average
In this paper, we consider the class of all scheduling policies, denoted by \( \Pi \), that make transmission decision \( \{x_i(t)\}_{i \in \mathcal{N}} \) every slot such that the interference constraint is satisfied. Henceforth, we assume all transmissions satisfy the interference constraint. Every transmitted packet is assumed to take one slot to arrive at the base station [11]–[13], [18], [19]. When a packet arrives at the base station, it is redirected immediately to a corresponding processor.

All processors are non-preemptive. Each processor \( i \) can process one packet at a time, and the processing time takes \( D_i \) consecutive slots for any constant \( D_i \in \mathbb{N} \), where \( \mathbb{N} \) is the set of natural numbers. Therefore, when a new packet for processor \( i \) arrives at time \( t \) and the processor is idle, not processing any packet, the packet will be completely processed at the end of slot \( t + D_i - 1 \) and will be considered as processed information in slot \( t + D_i \). However, if processor \( i \) is busy (not idle), the newly arrived packet will be dropped and is no longer considered for processing. We call this situation wasteful transmission. Note that the system could introduce queues to store newly-arrived packets while processors are not idle. However, later we will show in Section III that those queues are unnecessary for optimality.

### B. Wasteful Transmission

To prevent wasteful transmission, where a transmitted packet is dropped, we introduce \( f_i(t) \) as an indicator variable that equals 1 if a transmission from node \( i \) in slot \( t \) will be wasteful, and \( f_i(t) = 0 \) otherwise. We assume all processors are initially idle, so \( f_i(1) = 0, \forall i \in \mathcal{N} \). When node \( i \) transmits a packet in slot \( t \) knowing that the transmission will not be wasteful, i.e., \( x_i(t) = 1 \) and \( f_i(t) = 0 \), at the beginning of slot \( t + 1 \) the processor \( i \) will be busy processing this transmitted packet until the end of slot \( t + D_i \) as illustrated in Figure 1. This implies that \( f_i(\tau) = 1 \) for \( \tau = \{t + 1, \ldots, t + D_i - 1\} \) and \( f_i(t + D_i) = 0 \). In particular, any packet from node \( i \) transmitted during \( \{t + 1, \ldots, t + D_i - 1\} \) will be dropped, but a packet transmitted at time \( t + D_i \) will get processed right after the previous processing completes.

If node \( i \) does not transmit in slot \( t \) and \( f_i(t) = 0 \), then \( f_i(t + 1) = 0 \). However, if node \( i \) transmits a packet in slot \( t \) knowing that the transmission will be wasteful, i.e., \( x_i(t) = 1 \) and \( f_i(t) = 1 \), then the transmitted packet will be dropped in slot \( t + 1 \) right after it has been redirected to processor \( i \). Also, this wasteful transmission will not affect the values of \( f_i(t) \). The dynamic of \( f_i(t) \) is illustrated in Figure 2.

### C. Age of Information

The Age of Information (AoI) represents how fresh the information received by the base station is. In this paper, we extend AoI to two metrics, considering unprocessed and processed information. They are formally defined with their relationship below. We also assume that every packet is generated right before transmission, and its age starts at zero.

The Age of Information Unprocessed (AoIU) of node \( i \) at time \( t \) represents how old the transmitted packet that the base station most recently received from node \( i \) is. This AoIU is
is illustrated in Figure 2 and is written as follows

\[
A_i^U(t+1) = \begin{cases} 
1, & \text{if } x_i(t) = 1; \\
A_i^U(t) + 1, & \text{otherwise.}
\end{cases}
\]  

To measure the AoIP of the entire system when a policy \( \pi \) is employed, we consider the Average Weighted Sum of AoIP (AWSAoIP) as follows, for every \( \pi \in \Pi \),

\[
J_P^\pi = \lim_{T \to \infty} \frac{1}{NT} \sum_{t=1}^{T} \sum_{i \in N} w_i A_i^P(t)
\]  

with the same weight \( w_i \) defined in (3).

The initial AoIU and AoIP can be any arbitrary positive integers. We assume that \( A_i^P(1) = A_i^U(1) = 1 \) for all \( i \in N \).

### D. System Objective

In this paper, we develop scheduling policies that minimize the AWSAoIP in (5) under the constraint in (1) as follows

\[
J_P^* = \min_{\pi \in \Pi} \left\{ \lim_{T \to \infty} \frac{1}{NT} \sum_{t=1}^{T} \sum_{i \in N} w_i A_i^P(t) \right\}
\]  

s.t. \( \sum_{i \in N} x_i(t) \leq 1, \quad \forall t \in \{1, 2, \ldots\} \),

where \( J_P^* \) is the minimum AWSAoIP achieved under a scheduling policy that satisfies the interference constraint. While we aim to minimize the AWSAoIP, the evolution of \( A_i^P(t) \) in (4) is too cumbersome to deal with, compared to \( A_i^U(t) \) in (2). The next section considers a special class of policies that render our analysis tractable. Note that, while the scheduling problem in (6) is deterministic, computing the optimal policy offline needs to search over a policy space, which is exponentially large with respect to \( N \) and \( \prod_{i \in N} D_i \).

### III. DROP-FREE POLICIES AND PRELIMINARY

We observe that wasteful transmission consumes network resources without improving AWSAoIP. Therefore, we consider a drop-free scheduling policy that avoids scheduling a transmission from nodes that will cause a packet drop, once the packet arrives at the base station. We denote the class of drop-free policies by \( \Pi^{DF} \). Note that \( \Pi^{DF} \subset \Pi \). Next, we introduce some useful properties of drop-free policies, derive a lower performance bound, and state a preliminary lemma, where their proofs are provided in Section III of [22].

#### A. Drop-Free Policies

For each node \( i \), we begin with decomposing the first \( T \) slots into intervals between transmissions. Let \( V_i[T] \) be the total number of packets transmitted by node \( i \) within these \( T \) slots, i.e., \( \sum_{t=1}^{T} x_i(t) = V_i[T] \). Let \( I_i[m] \) be the number of slots between the \( (m - 1) \)-th and \( m \)-th packet deliveries from node \( i \) for all \( m \in \{1, 2, \ldots, V_i[T]\} \), and let \( R_i \) be the number of remaining slots after the last transmission to slot \( T \). We have

\[
T = \sum_{m=1}^{V_i[T]} I_i[m] + R_i, \quad \forall i \in N.
\]  

Next, we show that drop-free policies lead to a simple relationship between unprocessed and processed ages.
Lemma 1. For any drop-free scheduling policy, the relationship between $A^r_i(t)$ in (2) and $A^s_i(t)$ in (4) always follows:

$$A^r_i(t + D_i) = A^s_i(t) + D_i, \quad \forall t \geq I_i[1] + 1, \quad \forall i \in \mathcal{N}. \quad (8)$$

The implication of Lemma 1 is that the relationship between AoIU and AoIP becomes much simpler under drop-free policies. The next lemma considers their relationship.

Lemma 2 (AWSAoIU and AWSAoIP). For any system with $\{w_i, D_i\}_{i \in \mathcal{N}}$, the following holds for every drop-free scheduling policy, $\pi \in \Pi^{D, F}$,

$$J^\pi_p = J^T_p + \frac{1}{N} \sum_{i \in \mathcal{N}} w_i D_i, \quad (9)$$

where $J^\pi_p$ and $J^T_p$ are defined respectively in (5) and (3).

Next, we argue that considering the class of drop-free policies does not affect the optimal AWSAoIP in (6a).

Lemma 3 (Drop-free Optimal Scheduler). For any system with $\{w_i, D_i\}_{i \in \mathcal{N}}$, there exists a drop-free optimal scheduler.

Combining Lemma 2 and Lemma 3, we have that

$$J^\pi_p = \min_{\pi \in \Pi^{D, F}} J^\pi_p = \min_{\pi \in \Pi^{D, F}} J^T_p + \frac{1}{N} \sum_{i \in \mathcal{N}} w_i D_i. \quad (10)$$

This simplifies our optimization only to minimize AWSAoIU over drop-free policies. However, finding the optimal AWSAoIU is challenging, so we mitigate this by deriving a lower performance bound and later show that our policies in Section IV are within the proximity of the bound.

B. Lower Performance Bound

We consider the long-term transmission rate and derive its property before providing a lower bound. The long-term transmission rate of node $i$ under policy $\pi \in \Pi$ is defined by

$$r^\pi_i = \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} x_i(t) = \lim_{T \to \infty} \frac{V_i[T]}{T}. \quad (11)$$

Lemma 4 (Maximum Transmission Rates). For any system with $\{w_i, D_i\}_{i \in \mathcal{N}}$ and for any drop-free policy $\pi \in \Pi^{D, F}$, the long-term transmission rate from each node $i$ is at most $1/D_i$.

In particular, the following holds

$$r^\pi_i \leq \frac{1}{D_i}, \quad \forall i \in \mathcal{N}. \quad (12)$$

Theorem 1 (Lower bound). For any system with $\{w_i, D_i\}_{i \in \mathcal{N}}$, the optimal AWSAoIP $J^p_*$ has the lower bound $L_B$ that is the solution of the following problem, i.e., $J^p_* \geq L_B$ and

$$L_B = \min_{\pi \in \Pi^{D, F}} \left\{ \frac{1}{N} \sum_{i \in \mathcal{N}} w_i \left( D_i + \frac{1}{2} + \frac{1}{2r^\pi_i} \right) \right\} \quad (13a)$$

s.t.

$$\sum_{i \in \mathcal{N}} r^\pi_i \leq 1 \quad (13b)$$

and

$$0 < r^\pi_i \leq \frac{1}{D_i}, \quad \forall i \in \mathcal{N}. \quad (13c)$$

Proof. The proof is in the supplementary material [22].

Intuitively, Theorem 1 provides the lower performance bound of an optimal drop-free policy. It is later used to analyze the performances of three scheduling algorithms in Section IV. Note that our lower bound in (13) differs from [14], [19] in that it involves i) processing constraints and ii) drop-free policies.

Furthermore, it is possible to show that this lower bound also holds for another system with queues placed in front of the processors, where processors can be either preemptive or non-preemptive. In other words, if some drop-free scheduling policy achieves the bound, that policy is also near-optimal for the system with queues. This claim can be proven by considering a rate of processed packets. Since this rate cannot exceed the transmission rate and packets get older when being queued, we can derive a similar bound as in (13).

The bound in (13) can be made explicit by finding the set of optimal rates $\{r^*_i\}_{i \in \mathcal{N}}$ that minimizes (13a) under constraints (13b) and (13c). These optimal rates are obtained from solving the convex optimization in (13) by a solver, such as CVX [23]. Substituting the optimal rates $\{r^*_i\}_{i \in \mathcal{N}}$ into (13a) gives

$$J^p_* \geq L_B = \frac{1}{N} \sum_{i \in \mathcal{N}} w_i \left( \frac{1}{2r^*_i} + \frac{1}{2} + D_i \right). \quad (14)$$

Furthermore, when $\sum_{i \in \mathcal{N}} w_i D_i \leq 1$, it is easy to see that $r^*_i = \frac{1}{D_i}$ for every node $i$, and we have

$$J^p_* \geq L_B = \frac{1}{N} \sum_{i \in \mathcal{N}} w_i \left( \frac{3}{2} D_i + \frac{1}{2} \right). \quad (15)$$

C. Preliminary of Performance Guarantee

To guarantee the performance of a scheduling policy $\pi \in \Pi$, we say $\pi$ is $k$-optimal if $J^\pi_p \leq k \times J^p_*$ where $J^p_* = \min_{\pi \in \Pi} J^p_*$. We first prove a preliminary lemma.

Lemma 5 (inter-deliver-optimal). When a scheduling policy has the property that $\left| \frac{1}{r^*_i} \right| \leq I_i[m] \leq 2 \left| \frac{1}{r^*_i} \right|$ for every $i \in \mathcal{N}$ and $m \in \mathbb{N}$, the policy is 2-optimal. Moreover, it is 4/3-optimal if $\sum_{i \in \mathcal{N}} \frac{1}{D_i} \leq 1$.

IV. SCHEDULING POLICIES

We present three scheduling policies, namely, Power-2, Back-Off, and Max-Weight, and their performance guarantees.

A. Power-2 Policy

The Power-2 policy minimizes AWSAoIP with a 2-optimal guarantee. It is distributed, cyclic, and has low complexity. The policy is inspired by the work in [18] for a different problem. We present our policy and derive its performance guarantee.

Without loss of generality, node indices are rearranged such that $r^\pi_1 \geq r^\pi_2 \geq \ldots \geq r^\pi_N$. We define $D_i^{PW} = 2^{\left\lceil \log_2(1/r^\pi_i) \right\rceil}$ as a fixed inter-delivery time for source node $i$. Note that $D_i^{PW} = D_i^{PW} \leq \ldots \leq D_N^{PW}$. Then, for each node $i$, we determine a basic time $B_i$, in which every packet from node $i$ is transmitted in slot $t$ where $t \equiv B_i \mod D_i^{PW}$. Intuitively, $B_i$ is the first slot node $i$ transmits a packet and it keeps transmitting every $D_i^{PW}$ slots. Therefore, node $i$ only needs to know $B_i$ and $D_i^{PW}$ to make transmission
decisions independently from other nodes, assuming that time
is synchronized.

The procedure for finding the basic time $\{B_i\}_{i \in N}$ so that
the policy is drop-free and satisfies the constraint in (1) is
summarized in Algorithm 1. This algorithm may be optimized
so that its implementation has time complexity $O(N)$.

Algorithm 1: Determining basic time $\{B_i\}$

for $i \in \{1, \ldots, N\}$ do
\begin{align*}
P_i &= \left\{ t \in \{1, 2, \ldots, D_i^{PW} \} \mid t \neq B_j \bmod D_j^{PW}, \forall j < i \right\} \\
B_i &= \min P_i
\end{align*}
end

Now, we show that Algorithm 1 always constructs the basic
times $\{B_i\}_{i \in N}$ before showing that the interference constraint
in (1) is satisfied by the transmissions generated by these basic
times. The proofs of Theorems 2, 3, and 4 are provided in [22].

Theorem 2 (Existence of Basic Times). For any system
with $\{w_i, D_i\}_{i \in N}$, Algorithm 1 always leads to non-empty $P_i$ for
all $i \in N$.

Theorem 3 (No Concurrent Transmissions). For any system
with $\{w_i, D_i\}_{i \in N}$, the Power-2 policy satisfies the interference
constraint in (1).

Theorem 4 (Performance Guarantee of Power-2). For any system
with $\{w_i, D_i\}_{i \in N}$, the Power-2 policy is 2-optimal, and it is 4/3-optimal if $\sum_{i \in N} \frac{1}{D_i} \leq 1$.

Intuitively, the Power-2 policy utilizes a set of fixed inter-
deliver times to achieve the 2-optimal guarantee with simple
and distributed implementation. However, this comes at a cost
of resource under-utilization, i.e., no transmission even when
some processor is idle. The next policy improves on this issue.

B. Back-Off Policy

The Back-Off policy employs a greedy approach to select a
node for transmission without causing wasteful transmissions.
This is done by restricting the inter-delivery time to be at least
$D_i^{BO} = \left\lceil \frac{1}{\tau_i^*} \right\rceil$ for every node $i \in N$. Later, we show that
the policy is 2-optimal and outperforms the Power-2 policy in
simulations. In particular, it has bounded maximum AoIP.

The Back-Off policy maintains a list of candidate nodes for
drop-free transmission in every slot. Let $b_i(t)$ be an indicator
variable that equals 0 if node $i$ is a candidate for slot $t$. If
$b_i(t)$ equals 1, node $i$ is backed off from the list. We initialize
$b_i(t) = 0 \ \forall i \in N \ \forall t \in N$, and then $b_i(t)$ is updated
dynamically as follows. If node $i$ transmits during slot $t$, it
will back off from the list for $D_i^{BO} - 1$ slots. That is $b_i(t) = 1$\nwhere $\tau \in \{ t + 1, t + 2, \ldots, t + D_i^{BO} - 1 \}$. Since $\left\lceil \frac{1}{\tau_i^*} \right\rceil \geq D_i$, the Back-Off policy is drop-free.

There could be multiple candidate nodes in a slot. Therefore,
we introduce a countdown time $C_i(t)$ as a variable that
quantifies how urgent node $i$ should transmit in slot $t$. This
$C_i(t)$ is initialized by $C_i(1) = D_i^{BO}$ and evolves as follows
\begin{equation}
C_i(t) = \begin{cases} D_i^{BO} & t > D_i^{BO} \text{ and } x_i(t-D_i^{BO}) = 1; \\
\infty & b_i(t) = 1; \\
C_i(t-1) - 1 & \text{otherwise.}
\end{cases}
\end{equation}

That is, when node $i$ is backed-off, the countdown time is
infinite. If it has just stopped being backed-off, then $C_i(t)$ is set
to $D_i^{BO}$. If the node is not backed-off and it does not transmit
a packet, then the countdown time reduces by 1. Intuitively, the
lower $C_i(t)$ is, the more urgent node $i$ needs to transmit a
packet in slot $t$. Thus, in each slot $t$, the Back-Off policy
schedules a transmission from node $j$ where
\begin{equation}
j = \arg \min_{i \in N : b_i(t) = 0} C_i(t).
\end{equation}

Also, if all nodes are backed off, $b_i(t) = 1$ for every $i \in N$,
nothing is transmitted during slot $t$. The policy is summarized
in Algorithm 2, which makes an online decision every slot
with time complexity $O(N)$.

Algorithm 2: Back-Off Policy

Initialize $b_i(t) \leftarrow 0, C_i(1) \leftarrow D_i^{BO}, \forall i \in N, \forall t \in \{1, 2, \ldots\}$
for $t \in \{1, 2, \ldots\}$ do
\begin{align*}
&\text{Initialize } x_j(t) \leftarrow 0, \forall i \in N \\
&\text{if some nodes are not being backed-off at time } t \\
&\quad \text{then} \\
&\quad \text{Let } j = \arg \min_{i \in N : b_i(t) = 0} C_i(t) \\
&\quad \text{Set } x_j(t) \leftarrow 1 \\
&\quad \text{Set } b_j(t) \leftarrow 1, \forall t \in \\
&\quad \{ t + 1, t + 2, \ldots, t + D_i^{BO} - 1 \} \\
&\quad \text{end} \\
&\text{Update } A_i^P(t+1), A_i^F(t+1), \text{ and } C_i(t+1) \\
&\quad \text{according to (2), (4), and (16)}
\end{align*}
end

The performance of the Back-Off policy can be analyzed
by showing that every countdown time never reaches a zero
as stated in the following lemma. Note that we set $D_i^{BO} = \left\lceil \frac{1}{\tau_i^*} \right\rceil$ for all $i \in N$, so $\sum_{i=1}^{N} \frac{1}{D_i^{BO}} \leq 1$ always holds.

Theorem 5 (Performance Guarantee of Back-Off). For any system
with $\{w_i, D_i\}_{i \in N}$, the Back-Off policy is 2-optimal, and it is 4/3-optimal if $\sum_{i \in N} \frac{1}{D_i} \leq 1$.

Corollary 1 (Bounded Maximum AoIP of Back-Off). The
Back-Off policy has bounded maximum AoIP satisfying
\begin{equation}
A_i^P(t) \leq 2 \left\lceil \frac{1}{\tau_i^*} \right\rceil + D_i - 1 \ \forall i \in N, \forall t \geq I_i[1] + D_i + 1.
\end{equation}

The proofs of Theorem 5 and Corollary 1 are provided in [22].
The Back-Off policy still has unused slots while some pro-
cessors are idle. The next section considers a scheduling policy
that always makes unwasteful transmission when possible.
C. Max-Weight Policy

The Max-Weight policy is a drop-free policy that requires a central scheduler. It is derived from a Lyapunov Optimization technique [5, 19]. We first derive the policy.

The network state is defined by $S(t) = \{A_i^u(t), f_i(t)\}_{i \in N}$, where $A_i^u(t)$ and $f_i(t)$ are AoI and the wasteful transmission indicator defined in Section II. We consider the linear Lyapunov function with a positive constant $\alpha_i$ for all $i$

$$L(S(t)) = \frac{1}{N} \sum_{i \in N} \alpha_i A_i^u(t). \quad (17)$$

The Lyapunov drift is defined by

$$\Delta(S(t)) = L(S(t + 1)) - L(S(t)). \quad (18)$$

The evolution of $A_i^u(t + 1)$ in (2) can be rewritten as

$$A_i^u(t + 1) = x_i(t) + (1 - x_i(t)) (A_i^u(t) + 1). \quad (19)$$

Substituting (19) and (17) to (18), we get

$$\Delta(S(t)) = \frac{1}{N} \sum_{i \in N} \alpha_i [A_i^u(t + 1) - A_i^u(t)]$$

$$= -\frac{1}{N} \sum_{i \in N} x_i(t) \alpha_i A_i^u(t) + \frac{1}{N} \sum_{i \in N} \alpha_i. \quad (20)$$

The Max-Weight policy minimizes the Lyapunov drift in (20) to reduce the progress of AoI in every slot. Since the only controllable variable is the transmission $x_i(t)$, the policy selects node $i$ that i) causes no wasteful transmission, $f_i(t) = 0$, and ii) has the largest coefficient, $\alpha_i A_i^u(t)$. In particular, node $j$ is selected for transmission when

$$j = \arg \max_{i \in N; f_i(t) = 0} \alpha_i A_i^u(t).$$

The Max-Weight policy with $\alpha_i = w_i/r_i^*$ for every $i \in N$. We summarize the policy in Algorithm 3, which makes an online decision every slot with time complexity $O(N)$. Recall that $r_i^*$ is the optimal rate of the problem in (13). The performance of Max-Weight is provided in Theorem 6.

Algorithm 3: Max-Weight Policy

```
Initialize $f_i(t) \leftarrow 0, \forall i \in N, \forall t \in \{1, 2, \ldots\}$
for $t \in \{1, 2, \ldots\}$ do
  Initialize $x_i(t) \leftarrow 0, \forall i \in N$
  if no wasteful transmission is possible at time $t$
    then
      Let $j = \arg \max_{i \in N; f_i(t) = 0} \frac{w_i}{r_i^*} A_i^u(t)$
      Set $x_j(t) \leftarrow 1$
      Set $f_j(t) \leftarrow 1, \forall \tau \in \{t + 1, t + 2, \ldots, t + D_j - 1\}$
  end
  Update $A_i^u(t + 1)$ and $A_i^f(t + 1)$ according to (2) and (4)
end
```

Theorem 6 (Performance Guarantee of Max-Weight). For any system with $\{w_i, D_i\}_{i \in N}$ and corresponding optimal rate $\{r_i^*\}_{i \in N}$, the Max-Weight policy with $\alpha_i = w_i/r_i^*$ for all $i \in N$ is 2-optimal, and it is 5/3-optimal if $\sum_{i \in N} 1/D_i \leq 1$.

Proof. The proof is in the supplementary material [22].

V. SIMULATION RESULTS

This section evaluates the performance of Power-2, Back-Off, and Max-Weight policies in two aspects: AWSAoIP and maximum AoI. We use CVXPY [24] to solve the convex optimization in (13) for optimal transmission rates. For each policy, we simulate $10^5$ slots to ensure convergence.

A. Average age of information processed

We evaluate AWSAoIP under the three policies in networks consisting of varying the number of nodes. We begin with a group of 5 nodes with processing times (24, 52, 70, 37, 54) and weights (4.1, 7.2, 1.1, 3.0, 1.4). These processing time yields $\sum_{i=1}^{5} 1/D_i \approx 0.12$. In Figure 3, we simulate 40 network setups. Each setup contains a multiple of groups, and the largest setup has $5 \times 40 = 200$ nodes.

The simulation result shows that both centralized policies—Back-Off and Max-Weight policies—are near-optimal as their AWSAoIP are close to the lower bound in (13). For large networks, Max-Weight policy slightly outperforms the Back-Off policy. The Power-2 policy trades performance for the simple distributed algorithm, yet it still achieves 2-optimal. The AWSAoIP s from these policies are flat until $N = 40$ because $\sum_{i=1}^{N} 1/D_i \leq 1$, so the systems are processor bound. After $N = 40$, the systems are communication bound and AWSAoIP increases linearly with the number of nodes.

In addition, we compare the numerical performance of our proposed scheduling policies against a non-drop-free Max-Weight policy, namely NDF Max-Weight, that schedules transmission from a node with the largest weight in each slot (i.e., $w_i A_i^u(t)/r_i^*$) without considering its processor’s availability. The simulation result is shown in Figure 3. When the number of nodes is small, the processing constraint is influential and degrades scheduling performance. Nevertheless, when the system is large, the communication constraint outweighs the processing constraint, so the NDF Max-Weight’s performance is identical to our Max-Weight policy.
B. Maximum age of information processed

We illustrate that the Back-Off policy guarantees a bound on the maximum AoIP as in Corollary 1. Consider a network setup consisting of 9 nodes, where $D_1 = 2$ and $D_i = 16$ for $i \in \{2, 3, \ldots, 9\}$ and all weights are 1. The $J_{A1}^{BO}$ and $J_{A1}^{MW}$ in this setup are 199.5 and 202.0 respectively. Figure 4 shows the frequencies of $A_1^F(t)$ and $A_9^F(t)$ under both policies, and the red dash lines are the maximum AoIP from Corollary 1. The Back-Off policy keeps all AoIPs within the maximum. Besides, it is possible to construct a setup for the Max-Weight policy that the frequency of event $A_1^F(t) = y$ is non-zero while $r_1^y = 0.5$ for any positive integer $y$, and the maximum AoIP is unbounded linearly by $c/r_1^y$ for any constant $c$.

VI. CONCLUSIONS AND FUTURE RESEARCH

This paper considered a wireless network with nodes transmitting unprocessed packets to their corresponding non-preemptive processors at the base station over a shared wireless channel. We designed three scheduling policies: Power-2, Back-Off, and Max-Weight to minimize the Average Weighted sum AoIP of the system. All policies are 2-optimal and have different advantages. The Power-2 policy is simple and distributed. The Back-Off policy is near-optimal and has a maximum AoIP bound guarantee. The Max-Weight policy achieves closest to the system’s lower performance bound. We validate them through simulations.

For future work, we will consider more challenging scenarios such as unreliable wireless channels, periodic packet generation, and random processing times.

REFERENCES

[1] O. Salem and A. Mehaoua, “A secure framework for remote healthcare monitoring using the internet of medical things,” in ICC 2022 - IEEE International Conference on Communications, 2022, pp. 1233–1238.
[2] M. Kong, Y. Guo, O. Alkhazragi, M. Sait, C. H. Kang, T. K. Ng, and B. S. Ooi, “Real-time optical-wireless video surveillance system for high visual-fidelity underwater monitoring,” IEEE Photonics Journal, vol. 14, no. 2, pp. 1–9, 2022.
[3] X. Xia, E. Hashemi, L. Xiong, and A. Khajepour, “Autonomous vehicle kinematics and dynamics synthesis for sideslip angle estimation based on consensus kalman filter,” IEEE Transactions on Control Systems Technology, vol. 31, no. 1, pp. 179–192, 2023.
[4] R. D. Yates, Y. Sun, D. R. Brown, S. K. Kaul, E. Modiano, and S. Ulukus, “Age of information: An introduction and survey,” IEEE Journal on Selected Areas in Communications, vol. 39, no. 5, pp. 1183–1210, 2021.
[5] Y. Sun, I. Kadota, R. Talak, and E. Modiano, “Age of information: A new metric for information freshness,” Synthesis Lectures on Communication Networks, vol. 12, no. 2, pp. 1–224, 2019.
[6] S. K. Kaul, R. D. Yates, and M. Gruteser, “Real-time status: How often should one update?” in Proceedings of the IEEE INFOCOM 2012, Orlando, FL, USA, March 25-30, 2012, A. G. Greenberg and K. Sohraby, Eds. Orlando, FL, USA: IEEE, 2012, pp. 2731–2735. [Online]. Available: https://doi.org/10.1109/INFOCOM.2012.6195689
[7] M. Moltzafet, M. Leinonen, and M. Codreanu, “On the age of information in multi-source queueing models,” IEEE Transactions on Communications, vol. 68, no. 8, pp. 5003–5017, 2020.
[8] D. Deng, Z. Chen, Y. Jia, L. Liang, S. Fang, and M. Wang, “Age of information in a multiple stream m/g/11 non-preemptive queue,” in ICC 2021 - IEEE International Conference on Communications. Montreal, QC, Canada: IEEE, 2021, pp. 1–6.
[9] Z. Fang, J. Wang, C. Jiang, X. Wang, and Y. Ren, “Average peak age of information in underwater information collection with sleep-scheduling,” IEEE Transactions on Vehicular Technology, vol. 71, no. 9, pp. 10322–10136, 2022.
[10] Y. H. Bae and J. W. Baek, “Age of information and throughput in random access-based iot systems with periodic updating,” IEEE Wireless Communications Letters, vol. 11, no. 4, pp. 821–825, 2022.
[11] H. Tang, J. Wang, L. Song, and J. Song, “Minimizing age of information with power constraints: Multi-user opportunistic scheduling in multi-state time-varying channels,” IEEE Journal on Selected Areas in Communications, vol. 38, no. 5, pp. 854–868, 2020.
[12] Y.-P. Hsu, E. Modiano, and L. Duan, “Scheduling algorithms for minimizing age of information in wireless broadcast networks with random arrivals,” IEEE Transactions on Mobile Computing, vol. 19, pp. 2903–2915, 2020.
[13] J. Lou, X. Yuan, P. Sigdel, X. Qin, S. Kompella, and N.-F. Tzeng, “Age of information optimization in multi-channels based multi-hop wireless networks,” IEEE Transactions on Communications, pp. 1–15, 2022.
[14] C. Li, S. Li, Y. Chen, Y. T. Hou, and W. Lou, “Age of information under general models for iot data collection,” IEEE Trans. Netw. Sci. Eng., vol. 7, no. 4, pp. 2256–2270, 2020. [Online]. Available: https://doi.org/10.1109/TNSSE.2019.2952764
[15] R. Li, Q. Ma, J. Gong, Z. Zhou, and X. Chen, “Age of processing: Age-driven status sampling and processing offloading for edge-computing-enabled real-time iot applications,” IEEE Internet of Things Journal, vol. 8, no. 19, pp. 14471–14484, 2021.
[16] S. Jayanth and R. Bhat, “Age of processed information minimization over fading multiple access channels,” IEEE Transactions on Wireless Communications, pp. 1–1, 2022.
[17] X. He, S. Wang, X. Wang, S. Xu, and J. Ren, “Age-based scheduling for monitoring and control applications in mobile edge computing systems,” in IEEE INFOCOM 2022 - IEEE Conference on Computer Communications, 2022, pp. 1009–1018.
[18] C. Li, S. Li, Y. Chen, Y. T. Hou, and W. Lou, “Aoi scheduling with maximum thresholds,” in 39th IEEE Conference on Computer Communications, INFOCOM 2020, Toronto, ON, Canada, July 6-9, 2020. Toronto, ON, Canada: IEEE, 2020, pp. 436–445. [Online]. Available: https://doi.org/10.1109/INFOCOM4043.2020.9155514
[19] I. Kadota, A. Sinha, and E. Modiano, “Scheduling algorithms for optimizing age of information in wireless networks with throughput constraints,” IEEE/ACM Transactions on Networking, vol. 29, no. 1, pp. 1359–1372, 2019.
[20] Y.-P. Hsu, E. Modiano, and L. Duan, “Scheduling algorithms for minimizing age of information in wireless broadcast networks with random arrivals,” IEEE Transactions on Mobile Computing, vol. 19, no. 12, pp. 2903–2915, 2020.
[21] I. Kadota and E. H. Modiano, “Minimizing the age of information in wireless networks with stochastic arrivals,” IEEE Transactions on Mobile Computing, vol. 20, pp. 1173–1185, 2021.
[22] C. Nikunram, W. Meesena, S. J. Turner, and S. Supittayapornpong, “Minimizing age of processed information in wireless networks,” 2022. [Online]. Available: https://arxiv.org/abs/2210.00956
[23] M. Grant and S. Boyd, “CVX: Matlab software for disciplined convex programming, version 2.1,” http://cvxr.com/cvx, Mar. 2014.
[24] S. Diamond and S. Boyd, “Cvxpy: A python-embedded modeling language for convex optimization,” Journal of Machine Learning Research, vol. 17, no. 83, pp. 1–5, 2016. [Online]. Available: http://jmlr.org/papers/v17/15-408.html