On Scheduling with AoI Violation Tolerance

Chengzhang Li Qingyu Liu Shaoran Li Yongce Chen Y. Thomas Hou Wenjing Lou
Virginia Tech, Blacksburg, VA, USA

IEEE INFOCOM 2021 - IEEE Conference on Computer Communications | 978-1-6654-0325-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/INFOCOM42981.2021.9488685

***Abstract*—We study an *Age of Information* (AoI) scheduling problem where AoI for each source at the base station (BS) can tolerate occasional violations, which we define as a *violation tolerance constraint*. The problem is to determine whether a set of users with given AoI deadlines, tolerance rates, and packet loss rates (due to each source's channel condition) is schedulable, and if so find a feasible scheduler. We study two cases: (i) the stable tolerant case where the tolerance rate is higher than the packet loss rate for all sources; (ii) the unstable tolerant case where the tolerance rate is lower than the packet loss rate for at least one source. For stable tolerant case, we design an algorithm called *stable tolerant scheduler* (STS), which can find a feasible scheduler for any network when system load is no greater than $\ln 2$. For unstable tolerance case, we develop *unstable tolerant scheduler* (UTS) and identify a schedulability condition for it. Through extensive simulations, we show that STS and UTS match our theoretical results.**

I. Introduction

Age of Information (AoI) is a novel application-layer metric for latency that was initially conceived by Kaul *et al.* [1], [2]. AoI is defined as the elapsed time for a sample (stored at a particular location, e.g., edge or cloud) between current time (now) and the time when the sample was first generated (collected) at its source. AoI measures the freshness of the sample from the time when it was initially generated. It is a latency metric at the application layer (in contrast to traditional network/link layer latency).

Since its inception, there has been active research on AoI (see an online bibliography in [3]), which includes a wide range of problems. An important class of problems that has attracted much attention is how to design schedulers to minimize AoI [4]–[29]. However, for many applications, we may not care much about AoI minimization (average or peak-wise). But rather, we may be more interested in whether or not the scheduler can meet their AoI requirements (or deadlines), which may vary widely across different source nodes. Further, for many applications, such a deadline requirement doesn't need to be hard (deterministic guarantee), as occasional violations of deadlines are usually not fatal and can be tolerated, as long as the long term violation rate is kept under a threshold. Unfortunately, this class of problems has not been studied in the literature, partly due to the level of difficulties involved in the analysis.

In this paper, we investigate this problem by considering a data collection network, where each source node collects information and forwards it to a base station (BS) through a shared wireless channel. At the BS, we assume there is an AoI deadline for information from each source and a tolerance rate for violating this deadline. Further, we assume there is a packet loss rate on the wireless link between each source and the BS. Denote the vectors (for all source nodes) of AoI thresholds, tolerance rates and packet loss rates as $\boldsymbol{d}$, $\boldsymbol{\epsilon}$ and $\boldsymbol{p}$. We are interested in addressing the following two questions: (i) For a given triple $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$, does there exist a feasible scheduler? (ii) If a feasible scheduler exists, then find it.

The main contributions of this paper are the following:

- We define a system load metric, denoted by $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$, that seamlessly fuses AoI requirement $\boldsymbol{d}$, channel condition $\boldsymbol{p}$ and violation tolerance $\boldsymbol{\epsilon}$ together. We show that $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ is schedulable only if $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq 1$.
- We address the scheduling problem by exploring the relationship between $\boldsymbol{\epsilon}$ and $\boldsymbol{p}$. In the case where $\epsilon_i \geq p_i$ for all i's, which we call *stable tolerant* case, we present *stable tolerant scheduler* (STS), which can find a feasible scheduler as long as $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq \ln 2$.
- STS is based on three key results. (i) We introduce the notion of *Almost Uniform Scheduler* (AUS), which follows a uniform or nearly uniform transmission schedule for each source. AUS serves as a key construct in our scheduler design throughout this paper. We also present a necessary and sufficient condition for an AUS to be feasible. (ii) We consider a special case of $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ with *step-down* rate vector, and present a procedure that can construct a feasible AUS for any $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ with a step-down rate vector as long as $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq 1$. (iii) For a general $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ that does not have a step-down rate vector, we present a dynamic programming (DP) solution to map it to a step-down rate vector, for which we can construct a feasible AUS.
- In the case where $\epsilon_i < p_i$ for some i's, which we call *unstable tolerant* case, we follow the same design roadmap (with modifications along the way) of STS and present a low-complexity algorithm called *Unstable Tolerant Scheduler* (UTS). We also give a sufficient condition for UTS to find a feasible scheduler for unstable tolerant $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$.

II. System Model

We consider a wireless network for data collection. We assume N source nodes and one base station (BS). Each source collects information from the environment and forwards it to the BS through a shared uplink wireless channel (see Fig. 1). Assume time is equally slotted and each source takes a new sample at the beginning of each time slot. Due to potential interference, at most one sample can be transmitted within one time slot. For each transmission, the source will forward its freshest (latest generated) sample to the BS. A scheduler (denoted by π) is needed to choose one source for transmission

978-1-6654-0325-2/21/$31.00 ©2021 IEEE

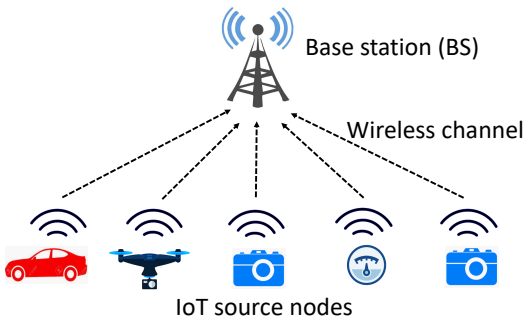

Figure 1: System model: N source nodes collect information and forward it to the BS.

at the beginning of each time slot. Denote $\pi_i(t) \in \{0,1\}$ as the scheduling decision for source i ($i = 1, 2, \cdots, N$) at time t ($t = 0, 1, 2, \cdots$), i.e.,

$$\pi_i(t) = \begin{cases} 1, & \text{if source } i \text{ is scheduled at time } t, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Clearly, the following must hold for any scheduler π:

$$\sum_{i=1}^{N} \pi_i(t) \leq 1, \ t = 0, 1, 2, \cdots. \tag{2}$$

The success (or failure) for transmitting the sample depends on the wireless channel condition. Denote $q_i(t)$ as an indicator function of the channel for node i at t, i.e., $q_i(t) = 1$ if the channel is under good condition and the transmission from source i will be successful, and 0 otherwise. Denote p_i as the probability of $q_i(t) = 0$, i.e., $\mathbb{P}\{q_i(t) = 0\} = p_i$. Then $\mathbb{P}\{q_i(t) = 1\} = 1 - p_i$. Note that p_i is location dependent (on source node i). Also, we assume p_i is time-invariant.

The BS only stores the freshest sample it has received from each source. Denote $U_i(t)$ as the generation time of the sample stored at the BS from source i at time t. Then the AoI of source i (at the BS) at time t, denoted as $A_i(t)$, is defined as the elapsed time between the current time t and the sample generation time $U_i(t)$, i.e.,

$$A_i(t) = t - U_i(t), \ t = 0, 1, 2, \cdots. \tag{3}$$

If the source i is not scheduled for transmission at time t (i.e., $\pi_i(t) = 0$), by definition of AoI, at time $(t+1)$ we have $A_i(t+1) = A_i(t) + 1$. We assume then the generation time of a new sample occurs at the beginning of each time slot t. Then if source i is scheduled for transmission at time t (i.e., $\pi_i(t) = 1$), then: (i) if the sample is successfully received by the BS (i.e., $q_i(t) = 1$), then at time $(t+1)$, we have $U_i(t+1) = t$, and $A_i(t+1) = 1$; (ii) if the sample is not received (i.e., $q_i(t) = 0$), then at time $(t+1)$ we have $A_i(t+1) = A_i(t) + 1$. Combining the two cases, we have:

$$A_i(t+1) = \begin{cases} 1, & \text{if } \pi_i(t) \cdot q_i(t) = 1, \\ A_i(t) + 1, & \text{otherwise.} \end{cases} \tag{4}$$

III. Problem Statement

In this paper we assume there is an *AoI threshold* d_i (d_i is a positive integer) and a *tolerance rate* $\epsilon_i \in [0, 1)$ for each source i. Due to the difference in types of application, d_i and ϵ_i may vary among different sources. Our goal is to design a scheduler such that for each source i the fraction of time slots when its AoI exceeds its threshold d_i is no greater than its tolerance rate ϵ_i. More formally, we say a scheduler π is *feasible* if it satisfies the following *violation tolerance* constraint:

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} [A_i(t) > d_i] \leq \epsilon_i, \ i = 1, 2, \cdots, N, \tag{5}$$

where "$[\cdot]$" is Iverson bracket, returning 1 if the inside statement is true and 0 otherwise [30].

Denote $\boldsymbol{d} = [d_1, d_2, \cdots, d_N]$ as the vector of AoI thresholds for all N sources, $\boldsymbol{\epsilon} = [\epsilon_1, \epsilon_2, \cdots, \epsilon_N]$ as the vector of all N tolerance rates, and $\boldsymbol{p} = [p_1, p_2, \cdots, p_N]$ as the vector of all N packet loss probabilities. We say an ordered triple $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ is *schedulable* if there exists at least one feasible scheduler for it. The problem we want to study in this paper is to determine whether or not a given ordered triple $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ is schedulable; and if so, how to find a feasible scheduler.

It turns out that our answers to the above two questions heavily depend on the relationship between $\boldsymbol{\epsilon}$ and $\boldsymbol{p}$. For now, let's define two cases: (i) if $\epsilon_i \geq p_i$ for *all* $i = 1, 2, \cdots, N$, then we say $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ is *stable tolerant*; (ii) if there exists at least some $i = 1, 2, \cdots, N$ such that $\epsilon_i < p_i$, the we say $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ is *unstable tolerant*. Note that the two cases are complimentary in the entire space for $(\boldsymbol{\epsilon}, \boldsymbol{p})$. Thus, it is sufficient to examine these two cases.

IV. System Load

Recall that one of our goals is to determine schedulability of an ordered triple $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$. In this section we derive a necessary condition for $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ to be schedulable. We define a concept called *system load*, denoted by $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$, and show that $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ is schedulable only if $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq 1$.

Under a scheduler π, we define the average scheduling rate r_i for each source i as

$$r_i = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \pi_i(t), \tag{6}$$

which represents the fraction of time slots in which source i is scheduled for transmission. Clearly, based on (2), we have

$$\sum_{i=1}^{N} r_i \leq 1. \tag{7}$$

Since the success probability for each transmission from source i is $(1 - p_i)$, the fraction of time slots in which its sample is successfully received by the BS is $r_i(1 - p_i)$.

Recall a scheduler π is feasible if it can ensure $A_i(t) \leq d_i$ for at least a fraction of $(1 - \epsilon_i)$ of time slots. Note that when a new update arrives at the BS, it will bring the AoI (for that source) at the BS immediately down to 1. This

effectively ensures that the AoI for that source will not exceed its threshold for up to d_i time slots. So for a feasible scheduler, the fraction of the time slots when a sample from source i is successfully received by the BS must be no less than $\frac{1-\epsilon_i}{d_i}$. That is,

$$r_i(1-p_i) \geq \frac{1-\epsilon_i}{d_i},\ i=1,2,\cdots,N, \tag{8}$$

which is equivalent to

$$r_i \geq \frac{1-\epsilon_i}{(1-p_i)d_i},\ i=1,2,\cdots,N. \tag{9}$$

Denote the RHS of (9) as r_i^{lb}, which is the lower bound of r_i. We have

$$r_i^{\text{lb}} = \frac{1-\epsilon_i}{(1-p_i)d_i}. \tag{10}$$

Clearly, under a feasible scheduler we must have $r_i \geq r_i^{\text{lb}}$ for each i. Considering (7), for a feasible scheduler we must have

$$\sum_{i=1}^{N} r_i^{\text{lb}} \leq 1. \tag{11}$$

We define the sum of r_i^{lb}'s as *system load* w.r.t. $(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$, which is written as

$$l(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p}) = \sum_{i=1}^{N} r_i^{\text{lb}} = \sum_{i=1}^{N} \frac{1-\epsilon_i}{(1-p_i)d_i}. \tag{12}$$

The following lemma shows a necessary condition for $(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$ to be schedulable.

Lemma 1 *An ordered triple $(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$ is schedulable only if $l(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p}) \leq 1$.*

The lemma follows directly from the discussions in this section.

V. The Stable Tolerant Case

In this section we will study scheduler design for the stable tolerance case, i.e., the case when $\epsilon_i \geq p_i$ for all $i = 1,2,\cdots,N$. First, we introduce *Almost Uniform Scheduler* (AUS) and offer a necessary and sufficient condition for an AUS to be feasible for $(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$. Then we consider a special case of $(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$'s and show how to construct an AUS for this special case. Finally we extend the same procedure from the special case to the general case of $(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$.

A. Almost Uniform Scheduler

We first define what we call a "cyclic" scheduler.

Definition 1 *We say a scheduler π is cyclic if there exists a cycle length c such that $\pi_i(t) = \pi_i(t+c)$ for all $i = 1,2,\cdots,N$ and $t \geq 0$.*

In other words, a scheduler is cyclic if its scheduling decision over a frame of length c time slots repeats itself over time. Since a cyclic scheduler is most easy to understand and implementation friendly, we will focus our attention on this class of schedulers.

Denote τ_i^k as the time slot when the k-th sample from source i is scheduled for transmission under scheduler π. Clearly we have $\pi_i(\tau_i^k) = 1$ for each $k = 1,2,\cdots$. Denote T_i^k as the time interval (in number of time slots) between the k-th and the $(k+1)$-th transmission for source i. Then we have:

$$T_i^k = \tau_i^{k+1} - \tau_i^k. \tag{13}$$

Clearly, when there is only one transmission for source i within a cycle c, we have $T_i^k = c$. When there are more than one transmissions within c, we have $T_i^k < c$.

Definition 2 *A cyclic scheduler π is an Almost Uniform Scheduler (AUS) if for each source i, there exists an integer b_i such that T_i^k is either b_i or (b_i+1) for any $k \geq 1$.*

By definition, if a scheduler π is an AUS, then the intervals between two consecutive transmissions of a source do not differ by more than 1 time slot. As an example, consider three sources A, B, and C. Denote "()" as a scheduling decision for one cycle. Then for scheduler $(ABACB)$ (with $c=5$), we have $T_A^1 = 2$, $T_A^2 = 3$, $\cdots$, $T_B^1 = 3$, $T_B^2 = 2$, $\cdots$, $T_C^1 = 5$. We have $b_A = 2$, $b_B = 2$ and $b_C = 5$ and this scheduler is AUS. In contrast, for scheduler $(AABACB)$ (with $c=6$), we have $T_A^1 = 1$, $T_A^2 = 2$, $T_A^3 = 3$, $\cdots$, $T_B^1 = 3$, $T_B^2 = 3$, $\cdots$, $T_C^1 = 6$. Since $T_A^1 = 1$ and $T_A^3 = 3$, we cannot find a b_A per AUS definition. So this scheduler is not AUS.

The following theorem shows that if a scheduler π is AUS, then there exists a necessary and sufficient condition for it to be feasible for $(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$.

Theorem 1 *In the stable tolerance case, an AUS π is feasible for $(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$ if and only if $r_i \geq r_i^{lb}$ for each $i = 1,2,\cdots,N$.*

Proof The proof for the "only if" part follows directly from Lemma 1. So our proof focuses on the "if" part, i.e., an AUS π is feasible if $r_i \geq r_i^{\text{lb}}$ for each $i = 1,2,\cdots,N$. In other words, we need to show that if $r_i \geq r_i^{\text{lb}}$, then (5) will be satisfied for source i.

By Definition 2, under AUS we have an integer b_i such that the interval length between any two consecutive transmissions of source i is either b_i or (b_i+1). We consider two cases.

- When $b_i < d_i$, we have $b_i + 1 \leq d_i$. Therefore, under AUS for any time slot t, within interval $[t-d_i, t-1]$ there will be at least one transmission for source i. Since the success probability for this transmission is $(1-p_i)$, for any t we have $\mathbb{P}\{A_i(t) \leq d_i\} \geq 1-p_i$. Therefore, we have

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T}[A_i(t) \leq d_i] \geq 1-p_i,$$

which is equivalent to

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T}[A_i(t) > d_i] \leq p_i.$$

Since $\epsilon_i \geq p_i$ (in the stable tolerance case), we have

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T}[A_i(t) > d_i] \leq \epsilon_i.$$

- When $b_i \geq d_i$, under AUS for any time slot t, within interval $[t, t+d_i-1]$ there must be at most one transmission for source i. Then for each t with $\pi_i(t)=1$, we have $\pi_i(t+1)=\pi_i(t+2)=\cdots=\pi_i(t+d_i-1)=0$, and $\mathbb{P}\{A_i(t+1)\leq d_i\}=\mathbb{P}\{A_i(t+2)\leq d_i\}=\cdots=\mathbb{P}\{A_i(t+d_i)\leq d_i\}=(1-p_i)$. Therefore, each scheduled transmission for source i will benefit the following d_i time slots, i.e., make the probability that the AoI is no greater than the threshold $(1-p_i)$ on the following d_i time slots. Considering the scheduling rate r_i, for a large time window T, the number of time slots when the AoI is no greater than the threshold is $Tr_i(1-p_i)d_i$, i.e,

$$\lim_{T\to\infty}\sum_{t=1}^{T}[A_i(t)\leq d_i]=Tr_i(1-p_i)d_i.$$

Dividing both sides by T, we have

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}[A_i(t)\leq d_i]=r_i(1-p_i)d_i,$$

which is equivalent to

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}[A_i(t)> d_i]=1-r_i(1-p_i)d_i.$$

Since $r_i\geq r_i^{\text{lb}}=\frac{1-\epsilon_i}{(1-p_i)d_i}$, we have

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}[A_i(t)> d_i]\leq 1-(1-p_i)d_i\cdot\frac{1-\epsilon_i}{(1-p_i)d_i}=\epsilon_i.$$

Combining both cases, we have completed the proof for the "if" part of the theorem. This completes our proof. ∎

B. Finding AUS for a Special Case of $(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$

In this section we study a special case of $(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$'s, for which we can construct an AUS when the system load $l(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$ is no greater than 1. We first introduce a concept called *step-down rate vector*. Denote $\boldsymbol{r}^{\text{lb}}$ as the lower bound rate vector, i.e., $\boldsymbol{r}^{\text{lb}}=[r_1^{\text{lb}},r_2^{\text{lb}},\cdots,r_N^{\text{lb}}]$, where r_i^{lb} is given in (10) for $i=1,2,\cdots,N$. Without loss of generality, we sort the elements in $\boldsymbol{r}^{\text{lb}}$ in non-increasing order, i.e., $r_1^{\text{lb}}\geq r_2^{\text{lb}}\geq\cdots\geq r_N^{\text{lb}}$.

Definition 3 *We say* $\boldsymbol{r}^{\text{lb}}$ *is a step-down rate vector if* $r_i^{lb}/r_{i+1}^{lb}\in\mathbb{N}^*$ *for* $i=1,2,\cdots,N-1$.

In other words, in a step-down rate vector $\boldsymbol{r}^{\text{lb}}$, the radio of an element over its succeeding element is a positive integer. For example, $\boldsymbol{r}^{\text{lb}}=[1/2,1/6,1/12,1/12,1/24]$ is a step-down vector.

Now we show how to construct an AUS that is feasible for a given $(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$ with a step-down rate vector $\boldsymbol{r}^{\text{lb}}$. We use an example to illustrate the main ideas. The complete pseudocode is given in Algorithm 1.

Example 1. Consider seven sources A, B, C, D, E, F and G with $(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})=([2,4,5,15,15,16,16],[0.4,0.4,0.25,0.25,0.25,0.2,0.2],[0.15,0.15,0.15,0.15,0.15,0.15,0.15])$. By (10), we have $\boldsymbol{r}^{\text{lb}}=[\frac{6}{17},\frac{3}{17},\frac{3}{17},\frac{1}{17},\frac{1}{17},\frac{1}{17},\frac{1}{17}]$, which is a step-down rate vector. Further, $l(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})=\frac{16}{17}<1$.

By Theorem 1, an AUS is feasible if $r_i\geq r_i^{\text{lb}}$ for each i. Since $l(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})<1$, we can set $r_i=r_i^{\text{lb}}/l(\boldsymbol{d},\boldsymbol{\epsilon},\boldsymbol{p})$ for each i, which still guarantees the AUS is feasible. We have $r_A=\frac{6}{16}$, $r_B=\frac{3}{16}$, $r_C=\frac{3}{16}$, $r_D=\frac{1}{16}$, $r_E=\frac{1}{16}$, $r_F=\frac{1}{16}$, and $r_G=\frac{1}{16}$. Since the smallest rate among the r_i's is $1/16$, we set the cycle length of the AUS to $c^{\text{AUS}}=16$. Denote N_i as the number of scheduled transmission for source i within one cycle. Since $N_i=r_i\cdot c^{\text{AUS}}$, we have $N_A=6$, $N_B=3$, $N_C=3$, $N_D=1$, $N_E=1$, $N_F=1$, and $N_G=1$.

To construct an AUS, we first consider a special case, which we call *Exact Uniform Scheduler* (EUS). We say a scheduler an EUS if for each source i, the interval between any two adjacent transmissions is exactly b_i.

Now we continue with our example. Since $N_A=6$, we propose to first construct an EUS with $c=\lceil\frac{c^{\text{AUS}}}{N_A}\rceil\cdot N_A=3\cdot 6=18$. This EUS has $18-16=2$ more time slots than our AUS. We will take care of these 2 empty slots in the last step.

First, we lay out an unassigned EUS cycle with $c=18$. Under each slot in the cycle, we label its position with an index number.

π	(□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□)
t	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Consider source A. Since $N_A=6$, in the EUS we have $b_A=18/6=3$. We allocate the first empty time slot to A, along with every 3 time slots after it.

π	(A	□	□	A	□	□	A	□	□	A	□	□	A	□	□	A	□	□)
t	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Then we consider the second source B. Since $N_B=3$, in the EUS we have $b_B=18/3=6$. Among all the empty time slots, the smallest elapsed time slots following a transmission of A are: $t=2,5,8,11,14,17$. We allocate the first one among these possibilities, i.e., $t=2$, to source B, along with every $b_B=6$ slots after it.

π	(A	B	□	A	□	□	A	B	□	A	□	□	A	B	□	A	□	□)
t	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Then we consider source C. Since $N_C=3$, in the EUS we have $b_C=18/3=6$. Among all the empty time slots, the smallest elapsed time after a transmission of A is 1 time slot: i.e., $t=5,11,17$ have this smallest elapsed time. Among time slots $t=5,11,17$, the smallest elapsed time from a transmission of B is 3 time slot: i.e., $t=5,11,17$. We allocate the first one among these possibilities, i.e., $t=5$, to source C, along with every $b_C=6$ slots after it.

π	(A	B	□	A	C	□	A	B	□	A	C	□	A	B	□	A	C	□)
t	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Then we consider source D. Recall $N_D=1$. Among all the empty time slots, the smallest elapsed time from a transmission of A is 2 time slot: i.e., $t=3,6,9,12,15,18$. Among time slots $t=3,6,9,12,15,18$, the smallest elapsed time from a transmission of B is 1 time slot: i.e., $t=3,9,15$. Among time slots $t=3,9,15$, the smallest elapsed time from a

Algorithm 1 AUS Construction

Input: A step-down rate vector $\boldsymbol{r}^{\text{lb}}$ with $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq 1$.
Output: A feasible AUS.

1: Set $r_i = r_i^{\text{lb}}/l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ for each $1 \leq i \leq N$. Set $c^{\text{AUS}} = 1/r_N$, and $N_i = r_i \cdot c^{\text{AUS}}$ for each $1 \leq i \leq N$.
2: Construct an unassigned EUS cycle with $c = \lceil \frac{c^{\text{AUS}}}{N_1} \rceil \cdot N_1$.
3: Allocate the first empty slot to source 1, along with every c/N_1 slots after it.
4: **for** $i = 2, 3, \cdots, N$ **do**
5: Let $\mathcal{S}_0$ be the set of empty time slots.
6: **for** $j = 1, 2, \cdots, i-1$ **do**
7: Let $\mathcal{S}_j$ be the subset of $\mathcal{S}_{j-1}$ containing time slots corresponding to the smallest elapsed time after a transmission of source j in the cycle.
8: **end for**
9: Allocate the first time slot in $\mathcal{S}_{i-1}$ to source i, along with every c/N_1 slots after it.
10: **end for**
11: Remove the $(c - c^{\text{AUS}})$ unassigned time slots in the EUS cycle and output the resulting schedule.

transmission of C is 4 time slot: i.e., $t = 3, 9, 15$. Among these three possibilities, we allocate the first one, i.e., $t = 3$, to source D.

π	(A	B	D	A	C	□	A	B	□	A	C	□	A	B	□	A	C	□)
t	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Following the same token, we allocate time slots $t = 9$ to E, $t = 15$ to F, and $t = 6$ to G, respectively.

π	(A	B	D	A	C	G	A	B	E	A	C	□	A	B	F	A	C	□)
t	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

It is easy to understand that the above scheduler is an EUS. In the final step, we remove the two extra (empty) slots in the EUS and we have:

π	(A	B	D	A	C	G	A	B	E	A	C	A	B	F	A	C)
t	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Readers can easily verify that the final scheduler is AUS based on Definition 2. ∎

Algorithm 1 present a pseudocode based on the ideas in Example 1.

Lemma 2 *For any stable tolerant $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ with a step-down rate vector $\boldsymbol{r}^{lb}$ and $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq 1$, the scheduler constructed by Algorithm 1 is an AUS.*

Due to space limit, we give a proof sketch of Lemma 2.

A Proof Sketch Denote π as the scheduler constructed by Algorithm 1. We will prove that under π, for source i, there exists a b_i such that T_i^k is either b_i or $(b_i + 1)$. After the last allocation in Algorithm 1 (to source N), we have an EUS with $(c - c^{\text{AUS}})$ empty slots. By design, it can be shown that there are no two consecutive empty slots in this EUS. Further, based on steps 7 and 9 in Algorithm 1, it can be shown that for any source i, there exists an x_i such that within an interval between any two consecutive transmissions of source i, the number of empty slots is either x_i or $(x_i + 1)$. So after removing the $(c - c^{\text{AUS}})$ unassigned time slots in the EUS, the scheduler is an AUS. ∎

The following theorem says that the scheduler by Algorithm 1 is not only AUS, but also feasible for the given $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$.

Theorem 2 *For any stable tolerant $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ with a step-down rate vector $\boldsymbol{r}^{lb}$ and $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq 1$, the scheduler constructed by Algorithm 1 is feasible for $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$.*

Proof Denote π as the scheduler constructed by Algorithm 1. By Lemma 2, π is an AUS. Note that in Algorithm 1, we set $r_i = r_i^{\text{lb}}/l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ for π. Since $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq 1$, we have $r_i \geq r_i^{\text{lb}}$. Then by Theorem 1, π is feasible for $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$. ∎

Complexity We now analyze the time complexity of Algorithm 1. For each iteration of step 7 in Algorithm 1, we need to visit all unassigned time slots in the EUS, which has a complexity of $O(c)$. Since there are $O(N^2)$ such iterations in Steps 4–9 in Algorithm 1, the time complexity of all iterations is $O(N^2 c)$. Considering $c < 2c^{\text{AUS}} \leq 2/r_N^{\text{lb}}$, so $O(c) = O(1/r_N^{\text{lb}}) = O(d_{\max})$ where $d_{\max}$ is the largest element in $\boldsymbol{d}$. Here we assume both ϵ_i and p_i are smaller than 0.5. Therefore, the total complexity of Algorithm 1 is $O(N^2 c) = O(N^2 d_{\max})$.

C. Finding AUS for General Case of $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$

In this section, we consider the general case where $\boldsymbol{r}^{\text{lb}}$ may not be a step-down rate vector. Recall that by Theorem 1, an AUS is feasible if $r_i \geq r_i^{\text{lb}}$ for each i. So if we could find a step-down rate vector $\hat{\boldsymbol{r}} = [\hat{r}_1, \hat{r}_2, \cdots, \hat{r}_N]$ with $\sum_{i=1}^N \hat{r}_i \leq 1$ and $\hat{r}_i \geq r_i^{\text{lb}}$ for each i, then we can use Algorithm 1 to find an AUS with $r_i \geq \hat{r}_i \geq r_i^{\text{lb}}$ that is feasible for $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$.

To find $\hat{r}_i$ for $i = 1, 2, \cdots, N$, we can solve the following optimization problem. Again, we assume the elements in $\boldsymbol{r}^{\text{lb}}$ are sorted in non-increasing order, i.e., $r_1^{\text{lb}} \geq r_2^{\text{lb}} \geq \cdots \geq r_N^{\text{lb}}$.

$$\text{OPT: } \min_{\hat{\boldsymbol{r}}} \quad \sum_{i=1}^{N} \hat{r}_i$$
$$\text{s.t.} \quad \frac{\hat{r}_i}{\hat{r}_{i+1}} \in \mathbb{N}^*, \ i = 1, 2, \cdots, N-1,$$
$$r_i^{\text{lb}} \leq \hat{r}_i \leq 1, \ i = 1, 2, \cdots, N.$$

Denote the optimal solution to OPT as $\hat{\boldsymbol{r}}^*$, which is a step-down rate vector. If $\sum_{i=1}^N \hat{r}_i^* \leq 1$, then we can use Algorithm 1 (by setting $\boldsymbol{r}^{\text{lb}} = \hat{\boldsymbol{r}}^*$ in Algorithm 1) to find a feasible AUS for $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$. Otherwise, if $\sum_{i=1}^N \hat{r}_i^* > 1$, we cannot use Algorithm 1 to find a feasible scheduler.

A key step to solve OPT is to show that there exists some k $(k = 1, 2, \cdots, N)$ such that $\hat{r}_k^* = r_k^{\text{lb}}$ in the optimal solution $\hat{\boldsymbol{r}}^*$. Using this result, we can solve OPT by fixing $\hat{r}_k^* = r_k^{\text{lb}}$ for N times $(k = 1, 2, \cdots, N)$. For each k, since $\hat{r}_k^* = r_k^{\text{lb}}$ is fixed, we can divide OPT into to parts: (i) optimizing $\hat{r}_1, \hat{r}_2, \cdots, \hat{r}_{k-1}$; and (ii) optimizing $\hat{r}_{k+1}, \hat{r}_{k+2}, \cdots, \hat{r}_N$. Each part can be solved by dynamic

Algorithm 2 A DP Soluton to OPT

Input: A general $\boldsymbol{r}^{\text{lb}}$ with $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq 1$.
Output: An optimal solution to OPT, $\hat{\boldsymbol{r}}^*$

Set $\hat{r}_i^* = 1$ for each i. Set $J^* = N$.
for $k = 1, 2, \cdots, N$ **do**
  Set $\hat{r}_k = r_k^{\text{lb}}$ and $\mathcal{R}_k = \{\hat{r}_k\}$.
  if $k \geq 2$ **then**
    Set $s(\hat{r}_k, k) = 0$ and $\mathcal{R}_i = \{n\hat{r}_k \mid r_i^{\text{lb}} \leq n\hat{r}_k \leq 1, n \in \mathbb{N}^*\}$ for each $i = 1, 2, \cdots, k-1$.
    for $i = k-1, k-2, \cdots, 1$ **do**
      Compute $s(r, i) = r + \min_{\substack{r' \in \mathcal{R}_{i+1} \\ \& r/r' \in \mathbb{N}^*}} s(r', i+1)$ and $\text{prev}(r, i) = \arg\min_{\substack{r' \in \mathcal{R}_{i+1} \\ \& r/r' \in \mathbb{N}^*}} s(r', i+1)$ for each $r \in \mathcal{R}_i$
    end for
    Set $\hat{r}_1 = \arg\min_{r \in \mathcal{R}_1} s(r, 1)$.
    for $i = 2, 3, \cdots, k-1$ **do**
      Set $\hat{r}_i = \text{prev}(\hat{r}_{i-1}, i-1)$.
    end for
  end if
  if $k \leq N-1$ **then**
    Set $s(\hat{r}_k, k) = 0$ and $\mathcal{R}_i = \{\frac{\hat{r}_k}{n} \mid \frac{\hat{r}_k}{n} \geq r_i^{\text{lb}}, n \in \mathbb{N}^*\}$ for each $i = k+1, k+2, \cdots, N$.
    for $i = k+1, k+2, \cdots, N$ **do**
      Compute $s(r, i) = r + \min_{\substack{r' \in \mathcal{R}_{i-1} \\ \& r'/r \in \mathbb{N}^*}} s(r', i-1)$ and $\text{prev}(r, i) = \arg\min_{\substack{r' \in \mathcal{R}_{i-1} \\ \& r'/r \in \mathbb{N}^*}} s(r', i-1)$ for each $r \in \mathcal{R}_i$
    end for
    Set $\hat{r}_N = \arg\min_{r \in \mathcal{R}_N} s(r, N)$.
    for $i = N-1, N-2, \cdots, k+1$ **do**
      Set $\hat{r}_i = \text{prev}(\hat{r}_{i+1}, i+1)$.
    end for
  end if
  if $\sum_{i=1}^N \hat{r}_i < J^*$ **then**
    Set $\hat{r}_i^* = \hat{r}_i$ for $i = 1, 2, \cdots, N$. Set $J^* = \sum_{i=1}^N \hat{r}_i$.
  end if
end for
return $\hat{\boldsymbol{r}}^* = [\hat{r}_1^*, \hat{r}_1^*, \cdots, \hat{r}_N^*]$

programming (DP). Due to page limit, we will not elaborate on how we develop our solution. Instead, we present our solution algorithm (in pseudocode) in Algorithm 2. We encourage readers to study the algorithm and convince themselves that it solves OPT. It can be shown the time complexity of Algorithm 2 is $O(N^2 d_{\max}^2)$.

Now we are ready to present our complete procedure, which we call *Stable Tolerance Scheduler* (STS) in Algorithm 3. It can be shown that the time complexity of STS is $O(N^2 d_{\max}^2)$.

Note that Algorithm 3 isn't always able to find a feasible scheduler, even though $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq 1$. The next question to ask is: *Under what condition is Algorithm 3 guaranteed to to find a feasible scheduler?* The answer is given in the following lemma.

Algorithm 3 Stable Tolerant Scheduler (STS)

Input: $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ is stable tolerant, with $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq 1$
Output: A feasible scheduler π

1: Compute $\boldsymbol{r}^{\text{lb}}$ by (10) and sort its elements in non-increasing order.
2: Solve OPT by Algorithm 2 and obtain an optimal solution $\boldsymbol{r}^*$.
3: If $\sum_{i=1}^N r_i^* \leq 1$, use Algorithm 1 (by letting $\boldsymbol{r}^{\text{lb}} = \boldsymbol{r}^*$) to find a feasible scheduler π.

Lemma 3 *For any stable tolerant $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$, Algorithm 3 can always find a feasible scheduler if $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq \ln 2$.*

A Proof Sketch To prove Lemma 3, we assume STS cannot find a feasible scheduler for $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$. Then it is sufficient if we can prove $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) > \ln 2$.

Since STS cannot find a feasible scheduler for $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$, we must have $\sum_{i=1}^N \hat{r}_i^* > 1$. Since $\sum_{i=1}^N \hat{r}_i^*$ is the minimum, then for any $\hat{\boldsymbol{r}}$ that is feasible to OPT, we must have $\sum_{i=1}^N \hat{r}_i > 1$. For any $x > 0$, we define a vector $\hat{\boldsymbol{r}}^x$ as

$$\hat{r}_i^x = x \cdot 2^{\lceil \log_2(\frac{r_i^{\text{lb}}}{x}) \rceil}, \ i = 1, 2, \cdots, N. \tag{14}$$

It can be shown that $\hat{\boldsymbol{r}}^x$ is feasible to OPT, so we have $\sum_{i=1}^N \hat{r}_i^x > 1$. Define $g(x) = \frac{1}{x}$. We have $\int_{0.5}^1 g(x)dx = \ln 2$, so we have

$$\int_{0.5}^1 g(x) \cdot \sum_{i=1}^N \hat{r}_i^x > \ln 2. \tag{15}$$

It can be shown that the LHS of (15) equals to $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$. So we have $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) > \ln 2$. ∎

VI. From Stable Tolerant to Unstable Tolerant

In the last section, we studied the stable tolerant case (i.e., $\epsilon_i \geq p_i$ for all i's) and presented STS algorithm. In this section, we study the unstable tolerant case, i.e., $\epsilon_i < p_i$ for at least some i.

We will follow the same roadmap in the last section, with some necessary modifications along the way. We start with Theorem 1, which no longer holds in the unstable tolerance case. This is because $r_i \geq r_i^{\text{lb}}$ for each i is no longer a sufficient condition for an AUS to be feasible. As a fix to this problem, we propose *target rate* vector, defined as $\boldsymbol{r}^{\text{tar}} = [r_1^{\text{tar}}, r_2^{\text{tar}}, \cdots, r_N^{\text{tar}}]$. We wish to have an AUS be feasible if its $r_i \geq r_i^{\text{tar}}$.

In the unstable tolerant case, for those sources i's with $\epsilon_i \geq p_i$, we can just set $r_i^{\text{tar}} = r_i^{\text{lb}}$. So if $r_i \geq r_i^{\text{tar}}$ is satisfied, then (5) will be satisfied for these i's (from the proof of Theorem 1). Now we consider those i's with $\epsilon_i < p_i$. We want to find a r_i^{tar} such that: if $r_i \geq r_i^{\text{tar}}$ is satisfied, then (5) is also satisfied for these i's.

We consider one source i with $\epsilon_i < p_i$. Since it's difficult to analyze source i's AoI performance for a general AUS, we are going to consider an EUS, which guarantees a strict periodic transmission pattern for source i by relaxing certain transmission intervals with one extra slot. Let's consider an

EUS with transmission rate r_i' for node i. Clearly, if (5) can be satisfied under this EUS (with r_i' for source i), then under any AUS with $r_i > r_i'$ (5) is also satisfied for source i. So all we need to do is to find the smallest r_i' for an EUS such that (5) is satisfied, and choose it as r_i^{tar} for source i.

Under the EUS, over all time windows $[t-d_i, t-1]$ (for all $t > d_i$), the average number of transmissions within a window is $d_i r_i'$. It is also easy to see that the number of transmissions within any window does not differ by more than 1. Therefore for any t, within a time window $[t-d_i, t-1]$, there are either $\lfloor d_i r_i' \rfloor$ or $(\lfloor d_i r_i' \rfloor + 1)$ transmissions for source i. Denote a as the fraction of t's with $\lfloor d_i r_i' \rfloor$ transmissions in $[t-d_i, t-1]$. Then the fraction of t's with $(\lfloor d_i r_i' \rfloor + 1)$ transmissions in $[t-d_i, t-1]$ is $(1-a)$. We have

$$a \cdot \lfloor d_i r_i' \rfloor + (1-a) \cdot (\lfloor d_i r_i' \rfloor + 1) = d_i r_i' \,. \tag{16}$$

Solving the above for a, we have:

$$a = \lfloor d_i r_i' \rfloor + 1 - d_i r_i' \,. \tag{17}$$

At each time t, suppose there are k transmissions for source i within the window $[t-d_i, t-1]$. Then only if all of these k transmissions fail, we will have $A_i(t) > d_i$. That is, $\mathbb{P}\{A_i(t) > d_i\} = p_i^k$. Since $k = \lfloor d_i r_i' \rfloor$ with probability a and $k = \lfloor d_i r_i' \rfloor + 1$ with probability $(1-a)$, we have

$$\begin{aligned}\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} [A_i(t) > d_i] &= a \cdot p_i^{\lfloor d_i r_i' \rfloor} + (1-a) \cdot p_i^{\lfloor d_i r_i' \rfloor + 1} \\ &= (\lfloor d_i r_i' \rfloor + 1 - d_i r_i') \cdot p_i^{\lfloor d_i r_i' \rfloor} + (d_i r_i - \lfloor d_i r_i' \rfloor) \cdot p_i^{\lfloor d_i r_i' \rfloor + 1}.\end{aligned} \tag{18}$$

To satisfy (5), we need to have

$$(\lfloor d_i r_i' \rfloor + 1 - d_i r_i') \cdot p_i^{\lfloor d_i r_i' \rfloor} + (d_i r_i' - \lfloor d_i r_i' \rfloor) \cdot p_i^{\lfloor d_i r_i' \rfloor + 1} \le \epsilon_i. \tag{19}$$

It can be shown the LHS of (19) is monotonically decreasing w.r.t. r_i'. To ensure (19) holds, we can solve the following equation:

$$(\lfloor d_i x \rfloor + 1 - d_i x) \cdot p_i^{\lfloor d_i x \rfloor} + (d_i x - \lfloor d_i x \rfloor) \cdot p_i^{\lfloor d_i x \rfloor + 1} = \epsilon_i \,. \tag{20}$$

It can be shown that the solution to (20) is

$$x = \frac{(1-p_i)\lfloor \frac{\ln \epsilon_i}{\ln p_i} \rfloor + 1 - \epsilon_i p_i^{-\lfloor \frac{\ln \epsilon_i}{\ln p_i} \rfloor}}{(1-p_i) d_i}. \tag{21}$$

Then for any $r_i' \ge x$, (19) will hold. But under our EUS, $1/r_i'$ is the transmission interval length and must be an integer. So the smallest r_i' that ensures (19) holds is $r_i' = 1/\lfloor 1/x \rfloor$.

In summary, r_i^{tar} is given by

$$r_i^{\text{tar}} = \begin{cases} \frac{1}{\lfloor \frac{1}{x} \rfloor}, & \text{if } \epsilon_i < p_i, \\ r_i^{\text{lb}}, & \text{if } \epsilon_i \ge p_i, \end{cases} \tag{22}$$

where x is given by (21) and r_i^{lb} is given by (10).

Following the above discussions, we have the following lemma.

Algorithm 4 Unstable Tolerant Scheduler (UTS)

Input: $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ is unstable tolerant, with $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \le 1$
Output: A feasible scheduler π

1: Compute $\boldsymbol{r}^{\text{tar}}$ by (22), and sort its elements in non-increasing order.
2: Replace r_i^{lb}'s by r_i^{tar}'s in OPT, solve OPT by Algorithm 2, and obtain an optimal solution $\boldsymbol{r}^*$.
3: If $\sum_{i=1}^N r_i^* \le 1$, use Algorithm 1 (by letting $\boldsymbol{r}^{\text{lb}} = \boldsymbol{r}^*$) to find a feasible scheduler π.

Lemma 4 *In the unstable tolerant case, an AUS π is feasible for $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ if $r_i \ge r_i^{tar}$ for each $i = 1, 2, \cdots, N$, where r_i^{tar} is given in* (22)*.*

The proof of Lemma 4 is based on our discussion that once $r_i \ge r_i^{\text{tar}}$ for each $i = 1, 2, \cdots, N$, then (5) is satisfied for each i. Lemma 4 generalizes the results of the "if" part in Theorem 1. However, The "only if" part in Theorem 1 does not hold even if we generalize r_i^{lb} with r_i^{tar}. This is because we employed EUS relaxation when we define r_i^{tar} in Lemma 4 while there was no relaxation when we find r_i^{lb} in Theorem 1.

With Lemma 4 in hand, following the same roadmap in the last section, we then consider the special case of step-down rate vector $\boldsymbol{r}^{\text{tar}}$ (instead of $\boldsymbol{r}^{\text{lb}}$). We can use Algorithm 1 to construct an AUS for unstable tolerant $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ with step-down rate vector $\boldsymbol{r}^{\text{tar}}$ by merely replacing $\boldsymbol{r}^{\text{lb}}$ with $\boldsymbol{r}^{\text{tar}}$. We have the following lemma, which is similar to Theorem 2.

Lemma 5 *For any unstable tolerant $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ with a step-down rate vector $\boldsymbol{r}^{tar}$ and $\sum_{i=1}^N r_i^{tar} \le 1$, the scheduler constructed by Algorithm 1 (with $\boldsymbol{r}^{lb}$ replaced by $\boldsymbol{r}^{tar}$) is feasible for $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$.*

Then we consider the general unstable tolerant $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$'s that may not have a step-down rate vector $\boldsymbol{r}^{\text{tar}}$. Similar to what we did in the last section, we propose to find a step-down rate vector $\hat{\boldsymbol{r}}$ with $\sum_{i=1}^N \hat{r}_i \le 1$ and $\hat{r}_i \ge r_i^{\text{tar}}$ for each i. Again, we can solve an OPT, with r_i^{lb}'s replaced by r_i^{tar}'s.

Based on the above discussions, we present an algorithm, which we call *Unstable Tolerance Scheduler* (UTS), for the unstable tolerant case in Algorithm 4. The complexity of UTS is the same as the complexity of STS, which is $O(N^2 d_{\max}^2)$.

Following the same token as for Lemma 3, we have the following lemma for the unstable tolerant case.

Lemma 6 *For any unstable tolerant $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$, Algorithm 4 can always find a feasible scheduler if $\sum_{i=1}^N r_i^{tar} \le \ln 2$.*

VII. Numerical Results

In this section, we use simulations to validate our theoretical results and evaluate our algorithms.

A. Case Study

In this section we study two cases to validate that violation rate for each source is no greater than ϵ_i.

Table I: Stable tolerant case study

Source	1	2	3	4	5	6	7	8	9	10
d_i	6	10	10	12	15	16	18	20	25	30
p_i	0.1	0.05	0.15	0.05	0.1	0.05	0.1	0.05	0.2	0.1
ϵ_i	0.1	0.05	0.2	0.05	0.1	0.1	0.15	0.05	0.2	0.2
ρ_i	0.071	0.046	0.134	0.034	0.039	0.013	0.099	0.045	0.139	0.041

Table II: Unstable tolerant case study

Source	1-10	11-20	21-30	31-40	41-50	51-60	61-70	71-80	81-90	91-100
d_i	60	100	120	150	190	240	270	300	330	360
p_i	0.2	0.1	0.05	0.1	0.15	0.2	0.1	0.05	0.2	0.15
ϵ_i	0.1	0.05	0.15	0.2	0.1	0.05	0.15	0.1	0.2	0.1
$\rho_i \leq$	0.041	0.041	0.051	0.079	0.076	0.041	0.091	0.039	0.143	0.089

1) Stable Tolerant Case: We consider $N = 10$ source nodes, with parameters d_i's, p_i's and ϵ_i's for $i = 1, 2, \cdots, 10$ given in Table I. The load is $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) = 0.742$. As we can see, we have $\epsilon_i \geq p_i$ for each i, so $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ is stable tolerant and we will use STS to find a scheduler for it.

In STS, we first compute $\boldsymbol{r}^{\text{lb}} = [0.167\ 0.100\ 0.094\ 0.083\ 0.067\ 0.059\ 0.052\ 0.050\ 0.040\ 0.030]$. Then we solve OPT by Algorithm 2 and obtain an optimal solution $\boldsymbol{r}^* = [0.210\ 0.105\ 0.105\ 0.105\ 0.105\ 0.105\ 0.052\ 0.052\ 0.052\ 0.052]$.

We have $\sum_{i=1}^{N} r_i^* = 0.944 \leq 1$, so we use Algorithm 1 (to find a feasible scheduler π:

$$\pi = (ABDFJACEGABDFIACEH).$$

Readers can verify π is an AUS.

Under this AUS π, we simulate $T = 1,000,000$ time slots. Denote the measured (observed) violation rate for source i as $\rho_i = \frac{1}{T}\sum_{t=1}^{T}[A_i(t) > d_i]$ over these $1,000,000$ time slots. In Table I (last row), we show the measured ρ_i for each $i = 1, 2, \cdots, 10$. We find that $\rho_i < \epsilon_i$ for each i, so our AUS π is indeed feasible for the given $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$..

2) Unstable Tolerant Case: We now consider $N = 100$ source nodes as shown in Table II. The load is $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) = 0.658$. As we can see, this $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ is unstable tolerant (e.g., $p_i > \epsilon_i$ for sources 1–10, among others). So we will use UTS to find a scheduler for it. In UTS, we first compute $\boldsymbol{r}^{\text{tar}}$, then solve OPT by Algorithm 2, and obtain an optimal solution $\boldsymbol{r}^*$. We have $\sum_{i=1}^{N} r_i^* = 0.976 < 1$, so we use Algorithm 1 to find a feasible scheduler π (with cycle length $c = 240$). Under π, we simulate $T = 10,000,000$ time slots, and calculate the violation rate $\rho_i = \frac{1}{T}\sum_{t=1}^{T}[A_i(t) > d_i]$ for all i's. For each group of 10 nodes, we list the largest measured violation rate ρ_i in the group (e.g., for $i =$ 1–10, 0.041 is the largest measured violation rate among $\rho_1, \rho_2, \cdots, \rho_{10}$) in the last row of Table II. We can see that $\rho_i < \epsilon_i$ for all $i = 1, 2, \cdots, 100$. So π is indeed feasible for the given $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$.

B. Impact of Tolerance Rate ϵ

In this section we study the impact of tolerance rate ϵ for algorithms STS and UTS. We consider 50 sources, i.e., $N = 50$. For ease of presentation, we assume all sources have the same p_i and the same ϵ_i, i.e., $p_i = p$ and $\epsilon_i = \epsilon$ for all i's. We assume $p = 0.1$ and vary ϵ from 0.04 to 0.3, i.e., from a value smaller than p to a value greater than p. When $\epsilon < p$, we use UTS and when $\epsilon \geq p$, we use STS.

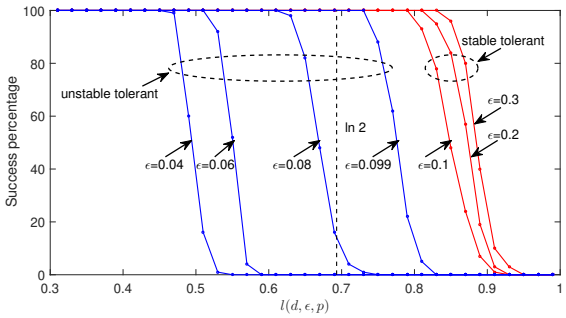

Figure 2: Success percentage for STS/UTS to find a feasible scheduler when $p = 0.1$.

We assume $d_i \in \{10, 20, \cdots, 300\}$. For each ϵ, we randomly generate 100 different $\boldsymbol{d}$'s for each load interval $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \in (0.3, 0.32], (0.32, 0.34], \cdots, (0.98, 1]$. For each load interval we run the algorithm and calculate the percentage of $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$'s for which our algorithm (either STS or UTS, depending on ϵ) successfully finds a feasible scheduler. The results are shown in Fig. 2. As we can see, as ϵ increases, the corresponding curve shifts to the right, meaning that for the same success percentage in finding a feasible scheduler, the network can take a higher load when ϵ increases. Alternatively, for the same load, the success percentage in finding a feasible scheduler increases with ϵ. We also show the load guarantee for STS, $\ln 2$ in the figure. We can see in the stable tolerant case, when $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq \ln 2$, the success percentage of STS is 100% (i.e., when ϵ is 0.1, 0.2, and 0.3), which confirms Lemma 3.

VIII. Conclusions

This paper investigates an important AoI scheduling problem that considers tolerance of occasional deadline violations. To the best of our knowledge, this class of problem has not been studied in the AoI research literature. We first present system load $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ that seamlessly integrates AoI requirement $\boldsymbol{d}$, channel condition $\boldsymbol{p}$ and violation tolerance $\boldsymbol{\epsilon}$ into one metric, and show that $(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p})$ is schedulable only if $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq 1$. Then we address the scheduling problem by exploring the relationship between $\boldsymbol{\epsilon}$ and $\boldsymbol{p}$ and identify two cases: stable tolerant and unstable tolerant. For the stable tolerant case, we present STS, which can find a feasible scheduler as long as $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq \ln 2$. For the unstable tolerant case, we present UTS, for which we give a sufficient condition to find a feasible scheduler. Simulation results show that the feasible scheduler found by our algorithm, either STS or UTS, can always guarantee $\boldsymbol{\epsilon}$. Further, STS does offer a guarantee of finding a feasible scheduler if $l(\boldsymbol{d}, \boldsymbol{\epsilon}, \boldsymbol{p}) \leq \ln 2$.

Acknowledgments

This research was supported in part by ONR under MURI Grant N00014-19-1-2621, Virginia Commonwealth Cyber Initiative (CCI), and Virginia Tech Institute for Critical Technology and Applied Science.

REFERENCES

[1] S. Kaul, M. Gruteser, V. Rai, and J. Kenney, "Minimizing Age of Information in Vehicular Networks," in *Proc. IEEE SECON,* pp. 350–358, Salt Lake City, UT, USA, June 27–30, 2011.

[2] S. Kaul, R. Yates, and M. Gruteser, "Real-Time Status: How Often Should One Update?" in *Proc. IEEE INFOCOM,* pp. 2731–2735, Orlando, FL, USA, Mar. 25–30, 2012.

[3] Y. Sun, "A Collection of Recent Papers on the Age of Information," available at http://www.auburn.edu/%7eyzs0078

[4] Y. Hsu, E. Modiano, and L. Duan, "Age of Information: Design and Analysis of Optimal Scheduling Algorithms," in *Proc. IEEE ISIT,* pp. 561–565, Archen, Germany, June 25–30, 2017.

[5] I. Kadota, E. Uysal-Biyikoglu, R. Singh, and E. Modiano, "Minimizing the Age of Information in Broadcast Wireless Networks," in *Proc. Allerton Conference,* pp. 844–851, Monticello, IL, USA, Sept. 27–30, 2016.

[6] J. Zhong, R.D. Yates, and E. Soljanin, "Two Freshness Metrics for Local Cache Refresh," in *Proc. IEEE ISIT,* pp. 1924–1928, Vail, CO, USA, June 17–22, 2018.

[7] A.M. Bedewy, Y. Sun, and N.B. Shroff, "Optimizing Data Freshness, Throughput, and Delay in Multi-Server Information-Update Systems," in *Proc. IEEE ISIT,* pp. 2569–2573, Barcelona, Spain, July 10–15, 2016.

[8] Y. Sun, E. Uysal-Biyikoglu, and S. Kompella, "Age-Optimal Updates of Multiple Information Flows," in *Proc. IEEE INFOCOM Workshops – Age of Information Workshop,* pp. 136–141, Honolulu, HI, USA, April 15–19, 2018.

[9] C. Li, S. Li, and Y.T. Hou, "A General Model for Minimizing Age of Information at Network Edge," in *Proc. IEEE INFOCOM,* pp. 118–126, Paris, France, Apr. 29 – May 2, 2019.

[10] C. Li, Y. Huang, Y. Chen, B. Jalaian, Y.T. Hou, and W. Lou, "Kronos: A 5G Scheduler for AoI Minimization under Dynamic Channel Conditions," in *Proc. IEEE ICDCS,* pp. 1466–1472, Dallas, TX, USA, July 7–9, 2019.

[11] Q. He, D. Yuan, and A. Ephremides, "Optimal Link Scheduling for Age Minimization in Wireless Systems," *IEEE Trans. on Information Theory,* vol. 64, issue 7, pp. 5381–5394, July 2018.

[12] C. Joo and A. Eryilmaz, "Wireless Scheduling for Information Freshness and Synchrony: Drift-based Design and Heavy-Traffic Analysis," in *Proc. WiOpt,* pp. 1–8, Paris, France, May 15–19, 2017.

[13] N. Lu, B. Ji, and B. Li, "Age-based Scheduling: Improving Data Freshness for Wireless Real-Time Traffic," in *Proc. ACM MobiHoc,* pp. 191–200, Los Angeles, CA, USA, June 26–29, 2018.

[14] R. Talak, S. Karaman, and E. Modiano, "Optimizing Information Freshness in Wireless Networks under General Interference Constraints," in *Proc. ACM MobiHoc,* pp. 61–70, Los Angeles, CA, USA, June 26–29, 2018.

[15] A.M. Bedewy, Y. Sun, and N.B. Shroff, "Age-Optimal Information Updates in Multihop Networks," in *Proc. IEEE ISIT,* pp. 576–580, Archen, Germany, June 25–30, 2017.

[16] R. Talak, S. Karaman, and E. Modiano, "Minimizing Age-of-Information in Multi-Hop Wireless Networks," in *Proc. Allerton Conference,* pp. 486–493, Monticello, IL, USA, Oct. 3–6, 2017.

[17] Z. Qian, F. Wu, J. Pan, K. Srinivasan, and N.B. Shroff, "Minimizing Age of Information in Multi-channel Time-sensitive Information Update Systems," in *Proc. IEEE INFOCOM,* pp. 446–455, online conference, July 6–9, 2020.

[18] J.P. Champati, R.R. Avula, T.J. Oechtering, and J. Gross, "On the Minimum Achievable Age of Information for General Service-Time Distributions," in *Proc. IEEE INFOCOM,* pp. 456–465, online conference, July 6–9, 2020.

[19] J. Lou, X. Yuan, S. Kompellay, and N. Tzeng, "AoI and Throughput Tradeoffs in Routing-aware Multi-hop Wireless Networks," in *Proc. IEEE INFOCOM,* pp. 476–485, online conference, July 6–9, 2020.

[20] A.M. Bedewy, Y. Sun, R. Singh, and N.B. Shroff, "Optimizing Information Freshness Using Low-Power Status Updates via Sleep-Wake Scheduling," in *Proc. ACM MobiHoc,* pp. 51–60, online conference, Oct. 11–14, 2020.

[21] A. Maatouk, S. Kriouile, M. Assaad, and A. Ephremides, "Asymptotically Optimal Scheduling Policy for Minimizing the Age of Information," in *Proc. IEEE ISIT,* pp. 1747–1752, online conference, June 21–26, 2020.

[22] T. Park, W. Saad, and B. Zhou, "Centralized and Distributed Age of Information Minimization with Non-linear Aging Functions in the Internet of Things," *IEEE Internet of Things Journal,* to appear, DOI: 10.1109/JIOT.2020.3046448.

[23] B. Zhou and W. Saad, "Minimum Age of Information in the Internet of Things with Non-Uniform Status Packet Sizes," *IEEE Trans. on Wireless Communication,* vol. 19, issue. 3, pp. 1933–1947, March 2020.

[24] H. Ma, S. Zhou, X. Zhang, and L. Xiao, "Transmission Scheduling for Multi-loop Wireless Networked Control Based on LQ Cost Offset," in *Proc. IEEE INFOCOM Workshops – Age of Information Workshop,* online conference, July 6, 2020.

[25] Y. Wang and W. Chen, "Adaptive Power and Rate Control for Real-time Status Updating over Fading Channels," *IEEE Trans. on Wireless Communication,* to appear, DOI: 10.1109/TWC.2020.3047426.

[26] B. Sombabu and S. Moharir, "Age-of-Information Based Scheduling for Multi-Channel Systems," *IEEE Trans. on Wireless Communication,* vol. 19, issue. 7, pp. 4439–4448, July 2020.

[27] H.H. Yang, A. Arafa, T.Q.S. Quek, and V. Poor, "Optimizing Information Freshness in Wireless Networks: A Stochastic Geometry Approach," *IEEE Trans. on Mobile Computing,* to appear, DOI: 10.1109/TWC.2020.3047426.

[28] J. Sun, Z. Jiang, B. Krishnamachari, S. Zhou, and Z. Niu, "Closed-Form Whittle's Index-Enabled Random Access for Timely Status Update," *IEEE Trans. on Communications,* vol. 68, issue. 3, pp. 1538–1551, March 2020.

[29] 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 J. Selected Areas in Commun.,* vol. 38, issue. 5, pp. 854–868, May 2020.

[30] R.L. Garham, D.E. Knuth, and O. Patashnik, "Concrete Mathematics," Chapter 2, *Addison-Wesley,* 1989.