Minimizing Age of Information Under General Models for IoT Data Collection

Chengzhang Li, *Student Member, IEEE*, Shaoran Li, *Student Member, IEEE*, Yongce Chen, *Student Member, IEEE*, Y. Thomas Hou, *Fellow, IEEE*, and Wenjing Lou, *Fellow, IEEE*

***Abstract*—A critical component in IoT infrastructure and applications is data collection at network edge. Recently, a new metric, called age of information (AoI), has become popular to quantify the freshness of information collected at network edge. AoI research is still in infancy and most efforts assume overly simplified models in their investigation, making their results far from useful when addressing practical problems in IoT applications. In this article, we close this gap by considering more general models for AoI research that are more relevant in the real world. Specifically, we consider general and heterogeneous sampling behaviors among source nodes, varying sample size, and a transmission model with multiple transmission units in each time slot. Based on these generalizations, we develop new theoretical results (in terms of fundamental properties and performance bounds) and a new near-optimal low-complexity scheduling algorithm to minimize AoI. Our results make a major advance to existing AoI research and help bridge the gap between theory and practice.**

***Index Terms*—Age of Information (AoI), Internet of Things (IoT), sampling, transmission, scheduling.**

I. Introduction

THE availability of fresh information is of upmost importance for many IoT applications, including autonomous vehicles [2], UAV [3], industrial automation [4], and smart grid [5]. To quantify the level of *freshness,* the concept of "Age of Information" (AoI) was conceived [6], [7]. AoI metric has since captured the attention of the research community and is now under intensive investigation (see a survey on AoI in [8] and an online bibliography in [9]).

AoI is an application-layer performance metric for information latency and 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. At a particular point in the network (e.g., edge or cloud), an existing sample may be replaced by a newly received sample, resulting in an updated (smaller) AoI. AoI is an application-layer metric and is fundamentally different from delay or latency at transport/network/link layer, which only focuses on the lapsed time for moving information between two points inside the network. For the latter, once its journey is completed, the information's delay (or latency) will no longer change. In contrast, AoI measures the accrued time since the generation of a sample until the present, which includes transit delay (or latency) as only one of its components in its accounting of total elapsed time. In other words, AoI measures the *freshness* of the information from the time it was initially generated, which is more of a concern from an application layer's perspective than merely delay (or latency) for the information to transit through the network.

AoI research is still in its infancy (see Section II). This is evident from the wide use of extremely simple models and unrealistic assumptions in existing efforts. As an example, consider a typical AoI problem at IoT network edge as illustrated in Fig. 1. Such an edge IoT model, in its basic abstraction, has been widely used in some key AoI studies (see, e.g., [10]–[15]). Some common assumptions in these studies are: (i) each source node takes a sample in every time slot, (ii) each sample is of one unit data size, and (iii) at most one unit of data can be transmitted to the base station (BS) in a time slot. These simple assumptions have also been used in AoI queuing models [7], [16]–[24], multi-link AoI network models [25]–[30], multi-hop AoI network models [31], [32], and so forth (see an online bibliography in [9]).

Although results from such simple models offer some initial understanding on AoI, the potential applications of these results in practice are very limited due to the following reasons:

- *Sampling Behavior.* Most existing AoI research assumes time is slotted and all source nodes collect a sample in each time slot. Although simple, such a sampling model can hardly capture what is happening in reality. For example, a thermometer may take temperature measurement every a few seconds while a video camera may take 30 samples (frames) per second. In other words, there is a wide range of sampling behavior that a source node may follow, depending on its application.
- *Sample Size.* In addition to unrealistically sampling behavior, most existing AoI research also assumes that the sample size from each source node is identical (one unit of data). Again such a simple model is disconnected from the real world, where the sample size varies from each source node and is determined by its underlying IoT application.

Manuscript received April 1, 2019; revised September 16, 2019; accepted November 6, 2019. Date of publication November 11, 2019; date of current version December 30, 2020. This work was supported in part by ONR MURI under Grant N00014-19-1-2621. The work of Y.T. Hou was also supported in part by NSF under Grant CNS-1617634. An abridged version of this paper appears in Proc. IEEE INFOCOM, Paris, France, April 2019 [1]. Recommended for acceptance by Dr. Q. Ye. *(Corresponding author: Y. Thomas Hou.)*

The authors are with the Virginia Polytechnic Institute and State University, Blacksburg, VA 24061 USA (e-mail: licz17@vt.edu; shaoran@vt.edu; yongce@vt.edu; thou@vt.edu; wjlou@vt.edu).

Digital Object Identifier 10.1109/TNSE.2019.2952764

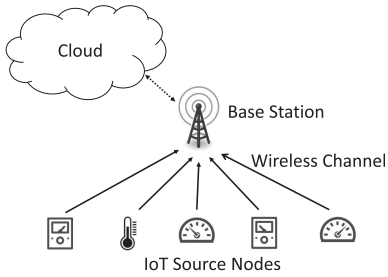


Fig. 1. System model: N IoT source nodes collect information and forward it to a BS.

- *Transmission Capacity.* Many existing efforts consider that transmission capacity is one unit of data in each time slot, which conveniently matches their simple sampling behavior and sample size. Such a simplified model, however, does not reflect the capability of state-of-the-art transmission technologies such as 4G LTE [33] and 5G NR [34]). For example, under 4G LTE or 5G cellular, transmission resource occupies both temporal and spectral domains, and there are a large number of transmission units available to the source nodes for transmission in each time slot. Such transmission capability offers a much greater scheduling space than existing AoI transmission models.

In this article, we study AoI in a general setting that is more aligned to IoT applications in the real world. Specifically, we consider the following general models. (i) We consider various sampling periods at each source node, such as arbitrary sampling, periodic sampling, and per time slot sampling. Note that per time slot sampling, the simplest sampling behavior, is what has been mostly studied in the literature. (ii) We allow sample size collected at each source node to vary, depending on the underlying application. (iii) We generalize the transmission capacity with multiple data units in each time slot to model a cellular environment (e.g., 4G LTE or 5G).

The main contributions of this paper are the following:

- We study AoI with a much more general model than those used in the state-of-the-art in terms of sampling period at the source nodes, sample size collected from each source, and transmission capacity. Such generalizations offer a much better characterization of source heterogeneity and transmission behavior in a heterogeneous IoT world. As a result, findings based on this general model not only have more significance from theoretical perspective, but also have greater impacts on IoT applications in the field.
- Under this general model, it becomes much more challenging to design an AoI minimization scheduler, due to a much larger search space than those considered in the literature. As a first step, we develop two fundamental properties for an optimal scheduling solution. We show how these properties can help reduce the search space for the optimal and near-optimal solution. These properties also serve as an aid for the development of optimal AoI scheduling algorithms.
- In the reduced search space, we further develop theoretical lower bounds for AoI under different sampling periods (arbitrary, periodic and per time slot). These lower bounds serve as performance benchmarks to assess the quality of a scheduling algorithm under different sampling behaviors.
- Finally, we design a low-complexity scheduling algorithm (Juventas). Through theoretical analysis, we find that Juventas can guarantee a factor of 3 from the objective value. Through a simulation study, we find that Juventas is near-optimal when there is no synchronization among the source nodes during sampling.

II. Related Work

In this section we review the most relevant research on AoI to this paper.

There is a body of work on modeling, analysis, and simulation of the AoI metric. In [7], [16]–[24], the authors considered different information-update queueing models (such as M/M/1, M/G/1/, D/M/1) and analyzed the AoI metric under these models. In [35], [36], the authors analyzed AoI under priority-based queuing systems. In [21], Kosta *et al.* analyzed AoI under a non-linear aging model for queuing system. In [24], Yates analyzed AoI in a network where a source updates information to a monitor through multiple servers, with each server modeled as a queue. In [37], Kam *et al.* analyzed AoI under random updates and in [38], they investigated AoI under different buffer (queue) sizes and deadlines. In [39], Kaul *et al.* analyzed AoI under both scheduled and random access (slotted-ALOHA). In [40], [41], Farazi *et al.* analyzed AoI in energy harvesting status update systems under an M/M/1 queuing model. Most of these analyses were based on overly simplified models in terms of sampling periods, sample size, transmission rate and channel conditions. Therefore, results from these analyses are only of interest from an information-theoretic perspective and are not applicable to address AoI problems under practical IoT settings.

There is an active body of work on AoI scheduling. The goal is to develop scheduling algorithms that can minimize AoI (either weighted average or otherwise) for all information sources. In [10]–[15], the authors considered an infrastructure-based model where information sources share a common channel and only one packet (from at most one source) can be transmitted to the BS in one time slot. Specifically, in [10], Hsu *et al.* considered the Bernoulli packet arrival model. In [11], [12], Kadota *et al.* considered an unreliable channel, and in [12], they considered some throughput constraints. In [15], Zhong *et al.* explored synchronization together with AoI. In [42], [43], the authors considered a multi-channel system where information sources share multiple independent wireless channels and one packet can be transmitted on each channel to the BS in a time slot. In [25]–[30], the authors considered a multi-link network environment where a subset of links can transmit simultaneously in a time slot when they are not interfering with each other. Specifically, in [28] and [29], Talak *et al.* developed a centralized and a distributed AoI scheduling algorithm, respectively. In [26], Joo

TABLE I
NOTATION

Symbol	Definition
$\bar{A}^{\mathrm{B}}$	Weighted long-term average AoI over all source nodes at the BS
$\bar{A}_i^{\mathrm{B}}$	Long-term average AoI from source node i at the BS
$\bar{A}_i^{\mathrm{s}}$	Long-term average AoI at source node i
$A_i^{\mathrm{B}}(t)$	AoI from source node i at the BS at time slot t
$A_i^{\mathrm{s}}(t)$	AoI at source node i at time slot t
$\alpha_{(\mathrm{PTS},M)}$	A lower bound of $\bar{A}$ under per time slot sampling and finite link capacity
$\alpha_{(\mathrm{ARB},\infty)}$	A lower bound of $\bar{A}$ under arbitrary time slot sampling and infinite link capacity
$\alpha_{(\mathrm{ARB},M)}$	A lower bound of $\bar{A}$ under arbitrary time slot sampling and finite link capacity
$\alpha_{(\mathrm{PRD},M)}$	A lower bound of $\bar{A}$ under periodic sampling and finite link capacity
$b_i(k)$	Beginning time slot of k-th transmission from node i
$\Delta_i(t)$	AoI decrease for potential transmission from node i in time slot t
$e_i(k)$	Ending time slot of the k-th transmission from node i
L_i	Size of information sampled at node i
M	Uplink link capacity (the number of transmission units) at each time slot
N	Number of source nodes in the network
$\boldsymbol{S}(t)$	State of the entire network at time slot t
T_i	Sampling cycle (in number of time slots) at node i
$U_i^{\mathrm{B}}(t)$	Generation time of the freshest (and complete) sample at the BS from source node i at time slot t
$U_i^{\mathrm{s}}(t)$	Generation time of the freshest sample at source node i at time slot t
w_i	Weight for node i
$x_i(t)$	Number of data units allocated to user i at time slot t
$\boldsymbol{X}(t)$	Scheduling decision for time slot t

and Eryilmaz considered both AoI and information synchronization. In [27], Lu *et al.* developed scheduling algorithms to satisfy a per-flow throughput requirement. Finally, AoI scheduling has also been studied in multi-hop networks [31], [32]. Again, most of these works on AoI scheduling algorithms only considered extremely simple sampling behaviors, sample size, and transmission technologies.

III. System Model

Consider a network consisting of N IoT source nodes and one BS as shown in Fig. 1. Each source node samples information from its environment and attempts to forward it to the BS. Such IoT data collection corresponds to uplink data transmission in cellular terminology. For multiplexing, time is divided into time slots and each time slot can accommodate a certain number of data transmission units. Denote M as the number of data transmission units per time slot over the uplink bandwidth. Then, in each time slot, the BS will allocate M data transmission units to one more more source nodes for uplink data transmission. For simplicity, with respect to each source node, we assume each transmission unit carries the same amount of information over all time slots.[1]

At each source node, information is collected (or generated) with a specific sampling period at this source. Denote L_i (in number of transmission units) as the amount of information in each sample (sample size) for source node i. Due to the heterogeneity of IoT source nodes, the sampling periods and sample sizes generally differ among the source nodes. A wide range of sampling behavior are possible among the N sources, such as:

- *Arbitrary Sampling.* Each node perform its own sampling (either following a random or deterministic pattern) and is independent of other sources. This sampling behavior is the most general one among all sampling behaviors.
- *Periodic Sampling.* Source node i performs sampling at every T_i time slots. The sampling intervals (T_i's) are generally different among different source nodes. This sampling behavior is likely the most prevailing sampling behavior among real-world IoT applications. For instance, temperature sensors usually sample information with a lower rate, while accelerometers sample with a higher rate.
- *Per Time Slot Sampling.* Each source node samples information in every time slot. This is the special case for periodic sampling with $T_i = 1$ for every node i. It is a model used by most works in AoI research (see, e.g., [12], [14], [30]). Although simple, this model may not be an accurate characterization of real world sampling behavior as each source node usually samples at different rate, due to difference in applications.

When the BS allocates transmission units to a source node, the source node will always transmit its freshest sample (the most recently generated sample). Recall each sample from each node i consists of L_i units of information. Due to the size of L_i, it may take multiple time slots to complete the transmission of this sample. Only after all L_i units of the sample from source i are transmitted to the BS, we say the BS has received this sample. Once the transmission of a sample begins, the remaining unfinished units from this sample must be transmitted (over multiple time slots if needed) before any new sample is considered (even if the new sample is fresher than the one currently under transmission).

The BS maintains the sample that it has most recently received from each source node and considers it the freshest information that it possesses from that source. Again, a sample from a source is not considered received until the sample (consisting of multiple units) is received in its entirety (possibly requiring multiple time slots). Upon receiving a sample from source i completely, the BS replaces the previous sample from source i with this newly received sample.

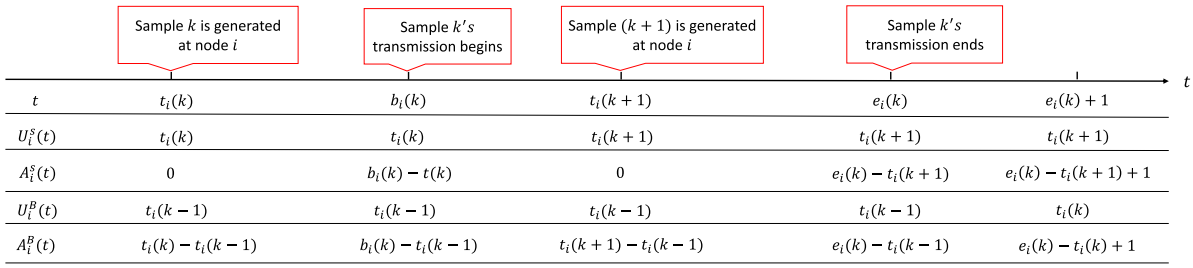
IV. AoI Modeling and Problem Statement

At each source node i, denote $U_i^{\mathrm{s}}(t)$ as the generation time of the most recent sample at time slot t. Denote $A_i^{\mathrm{s}}(t)$ as the AoI at source node i at time slot t. Recall that AoI is defined as the elapsed time for a sample between current time (now) and its generation time, we have

$$A_i^{\mathrm{s}}(t) = t - U_i^{\mathrm{s}}(t). \tag{1}$$

Note that $A_i^{\mathrm{s}}(t)$ is a zigzag-like function with a slope of 1 between sampling intervals and is reset to 0 in each time slot

[1] The more general case that considers channel diversity in time and frequency domains will be explored in a future work.



t	$t_i(k)$	$b_i(k)$	$t_i(k+1)$	$e_i(k)$	$e_i(k)+1$
$U_i^s(t)$	$t_i(k)$	$t_i(k)$	$t_i(k+1)$	$t_i(k+1)$	$t_i(k+1)$
$A_i^s(t)$	0	$b_i(k)-t(k)$	0	$e_i(k)-t_i(k+1)$	$e_i(k)-t_i(k+1)+1$
$U_i^B(t)$	$t_i(k-1)$	$t_i(k-1)$	$t_i(k-1)$	$t_i(k-1)$	$t_i(k)$
$A_i^B(t)$	$t_i(k)-t_i(k-1)$	$b_i(k)-t_i(k-1)$	$t_i(k+1)-t_i(k-1)$	$e_i(k)-t_i(k-1)$	$e_i(k)-t_i(k)+1$

Fig. 2. An example showing the evolution of $U_i^{\mathrm{S}}(t)$ and $A_i^{\mathrm{S}}(t)$ at source node i versus $U_i^{\mathrm{B}}(t)$ and $A_i^{\mathrm{B}}(t)$ at the BS during different time instances.

when a new sample is generated. Clearly, $A_i^{\mathrm{s}}(t)=0$ for all t under the per time slot sampling case, and $0 \le A_i^{\mathrm{s}}(t) < T_i$ under the periodic sampling case.

At the BS, it maintains the most recent (complete) sample that it has received from each of the N source nodes. Note that this sample maintained at the BS from source node i may be different from (older than) the freshest sample currently at source node i. Denote $U_i^{\mathrm{B}}(t)$ as the generation time of the sample from source node i that is currently maintained by the BS at time slot t. Denote $A_i^{\mathrm{B}}(t)$ as the AoI for this sample at the BS at time slot t. Then we have

$$A_i^{\mathrm{B}}(t) = t - U_i^{\mathrm{B}}(t). \tag{2}$$

Since $U_i^{\mathrm{B}}(t) \le U_i^{\mathrm{s}}(t)$, we have $A_i^{\mathrm{B}}(t) \ge A_i^{\mathrm{s}}(t)$, i.e., the AoI for source node i as perceived (maintained) by the BS is older (larger) than or equal to that at the source node, which is intuitive. Note that $A_i^{\mathrm{B}}(t)$ is also a zigzag-like function with a slope of 1 between time instances when a sample is received and is reset at the end of each time slot when a new sample is completely received at the BS.

We now make a connection between $A_i^{\mathrm{B}}(t)$ and $A_i^{\mathrm{s}}(t)$. From source node i, for the k-th sample that is actually selected for transmission,[2] denote its beginning (starting) transmission time slot as $b_i(k)$, and ending (finishing) transmission time slot as $e_i(k)$, where $e_i(k) \ge b_i(k)$. Since this k-th sample is selected for transmission at time $b_i(k)$, it must be the freshest sample at source node i at that time, with a generation time of $U_i^{\mathrm{s}}\big(b_i(k)\big)$. After this k-th sample is completely sent to the BS at the end of time slot $e_i(k)$, in the beginning of the next time slot $(e_i(k)+1)$, we have

$$U_i^{\mathrm{B}}(e_i(k)+1) = U_i^{\mathrm{s}}\big(b_i(k)\big).$$

From (2) and (1), we have

$$\begin{aligned} A_i^{\mathrm{B}}(e_i(k)+1) &= e_i(k)+1-U_i^{\mathrm{B}}(e_i(k)+1) \\ &= e_i(k)+1-U_i^{\mathrm{s}}(b_i(k)) \\ &= e_i(k)+1-\Big(b_i(k)-A_i^{\mathrm{s}}\big(b_i(k)\big)\Big) \\ &= A_i^{\mathrm{s}}\big(b_i(k)\big)+e_i(k)-b_i(k)+1. \end{aligned}$$

[2] Recall that not every sample generated at source node i will be transmitted to the BS.

Therefore, over all t, we have

$$A_i^{\mathrm{B}}(t+1) = \begin{cases} A_i^{\mathrm{s}}\big(b_i(k)\big)+e_i(k)-b_i(k)+1, & \text{if } t=e_i(k), \\ A_i^{\mathrm{B}}(t)+1, & \text{otherwise.} \end{cases} \tag{3}$$

Note that $A_i^{\mathrm{s}}\big(b_i(k)\big)+e_i(k)-b_i(k) \ge A_i^{\mathrm{s}}\big(e_i(k)\big)$ at the source node. Considering (3), clearly, we have

$$A_i^{\mathrm{B}}(t) \ge A_i^{\mathrm{s}}(t-1)+1, \forall t. \tag{4}$$

This implies that it takes at least one time slot to transmit a sample from a source node to the BS, and $A_i^{\mathrm{B}}(t)$ is always larger than $A_i^{\mathrm{s}}(t-1)+1$.

An example of AoI evolution is given in Fig. 2.

Based on (3), the long-term average of source node i's AoI at the BS can be written as:

$$\bar{A}_i^{\mathrm{B}} = \lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T} A_i^{\mathrm{B}}(t). \tag{5}$$

Denote w_i as the weight of source node i's information, which can be used to reflect the priority of node i. Then the AoI over all source nodes at the BS can be written as

$$\bar{A}^{\mathrm{B}} = \sum_{i=1}^{N} w_i \bar{A}_i^{\mathrm{B}}. \tag{6}$$

Since there are only M data units available for transmission in each time slot, a scheduling algorithm is needed to decide how to allocate the M data units to a subset of source nodes in each time slot. Denote $\boldsymbol{X}(t)$ as the scheduling decision for time slot t, where $\boldsymbol{X}(t)$ is an $N\times 1$ vector with its i-th element $x_i(t)$ being the number of data units that is allocated to user i at time slot t. Since each transmission data unit can be allocated to at most one source node, we have

$$\sum_{i=1}^{N} x_i(t) \le M. \tag{7}$$

Clearly, each different scheduling algorithm will yield a very different performance of $\bar{A}^{\mathrm{B}}$ in (6). Our goal is to find an optimal scheduling algorithm under which $\bar{A}^{\mathrm{B}}$ is minimized.

Based on (5), minimizing $\bar{A}_i^{\mathrm{B}}$ requires the design of a scheduling algorithm over an infinite number of time slots, which makes the search space for $\boldsymbol{X}(t)$ infinite. To address this problem, we show, in the next section, how to reduce the search space for an optimal scheduling solution.

V. Properties for An Optimal Scheduling Algorithm

Given that an optimal solution to our scheduling problem may not be unique, an efficient approach to reduce the search space is to find some properties associated with a particular optimal scheduling solution. Based on these properties, it becomes more tractable to find an optimal solution or design a near optimal solution.

A. An Order-based Scheduling

At each time slot t, it's intuitive to perform an order-based scheduling, that is, to assign an order for all source nodes and allocate data transmission units to the source node currently with the highest order before allocating to the source node with the second highest order and so forth. The order can be designed based on $A_i^{\mathrm{s}}(t)$, $A_i^{\mathrm{B}}(t)$, w_i, L_i, and transmission status (i.e, how many units of data that have been transmitted before the current time slot) for each source node.

The following lemma states that for each time slot t, there exists an optimal order-based scheduling algorithm that minimizes $\bar{A}^{\mathrm{B}}$.

Lemma 1: Under arbitrary sampling, there exists an order-based scheduling algorithm that achieves the optimal objective.

Proof: We prove this lemma by construction. Suppose $\boldsymbol{X}^*(t)$ is an optimal scheduling algorithm that minimizes $\bar{A}^{\mathrm{B}}$. For any time slot t, suppose the scheduled transmission samples are from source nodes $i_1, i_2, \ldots, i_P$ with ending time slots $e_1, e_2, \ldots, e_P$ such that $t \leq e_1 \leq e_2 \leq \cdots \leq e_P$. Denote $\mathcal{S}$ as the set of transmission units that are allocated to source nodes $i_1, i_2, \ldots, i_P$ to complete their current samples from time t (inclusive) under $\boldsymbol{X}^*(t)$. We define the order of those source nodes as $i_1 > i_2 > \cdots > i_P$. Based on this order, we can reallocate the transmission units in $\mathcal{S}$ as follows. We first allocate transmission units to finish i_1's sample in its entirety and then move to i_2's sample, and so on. By doing this, the ending time slot of each node will not increase and thus the new $\bar{A}^{\mathrm{B}}$ is either equal or smaller than the previous objective. Since the previous objective is optimal, then only equality is possible and such order-based scheduling is optimal. By performing this transmission-unit-reallocation for each time slot t, we can construct a new optimal scheduling algorithm that follows the order-based pattern. ■

Using this property, we only need to find or design an order-based scheduling algorithm. This property allows us to work in a much smaller search space. Also note that since this property is for arbitrary sampling, it applies to periodic and per time slot sampling policies as well.

B. Cyclic Transmission

Another property that we want to explore is whether an optimal scheduling algorithm exhibits a cyclic (periodic) transmission pattern, i.e., with the same scheduling decision for every, say T_c, time slots. Intuitively, if the sampling behaviors are not periodic, it makes no sense to perform periodic scheduling decision, so this property is hard to establish under arbitrary sampling policy. We will focus on periodic sampling policy, which also includes per time slot sampling policy.

More formally, we say a scheduling algorithm is *cyclic* if it repeats its scheduling decision for a fixed number of time slots. Denote $\boldsymbol{X}_c(t)$ as a cyclic scheduling algorithm and T_c as its cycle (in number of time slots). Then there exists a t_0 such that for any $t > t_0$, we have

$$\boldsymbol{X}_c(t) = \boldsymbol{X}_c(t + T_c).$$

The following lemma states the existence of such an optimal cyclic scheduler under periodic sampling policy.

Lemma 2: When each source is sampled periodically (even with different periods), there exists a cyclic scheduling algorithm that achieves optimal objective.

Proof: We prove this lemma by construction. We first define the *state* of the network in a time slot as the complete information of current AoI at the source nodes, current AoI at the BS, the number of remaining transmission units that are still needed for each source node, and the generation time of the unfinished sample (if there is) for each source node. Denote $\boldsymbol{S}(t)$ as the state at time slot t. Under periodic sampling, for two different time slots, if the states of the system are identical, then under the same scheduling decision, the states for the following two respective time slots are also identical. That is, if $\boldsymbol{S}(t_1) = \boldsymbol{S}(t_2)$, then if $\boldsymbol{X}(t_1) = \boldsymbol{X}(t_2)$, we have $\boldsymbol{S}(t_1 + 1) = \boldsymbol{S}(t_2 + 1)$.

Denote $W\big(\boldsymbol{S}(t)\big)$ as the corresponding weighted-sum AoI at the BS for state $\boldsymbol{S}(t)$. Then the objective can be written as

$$\bar{A}^{\mathrm{B}} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} W\big(\boldsymbol{S}(t)\big). \tag{8}$$

Suppose $\boldsymbol{X}^*(t)$ is an optimal algorithm with states $\boldsymbol{S}^*(t)$ and average AoI $\bar{A}^*$ (at the BS). Among all the time slots (from 1 to infinity), it's easy to see that those states with $W\big(\boldsymbol{S}^*(t)\big) \leq 2\bar{A}^*$ should be visited for infinite times (actually, $\boldsymbol{X}^*(t)$ visits these states in more than a half of the time slots). Notice that the number of states with $W\big(\boldsymbol{S}(t)\big) \leq 2\bar{A}^*$ is finite because the number of the possible values for each component in a state is finite when $W\big(\boldsymbol{S}(t)\big)$ is bounded. From the pigeonhole principle, we know there must be one state $\boldsymbol{S}_1$ that is visited for infinite times under $\boldsymbol{X}^*(t)$.

We then divide the time domain into infinite number of segments based on the appearance of this state, with this state appearing in the first time slot of each segment. Obviously, there must be a segment with average AoI at the BS smaller than or equal to $\bar{A}^*$. Since $X^*(t)$ is optimal, only equality is possible. Then we can construct an optimal cyclic scheduling algorithm by repeating the states within this segment. ■

Since Lemma 2 applies to periodic sampling policy, it also applies to per time slot sampling policy.

C. Complexity Analysis

Under arbitrary sampling, Lemma 1 helps greatly reduce the search space since we only need to assign orders to the source nodes at each time slot. If sampling is periodic, the search space can be further reduced by Lemma 2. However, even under periodic sampling, the reduced search space for an optimal scheduling algorithm is still infinite since T_c can be any number. To the best of our knowledge, there is no efficient way to find the optimal scheduling algorithm, even under a very simple and special case (per time sampling, $L_i = 1$, $M = 1$) [14]. Therefore, we have to pursue an efficient heuristic algorithm to achieve near-optimal performance.

VI. Performance Bounds

Before we design a scheduling algorithm, we first develop some lower bounds for our objective, $\bar{A}^{\mathrm{B}}$, under different cases (sampling behaviors and link capacities). These results are not only important to serve as a performance benchmark to assess the scheduling algorithm that we will develop (in Section VII), they are also of significant theoretical value on their own as they generalize a number of results (developed for special or simple cases) in the existing literature.

We will develop lower bounds of our objective function, denoted as $\alpha_{(*,*)}$ for the following four cases: (i) per time slot sampling under finite link capacity: $\alpha_{(\mathrm{PTS},M)}$, (ii) arbitrary sampling under infinite link capacity: $\alpha_{(\mathrm{ARB},\infty)}$, (iii) arbitrary sampling under finite link capacity: $\alpha_{(\mathrm{ARB},M)}$, and (iv) periodic sampling under finite link capacity: $\alpha_{(\mathrm{PRD},M)}$.

A. The Case of Per Time Slot Sampling Under Finite Link Capacity

In this case, each source node takes a sample at every time slot, i.e., $T_i = 1$ and $A_i^{\mathrm{s}}(t) = 0$ for all i. Here $\bar{A}_i^{\mathrm{B}}$ is purely limited by the link capacity, M. In the literature (see, e.g., [12], [14]), lower bounds for the same objective function have been developed for the simple case where $M = 1$ and $L_i = 1$ for each source node i. Our development here is for a general value of $M \geq 1$ and different values of L_i for each different source node, which is what happens in practice.

Since per time slot sampling is a special case of periodic sampling, by Lemma 2, there exists an optimal cyclic algorithm $\boldsymbol{X}_c^*(t)$ with a cycle T_c. Denote N_i as the number of fully transmitted samples from node i over a cycle of T_c time slots. In a cycle, the number of transmission units allocated among the source nodes cannot be more than the total number of available transmission units. We have

$$\sum_{i=1}^{N} N_i L_i \leq M \cdot T_c. \tag{9}$$

Define r_i as the transmission rate for source node i, i.e.,

$$r_i = \frac{N_i}{T_c}. \tag{10}$$

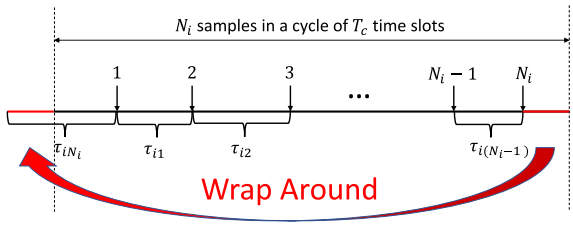


Fig. 3. A cycle with N_i samples. Each sample has its time interval since the last sample. The first interval is formed by connecting two partial intervals in the beginning and end of this cycle.

Under per time slot sampling, since at most one sample from each source node can be sent to the BS at each time slot, we have

$$0 < r_i \leq 1. \tag{11}$$

Intuitively, r_i shows the percentage of a sample from source node i that can be transmitted over a long term.

Dividing (9) by T_c and using (10), we have

$$\sum_{i=1}^{N} r_i L_i \leq M. \tag{12}$$

For source node i, there are N_i samples transmitted in a cycle. Consider the time interval between two successive transmitted samples. Then we have $(N_i - 1)$ intervals. By wrapping around a transmission cycle and viewing it cyclically (see Fig. 3), the time from the beginning of the cycle until the first transmitted sample and the time from the N_i-th transmitted sample to the end of the cycle together can be considered as the interval time for the first transmitted sample. We have, therefore, a total of N_i intervals for source node i in T_c. Denote these N_i intervals as $\tau_{i1}, \tau_{i2}, \ldots \tau_{iN_i}$ (see Fig. 3), we have

$$\sum_{j=1}^{N_i} \tau_{ij} = T_c.$$

Since it takes at least one time slot to transmit a sample from source node i to the BS, we have $A_i^{\mathrm{B}}(t) \geq 1$. Then, during time interval τ_{ij}, the sum of AoI at the BS (for source node i) is at least

$$1 + 2 + \cdots + \tau_{ij} = \frac{\tau_{ij}^2 + \tau_{ij}}{2}.$$

So in a cycle with T_c time slots, a lower bound for $\bar{A}_i^{\mathrm{B}}$ can be found by taking time average (over T_c time slots) of $\bar{A}_i^{\mathrm{B}}(t)$ for N_i time intervals. We have

$$\bar{A}_i^{\mathrm{B}} \geq \frac{1}{T_c} \sum_{j=1}^{N_i} \frac{\tau_{ij}^2 + \tau_{ij}}{2} = \frac{1}{T_c} \sum_{j=1}^{N_i} \frac{\tau_{ij}^2}{2} + \frac{1}{2}. \tag{13}$$

Applying the Cauchy-Schwarz inequality, we have

$$N_i \sum_{j=1}^{N_i} \tau_{ij}^2 \geq \left(\sum_{j=1}^{N_i} \tau_{ij} \right)^2 = T_c^2. \tag{14}$$

Applying (14) to (13) we have

$$\bar{A}_i^{\mathrm{B}} \geq \frac{T_c}{2N_i} + \frac{1}{2} \text{ or } \bar{A}_i^{\mathrm{B}} \geq \frac{1}{2r_i} + \frac{1}{2}.$$

Based on (6), we have

$$\bar{A}^{\mathrm{B}} \geq \sum_{i=1}^{N} w_i \left(\frac{1}{r_i} + \frac{1}{2} \right). \tag{15}$$

To find a lower bound for $\bar{A}^{\mathrm{B}}$, we can use a lower bound for $\sum_{i=1}^{N} w_i(\frac{1}{2r_i} + \frac{1}{2})$, which means we need to find the minimum of $\sum_{i=1}^{N} \frac{w_i}{r_i}$. We have the following optimization problem:

$$\begin{aligned} \min_{r_i} \quad & \sum_{i=1}^{N} \frac{w_i}{r_i} \\ \text{s.t.} \quad & \text{Constraints (11) and (12)} \end{aligned} \tag{16}$$

The above optimization problem is convex and can be easily solved. In the optimal solution, suppose there are K nodes ($0 \leq K \leq N$) with their $r_i^* = 1$ and the remaining $N - K$ nodes with their $r_i^* < 1$. Without loss of generality, we assume $r_i^* = 1$ for $i \leq K$ and $r_i^* < 1$ for $i > K$. Define

$$M_K = M - \sum_{i=1}^{K} L_i. \tag{17}$$

In solving the convex optimization, the KKT conditions require, for $i \leq K$,

$$\sqrt{\frac{w_i}{L_i}} \geq \frac{\sum_{j=K+1}^{N} \sqrt{w_j L_j}}{M_K}, \tag{18}$$

and for $i > K$, r_i^* is given as:

$$r_i^* = \frac{M_K \sqrt{\frac{w_i}{L_i}}}{\sum_{j=K+1}^{N} \sqrt{w_j L_j}} < 1. \tag{19}$$

With the optimal solution to (16), a lower bound of $\bar{A}^{\mathrm{B}}$, denoted by $\alpha_{(\mathrm{PTS},M)}$, is given by

$$\begin{aligned} \alpha_{(\mathrm{PTS},M)} &= \sum_{i=1}^{N} w_i \left(\frac{1}{2r_i^*} + \frac{1}{2} \right) \\ &= \frac{1}{2M_K} \left(\sum_{i=K+1}^{N} \sqrt{w_i L_i} \right)^2 + \frac{1}{2} \sum_{i=K+1}^{N} w_i + \sum_{i=1}^{K} w_i. \end{aligned} \tag{20}$$

In the special case of $M = 1$ and $L_i = 1$, we have $K = 0$ and $M_K = 0$. Therefore,

$$r_i^* = \frac{\sqrt{w_i}}{\sum_{j=1}^{N} \sqrt{w_j}} (1 \leq i \leq N), \tag{21}$$

and the lower bound

$$\alpha_{(\mathrm{PTS},1)} = \frac{1}{2} \left(\sum_{i=1}^{N} \sqrt{w_i} \right)^2 + \frac{1}{2} \sum_{i=1}^{N} w_i, \tag{22}$$

which are the main results reported in [14].

B. The Case of Arbitrary Sampling Under Infinite link Capacity

In this case the link capacity is infinite, i.e., $M \to \infty$. Here the BS can update information for all nodes in every time slot. $\bar{A}^{\mathrm{B}}$ is purely limited by the source sampling, $A_i^{\mathrm{s}}(t)$.

We define the average AoI at the source node i as

$$\bar{A}_i^{\mathrm{s}} = \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} A_i^{\mathrm{s}}(t). \tag{23}$$

From the evolution of AoI (3), under infinite link capacity, we have

$$A_i^{\mathrm{B}}(t) = A_i^{\mathrm{s}}(t-1) + 1, \text{for } t > 0.$$

So in this case $\bar{A}^{\mathrm{B}}$ equals to

$$\alpha_{(\mathrm{ARB},\infty)} = \sum_{i=1}^{N} \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} w_i (A_i^{\mathrm{s}}(t) + 1) = \sum_{i=1}^{N} w_i (\bar{A}_i^{\mathrm{s}} + 1). \tag{24}$$

This means the average AoI at the BS side is the average AoI at the source side plus 1. Here "1" is the transport delay of the network, meaning that the fresh information at the source needs one time slot to be sent to the BS. Note that under infinite link capacity, $\alpha_{(\mathrm{ARB},\infty)}$ is actually a constant rather than a theoretical bound.

It appears that none of the existing works considered the limitation of source sampling. $\alpha_{(\mathrm{ARB},\infty)}$ reveals an important fact that infinitely increasing link capacity cannot decrease $\bar{A}^{\mathrm{B}}$ to 0. Instead, $\bar{A}^{\mathrm{B}}$ will converge to $\alpha_{(\mathrm{ARB},\infty)}$.

C. The Case of Arbitrary Sampling Under Finite Link Capacity

In Section VI-A and VI-B, we have already derived two lower bounds, $\alpha_{(\mathrm{PTS},M)}$ and $\alpha_{(\mathrm{ARB},\infty)}$, respectively from the limitation of link capacity and source sampling. In the general case of arbitrary sampling and finite link capacity, both $\alpha_{(\mathrm{PTS},M)}$ and $\alpha_{(\mathrm{ARB},\infty)}$ will apply and we can choose the tighter of the two as the lower bound. That is,

$$\alpha_{(\mathrm{ARB},M)} = \max(\alpha_{(\mathrm{PTS},M)}, \alpha_{(\mathrm{ARB},\infty)}). \tag{25}$$

In the previous works (see, e.g., [12], [14]), the authors did not consider the impact of source sampling when developing a

lower bound. Thus, under arbitrary sampling and finite link capacity where source sampling is the major limiting for $\bar{A}^{\mathrm{B}}$ (e.g., when M is relatively large), $\alpha_{(\mathrm{PTS},M)}$ (a generalization of the lower bounds in [12], [14]) can be much looser than $\alpha_{(\mathrm{ARB},M)}$ developed in this paper.

D. The Case of Periodic Sampling Under Finite Link Capacity

Under the periodic sampling case (under finite link capacity), we use a new relaxation technique to develop a tighter lower bound, $\alpha_{(\mathrm{PRD},M)}$.

By Lemma 2, there exists an optimal cyclic algorithm $\boldsymbol{X}_c^*(t)$ with a cycle T_c. It's easy to see T_c should be a multiple number of each node's sampling cycle, T_i. Just as in Section VI-A, denote N_i as the number of fully transmitted samples from source node i over a cycle. Eq. (9) still holds. Other than the transmission rate r_i, for the periodic sampling case, we define p_i as the transmission percentage for source node i, i.e.,

$$p_i = \frac{N_i T_i}{T_c}. \tag{26}$$

Intuitively, p_i represents the percentage of fully transmitted samples over all generated samples in a cycle of T_c time slots. Clearly, we have

$$0 < p_i \leq 1. \tag{27}$$

Dividing (9) by T_c and using (26), we have

$$\sum_{i=1}^{N} \frac{p_i L_i}{T_i} \leq M. \tag{28}$$

Under periodic sampling we can also find N_i time intervals for each source node i in one cycle, $\tau_{i1}, \tau_{i2}, \ldots \tau_{iN_i}$, as we did in Section VI-A. To obtain a lower bound of $\bar{A}^{\mathrm{B}}$, we assume that transmission of each sample can be finished in one time slot. Consider the following problem: If N_i is given, when should these N_i transmissions occur in order to minimize $\bar{A}^{\mathrm{B}}$ in a cycle? Under the optimal strategy (to achieve the smallest $\bar{A}^{\mathrm{B}}$), transmission of a sample should occur in the time slot immediately following the time instance when the sample is taken. Further, under the optimal transmission strategy, the lengths of transmission intervals should be similar. That is, the difference between any two transmission intervals is at most one T_i. Otherwise, we can use the average of their intervals for transmission and obtain a smaller $\bar{A}^{\mathrm{B}}$.

Therefore, if we define $H_i = \lfloor \frac{1}{p_i} \rfloor$ (where $\lfloor \cdot \rfloor$ is the floor function), to minimize $\bar{A}^{\mathrm{B}}$ in one cycle, each transmission interval τ_{ij} should be equal to either $H_i T_i$ or $(H_i+1)T_i$. Suppose in one cycle, there are n_1 intervals with length $H_i T_i$ and n_2 intervals with length $(H_i+1)T_i$. We have

$$\begin{cases} n_1 + n_2 = N_i \\ n_1 H_i T_i + n_2 (H_i+1) T_i = T_c. \end{cases} \tag{29}$$

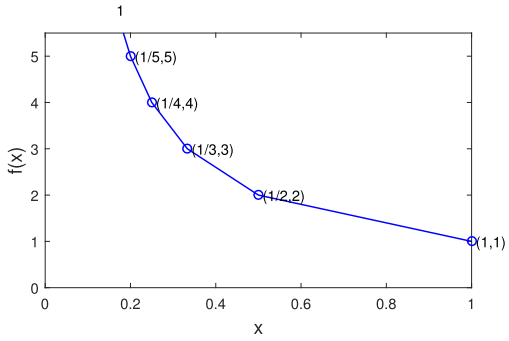


Fig. 4. The graph of function f.

Solving n_1 and n_2, we have

$$\begin{cases} n_1 = (H_i+1)N_i - \frac{T_c}{T_i} \\ n_2 = \frac{T_c}{T_i} - H_i N_i. \end{cases} \tag{30}$$

Since (13) still holds in this case, we can substitute (30) into (13) and we have

$$\begin{aligned} \bar{A}_i^B &\geq \left(\left((H_i+1)N_i - \frac{T_c}{T_i} \right) (H_i T_i)^2 \right. \\ &\quad \left. + \left(\frac{T_c}{T_i} - u_i N_i \right) \left((H_i+1)T_i \right)^2 \right) \frac{1}{2T_c} + \frac{1}{2} \\ &= \frac{T_i}{2} \left(2H_i + 1 - (H_i^2 + H_i) p_i \right) + \frac{1}{2} \\ &= \frac{T_i}{2} f(p_i) + \frac{1}{2}, \end{aligned}$$

where $f(p_i)$ is defined by

$$f(p_i) = 2 \left\lfloor \frac{1}{p_i} \right\rfloor + 1 - \left(\left\lfloor \frac{1}{p_i} \right\rfloor^2 + \left\lfloor \frac{1}{p_i} \right\rfloor \right) p_i. \tag{31}$$

The graph of function f is shown in Fig. 4. We can see function f is a piecewise linear function. In particular, it connects each pair of adjacent points $(1/n, n)$ and $(1/(n+1), n+1)$, $n \in \mathbb{N}$.

To find a lower bound for $\bar{A}^{\mathrm{B}}$, we can use a lower bound for $\sum_{i=1}^{N} w_i (\frac{T_i}{2} f(p_i) + \frac{1}{2})$, which means we need to find the minimum of $\sum_{i=1}^{N} w_i T_i f(p_i)$. We have the following optimization problem:

$$\begin{aligned} \min_{p_i} \quad & \sum_{i=1}^{N} w_i T_i f(p_i) \\ \text{s.t.} \quad & \text{Constraints (27) and (28).} \end{aligned} \tag{32}$$

Look at the graph of function f, Fig. 4. We denote $c_i = f(p_i)$, then we have

$$c_i \geq 2u_i + 1 - (u_i^2 + u_i) p_i, \forall u_i \in \mathbb{N}. \tag{33}$$

And the optimization problem (32) can be rewritten as

$$\min_{c_i,p_i} \quad \sum_{i=1}^{N} w_i T_i c_i$$
$$\text{s.t.} \quad \text{Constraints } (27), (28) \text{ and } (33). \tag{34}$$

Optimization problem (34) is a linear programming (LP) problem. However, we can not directly solve it by commercial LP solvers because there are infinite constraints lying in (33) (u_i can be any positive integer). To address this problem, we use the following procedure to solve optimization problem (34).

For each i, we reduce constraints (33) to a finite number of constraints:

$$c_i \geq 2u_i + 1 - (u_i^2 + u_i)p_i, \quad \forall u_i \in \mathbb{U}_i. \tag{35}$$

Here $\mathbb{U}_i$ is a finite subset of the set of natural numbers, $\mathbb{N}$. Compared to (33), in (35) we ignore some constraints to make the number of constraints finite. Then we construct a new optimization problem

$$\min_{c_i,p_i} \quad \sum_{i=1}^{N} w_i T_i c_i$$
$$\text{s.t.} \quad \text{Constraints } (27), (28) \text{ and } (35). \tag{36}$$

The new optimization problem (36) is a LP problem and can be easily solved by commercial solvers to get its optimal solution p_i^*. We have the following lemma.

Lemma 3: If $\lfloor 1/p_i^* \rfloor \in \mathbb{U}_i$ for each i in the optimal solution to (36), then p_i^* is also an optimal solution to (34).

Proof: Denote the optimal objective of (34) is J_1^*, and the optimal objective of (36) is J_2^*. Since the set of constraints of (36) is a subset of the set of constraints of (34), we have $J_1^* \geq J_2^*$. By observing the graph of function f (Fig. 4), we know that for any c_i and p_i, if $c_i \geq 2\lfloor \frac{1}{p_i} \rfloor + 1 - \left(\lfloor \frac{1}{p_i} \rfloor^2 + \lfloor \frac{1}{p_i} \rfloor \right) p_i$, then we have $c_i \geq 2u + 1 - (u^2 + u)p_i$ for any $u \in \mathbb{N}$. Therefore, if $\lfloor 1/p_i^* \rfloor \in \mathbb{U}_i$ for each i in the solution to (36), then p_i^* is also a feasible solution to (34) with the objective J_2^*. Recall $J_1^* \geq J_2^*$. So J_2^* is the optimal objective of (34), and p_i^* is the optimal solution to (34). ■

We then propose a computation procedure (shown in Fig. 5) to solve the optimization problem (34), which iteratively expands $\mathbb{U}_i$ until $\lfloor 1/p_i^* \rfloor \in \mathbb{U}_i$ for each i. At the end of this procedure we have $\lfloor 1/p_i^* \rfloor \in \mathbb{U}_i$ for each i. By Lemma 3, p_i^* is the optimal solution to (34).

With the optimal solution to (34), a lower bound of $\bar{A}^{\mathrm{B}}$, denoted by $\alpha_{(\mathrm{PRD},M)}$, is given by

$$\alpha_{(\mathrm{PRD},M)} = \sum_{i=1}^{N} w_i \left(\frac{T_i}{2} f(p_i^*) + \frac{1}{2} \right). \tag{37}$$

In the above derivation for $\alpha_{(\mathrm{PRD},M)}$, we consider the impacts of both link capacity and source sampling. With consideration of the fact that $f(p_i) \geq 1/p_i$ and $f(p_i) \geq 1$, we can find $\alpha_{(\mathrm{PRD},M)}$ is always tighter than both $\alpha_{(\mathrm{PTS},M)}$ and $\alpha_{(\mathrm{ARB},\infty)}$. Therefore, $\alpha_{(\mathrm{PRD},M)}$ is always tighter than the lower bound for arbitrary sampling, $\alpha_{(\mathrm{ARB},M)}$.

1: Decide the initial $\mathbb{U}_i$.
2: Solve (36), and get the optimal solution p_i^*
3: Check whether each $\lfloor 1/p_i^* \rfloor \in \mathbb{U}_i$ or not.
4: If Yes, stop and output p_i^* as the optimal solution to (34).
5: If No, expand those $\mathbb{U}_i$ to let each $\lfloor 1/p_i^* \rfloor \in \mathbb{U}_i$. Go to Step 2.

Fig. 5. Key steps to solve optimization problem (34).

VII. Juventas: A Near-Optimal Scheduler

In this section, we propose a low-complexity scheduling algorithm, code named Juventas[3], in the reduced search space derived in Section V. Under arbitrary sampling, we want to develop an order-based scheduling algorithm, which requires us to assign each node an order based on $A_i^{\mathrm{s}}(t)$, $A_i^{\mathrm{B}}(t)$, w_i, L_i in each time slot. In the following we will design this order.

For source node i, suppose transmission of a sample begins at t_1 and ends at t_2 ($t_2 \geq t_1$). Then, at time slot $(t_2 + 1)$, based on (3), we have

$$A_i^{\mathrm{B}}(t_2 + 1) = A_i^{\mathrm{s}}(t_1) + t_2 - t_1 + 1. \tag{38}$$

On the other hand, if during the same time interval $[t_1, t_2]$, source node i is not scheduled for any transmission, then based on (3), we have

$$A_i^{\mathrm{B}}(t_2 + 1) = A_i^{\mathrm{B}}(t_1) + t_2 - t_1 + 1. \tag{39}$$

Note that $A_i^{\mathrm{B}}(t_2 + 1)$ in (39) is greater than $A_i^{\mathrm{B}}(t_2 + 1)$ in (38) if the sample does not complete its transmission by time t_2. So the benefit of completing transmission of this sample by t_2 (in terms of decrease of $A_i^{\mathrm{B}}(t_2 + 1)$) is the difference on the RHS in (39) and (38), i.e.,

$$A_i^{\mathrm{B}}(t_1) - A_i^{\mathrm{s}}(t_1).$$

Note that this decrease of $A_i^{\mathrm{B}}(t)$ after t_2 is dependent on AoI difference between the BS and source node i at t_1. So the amount of AoI decrease at the BS w.r.t. source node i when a sample completes its transmission has already been determined by AoI status at an *earlier* time slot, i.e., the time slot when the sample starts its transmission.

Suppose the transmission of a sample at source node i starts at time slot t. Denote $\Delta_i(t)$ as the AoI *outage* which is given by

$$\Delta_i(t) = A_i^{\mathrm{B}}(t) - A_i^{\mathrm{s}}(t). \tag{40}$$

At each time slot t, we will use $\Delta_i(t)$ to make a scheduling decision to transmit new samples.[4] The motivation is intuitive:

[3] Juventas is the ancient Roman goddess for youth and rejuvenation.

[4] For a sample that is not finishedin the previous time slot, Juventas will use as many transmission units as needed in the current time slot to complete it (before allocating transmission units to start new samples), as shown in Fig. 6.

Juventas Algorithm

1: For each time slot t, do the following:
2: Complete transmission of the un-completed sample from the previous time slot (if there is any).
3: Among all other source nodes, find node i with the largest $\sqrt{w_i/L_i} \cdot \Delta_i(t)$.
4: If L_i is less than or equal to the number of remaining transmission units, complete transmission of this sample. Go to Step 3.
5: If L_i is greater than the number of remaining transmission units, transmit this sample incompletely with all remaining transmission units.

Fig. 6. Key steps of Juventas algorithm.

serving the node with largest $\Delta_i(t)$ will offer the greatest relieve in reducing its AoI at the BS. However, $\Delta_i(t)$ alone is not sufficient to be the scheduling metric. Both the weight w_i and packet size L_i must also be taken into considerations, as shown in (6). Therefore, we propose to use $\sqrt{w_i/L_i} \cdot \Delta_i(t)$ as the scheduling metric deciding the order for source node i. The source node with the largest value of $\sqrt{w_i/L_i}\Delta_i(t)$ will firstly be selected for transmission and the BS will allocate as many transmission units as available to transmit this sample before considering others.[5] The key steps of Juventas are shown in Fig. 6.

Besides, under periodic sampling, Juventas always makes the same scheduling scheduling decision under the same network state (recall the definition of state in Section V-B). It's obvious that for the periodic sampling case under Juventas $A_i^{\mathrm{B}}(t)$ for each source node cannot go to infinity. Therefore, following the same way in the proof of Lemma 2, we can show Juventas must visit a single state for at least 2 times. Once Juventas visits the same state twice, since its scheduling decisions at the two time slot are identical, their following states (states at the next time slots) are also identical. Then clearly, we can see Juventas follows a cyclic pattern, in which the cycle is the time interval between the two identical states.

Therefore, Juventas is an order-based algorithm under arbitrary sampling and a cyclic algorithm under periodic sampling. So we can say Juventas lies in the reduced search space that we have derived in Section V.

The following theorem offers a performance guarantee of Juventas (with a factor 3) when $L_i \leq M$.[6]

Theorem 1: Under arbitrary sampling, if $L_i \leq M$ for each source node i, $\bar{A}^{\mathrm{B}}$ under Juventas scheduling algorithm is upper bounded by

$$\bar{A}^{\mathrm{B}} \leq 3\bar{A}^* + \sum_{i=1}^{N} w_i \tag{41}$$

where $\bar{A}^*$ is the optimal objective at the BS.

Proof: Firstly we introduce an interesting property about $\Delta_i(t)$ for any scheduling algorithm (not specific to Juventas). Denote $y_i(t)$ as a binary indicator on whether or not source node i starts to transmit a sample at time slot t. Over an interval of T time slots, the sum of AoI decrease for source node i at the BS is $\sum_{t=1}^{T} y_i(t)\Delta_i(t)$. On the other hand, A_i^{B} increases by 1 at each time slot if it does not decrease (no new sample received in the time slot). When T becomes large, the sum of increase and decrease for A_i^{B} balance out. We have

$$\lim_{T\to\infty} \frac{\sum_{t=1}^{T} y_i(t)\Delta_i(t)}{\sum_{t=1}^{T} 1} = 1. \tag{42}$$

Equation (42) is equivalent to

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} y_i(t)\Delta_i(t) = 1. \tag{43}$$

We then consider Juventas. When $L_i \leq M$, under Juventas, each sample finishes its transmission within at most 2 time slots. we define $d_{il}^{\mathrm{c}}(t)$, $d_{il}^{\mathrm{p}}(t)$ and $d_{il}^{\mathrm{n}}(t)$ are the binary variables indicating whether the l-th unit from node i begins its transmission (i.e., the unit is at the first time slot of a sample's transmission), ends its previous transmission (i.e., the unit is at the second time slot of a sample's transmission), or isn't transmitted at time slot t. We have

$$d_{il}^{\mathrm{c}}(t) + d_{il}^{\mathrm{p}}(t) + d_{il}^{\mathrm{n}}(t) = 1, \quad \forall i, l, t. \tag{44}$$

From (17), at each time slot nodes $1, 2, \ldots, K$ can use at most $M - M_K$ transmission units, so there are at least M_K transmission units for nodes $K+1, K+2, \ldots, N$ to use. Therefore we have

$$\sum_{i=K+1}^{N}\sum_{l=1}^{L_i}(d_{il}^{\mathrm{c}}(t) + d_{il}^{\mathrm{p}}(t)) \geq M_K, \forall t. \tag{45}$$

Recall each sample from source node i consists of L_i units of data. Note that $y_i(t)$ is the binary indicator on whether or not source node i starts to transmit a sample at time slot t. Considering the definition of $d_{il}^{\mathrm{c}}(t)$ and $d_{il}^{\mathrm{p}}(t)$, we have

$$\lim_{T\to\infty} \frac{\sum_{t=1}^{T} L_i y_i(t)\Delta_i(t)}{\sum_{t=1}^{T}\sum_{l=1}^{L_i}(d_{il}^{\mathrm{c}}(t)\Delta_i(t) + d_{il}^{\mathrm{p}}(t)\Delta_i(t-1))} = 1. \tag{46}$$

For each node i, considering (43) and (46), we have

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T}\sum_{l=1}^{L_i}(d_{il}^{\mathrm{c}}(t)\Delta_i(t) + d_{il}^{\mathrm{p}}(t)\Delta_i(t-1)) = L_i. \tag{47}$$

[5] Our idea is corroborated by the scheduling algorithm in [14] under per time slot scheduling with L_i and $M = 1$. The authors developed a near-optimal scheduling algorithm that allocates transmission rate in proportional to $\sqrt{w_i}$. Incidentally, Juventas performs better than this scheduling algorithm even in the same simple case with $M = 1$, $L_i = 1$ and per time slot sampling. This is because we make scheduling decision in each time slot while the one in [14] makes global scheduling decision at $t = 0$.

[6] The condition $L_i \leq M$ can be easily justified in the real world where the sample taken from a source node (e.g., sensor) is almost always smaller than the cellular transmission rate in a TTI (M).

For each node $1 \leq i \leq N$, we define

$$s_i(t) = A_i^{\mathrm{B}}(t+1) - A_i^{\mathrm{s}}(t-2) - 3.$$

Suppose $d_{jl}^{\mathrm{c}}(t) = 1$. For any source node i, at time slot t, there are 2 possibilities:

- The BS has completely received a sample from source node i at the end of t. In this case, we have $s_i(t) \leq 0$.
- The BS has partially received a sample at the end of t or doesn't receive a sample from source node i. In this case, from the principle of Juventas, we have $\sqrt{w_j/L_j}\Delta_j(t) \geq \sqrt{w_i/L_i}\Delta_i(t)$. Notice that $\Delta_i(t) \geq s_i(t)$. We have $\sqrt{w_j/L_j}\Delta_j(t) \geq \sqrt{w_i/L_i}s_i(t)$

So for any j, l, i, t, we have

$$d_{jl}^{\mathrm{c}}(t)\sqrt{\frac{w_j}{L_j}}\Delta_j(t) \geq d_{jl}^{\mathrm{c}}(t)\sqrt{\frac{w_i}{L_i}}s_i(t). \tag{48}$$

Suppose $d_{jl}^{\mathrm{p}}(t) = 1$. For any source node i, at time slot $t-1$, there are 2 possibilities:

- The BS has completely received a sample from source node i at the end of $(t-1)$. In this case, we have $s_i(t) \leq 0$.
- The BS doesn't receive a sample from source node i at time slot $(t-1)$. In this case, from the principle of Juventas, we have $\sqrt{w_j/L_j}\Delta_j(t-1) \geq \sqrt{w_i/L_i}\Delta_i(t-1)$. Notice that $\Delta_i(t-1) \geq s_i(t)$. We have $\sqrt{w_j/L_j}\Delta_j(t-1) \geq \sqrt{w_i/L_i}s_i(t)$

So for any j, l, i, t, we have

$$d_{jl}^{\mathrm{p}}(t)\sqrt{\frac{w_j}{L_j}}\Delta_j(t-1) \geq d_{jl}^{\mathrm{p}}(t)\sqrt{\frac{w_i}{L_i}}s_i(t). \tag{49}$$

Combining (48) and (49) at each time slot t for each node i, we have

$$s_i(t) \leq \frac{\sum_{j=K+1}^{N}\sqrt{\frac{w_j}{L_j}}\sum_{l=1}^{L_j}(d_{jl}^{\mathrm{c}}(t)\Delta_j(t) + d_{jl}^{\mathrm{p}}(t)\Delta_j(t-1))}{\sqrt{\frac{w_i}{L_i}}\sum_{j=K+1}^{N}\sum_{l=1}^{L_j}(d_{jl}^{\mathrm{c}}(t) + d_{jl}^{\mathrm{p}}(t))}. \tag{50}$$

Combining (50) with (45) and (47), we have

$$\bar{A}_i^{\mathrm{B}} \leq \bar{A}_i^{\mathrm{s}} + 3 + \sqrt{\frac{L_i}{w_i}}\frac{\sum_{j=K+1}^{N}\sqrt{w_j L_j}}{M_K}. \tag{51}$$

For nodes $i > K$, adding all i's in (51) together, we have

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\sum_{i=K+1}^{N} w_i A_i^{\mathrm{B}}(t) \leq \frac{1}{M_R}\left(\sum_{i=K+1}^{N}\sqrt{w_i L_i}\right)^2 + \sum_{i=K+1}^{N} w_i(\bar{A}_i^{\mathrm{s}} + 3). \tag{52}$$

For each node $i \leq K$, considering (51) and (18), we have

$$\bar{A}_i^{\mathrm{B}} \leq \bar{A}_i^{\mathrm{s}} + 4 (1 \leq i \leq K). \tag{53}$$

By combining (52) and (53), we can get the performance guarantee under Juventas

$$\begin{aligned}\bar{A}^{\mathrm{B}} &\leq \frac{1}{M_K}\left(\sum_{i=K+1}^{N}\sqrt{w_i L_i}\right)^2 + \sum_{i=K+1}^{N} w_i(\bar{A}_i^{\mathrm{s}} + 3) \\ &\quad + \sum_{i=1}^{K} w_i(\bar{A}_i^{\mathrm{s}} + 4) = 2\alpha_{(\mathrm{PTS},M)} + \alpha_{(\mathrm{ARB},\infty)} + \sum_{i=1}^{N} w_i \\ &\leq 3\alpha_{(\mathrm{ARB},M)} + \sum_{i=1}^{N} w_i \leq 3\bar{A}^* + \sum_{i=1}^{N} w_i.\end{aligned}$$

■

VIII. Simulation Results

In this section, we evaluate the performance of Juventas. In our simulation, we assume the source nodes can be classified into 10 groups, with each group having the same number of source nodes. The weight, sampling rate, and sample size for the source nodes within the same group are identical but differ from those in a different group. The weight of each source node is normalized w.r.t. $\sum_{i=1}^{N} w_i$ before each simulation (i.e., after normalizing we have $\sum_{i=1}^{N} w_i = 1$). For each simulation, we start at time slot $t = 1$, and terminate when the BS has received at least 100 samples from each source node at time slot T_{term}. Then we calculate $\bar{A}^{\mathrm{B}} = (1/T_{\mathrm{term}}) \cdot \sum_{t=1}^{T_{\mathrm{term}}} \sum_{i=1}^{N} w_i A_i^{\mathrm{B}}(t)$.

We consider the three sampling behaviors: per time slot sampling, periodic sampling and random sampling respectively, and evaluate the performance of Juventas. Under periodic sampling, we further explore the impact of synchronization among sources on the performance of Juventas.

A. Per Time Slot Sampling

In this section we evaluate the performance of Juventas under per time slot sampling (i.e., each source node samples information at every time slot). We also show the lower bound for per time slot sampling, $\alpha_{(\mathrm{PTS},M)}$, in the figures as a benchmark.

(i) With the parameter settings w_i, L_i, N given in Table II and $T_i = 1$ for each i, Fig. 7 shows the objective value, $\bar{A}^{\mathrm{B}}$, as a function of increasing link capacity M. We see that $\bar{A}^{\mathrm{B}}$ for Juventas decreases monotonically as M increases. Also shown in this figure is the lower bound for periodic sampling $\alpha_{(\mathrm{PTS},M)}$ derived in (20). Clearly, we see that Juventas can achieve near-optimal performance.

(ii) With the parameter settings w_i, L_i, M given in Table II and $T_i = 1$ for each i, Fig. 8 shows the objective value, $\bar{A}^{\mathrm{B}}$, as a function of increasing number of source nodes N. We see that $\bar{A}^{\mathrm{B}}$ for Juventas increases monotonically as N increases. The lower bound $\alpha_{(\mathrm{PTS},M)}$ is also shown in this figure, and we see that Juventas can achieve near-optimal performance.

TABLE II
SIMULATION PARAMETERS

Type	1	2	3	4	5	6	7	8	9	10
w_i	6	33	25	39	36	46	35	17	35	10
L_i	1	15	11	10	19	13	13	18	17	12
T_i	10	12	45	2	25	9	49	36	26	24
p_i	0.03	0.05	0.10	0.24	0.19	0.18	0.20	0.13	0.06	0.11
M	50									
N	100									

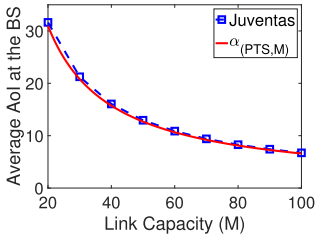


Fig. 7. Per time slot sampling: AoI for varying M.

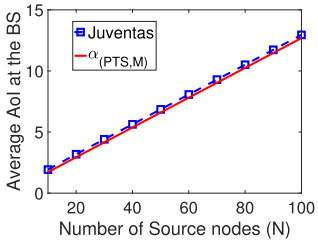


Fig. 8. Per time slot sampling: AoI for varying N.

If we compare per time slot sampling with other sampling behaviors (periodic sampling and random sampling), we can find the gap between the performance of Juventas and the lower bound is smallest under per time slot sampling (we can hardly distinguish the gaps in Fig. 7 and 8). Therefore, in terms of the gap between the performance and the lower bound, Juventas performs best under per time slot sampling.

B. Periodic Sampling

In this section we evaluate the performance of Juventas under periodic sampling (i.e., each source node i has a sampling cycle T_i, and it samples information at every T_i time slots). We also show the lower bound for periodic sampling, $\alpha_{(\mathrm{PRD},M)}$, in the figures as a benchmark.

(i) With the parameter settings w_i, L_i, T_i, N given in Table II, Fig. 9 shows the objective value, $\bar{A}^{\mathrm{B}}$, as a function of increasing link capacity M. We see that $\bar{A}^{\mathrm{B}}$ for Juventas decreases monotonically as M increase. Also shown in this figure is the lower bound for periodic sampling $\alpha_{(\mathrm{PRD},M)}$ derived in (37). we see that Juventas can achieve near-optimal performance.

(ii) With the parameter settings w_i, L_i, M, N given in Table II, Fig. 10 shows the objective value, $\bar{A}^{\mathrm{B}}$, as a function of increasing sampling cycle T. Here, all source nodes use the same sampling cycle T. The lower bound $\alpha_{(\mathrm{PRD},M)}$ is also shown in this figure, and we see that Juventas can achieve near-optimal performance. Note that $f(p_i)$ in (31) is a piecewise function so $\alpha_{(\mathrm{PRD},M)}$ doesn't increase monotonically as T increases.

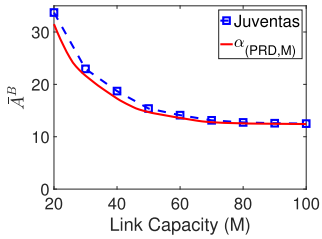


Fig. 9. Periodic sampling: AoI for varying M.

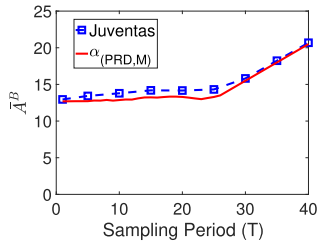


Fig. 10. Periodic sampling: AoI for varying T.

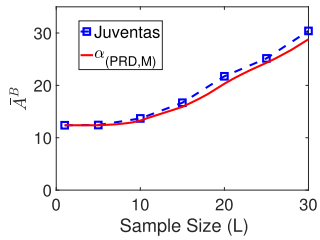


Fig. 11. Periodic sampling: AoI for varying L.

(iii) With the parameter settings w_i, T_i, M, N given in Table II, Fig. 11 shows the objective value, $\bar{A}^{\mathrm{B}}$, as a function of increasing sample size L. Here, all source nodes have the same sample size L. We see that $\bar{A}^{\mathrm{B}}$ for Juventas increases monotonically as L increases. The lower bound $\alpha_{(\mathrm{PRD},M)}$ is also shown in this figure, and we see that Juventas can achieve near-optimal performance.

(iv) With the parameter settings w_i, L_i, T_i, M given in Table II, Fig. 12 shows the objective value, $\bar{A}^{\mathrm{B}}$, as a function of increasing number of source nodes N. We see that $\bar{A}^{\mathrm{B}}$ for Juventas increases monotonically as N increases. The lower bound $\alpha_{(\mathrm{PRD},M)}$ is also shown in this figure, and we see that Juventas can achieve near-optimal performance.

C. Random Sampling

In this section we evaluate the performance of Juventas under random sampling. The sampling at each source node is modeled as an independent Bernoulli process. Specifically, each source node i samples information in probability p_i at each time slot independently. Since we haven't developed a specific lower bound for the Bernoulli sampling (random

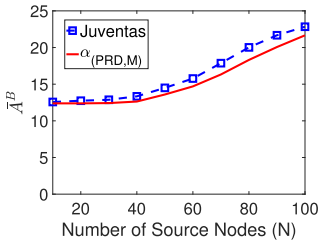


Fig. 12. Periodic sampling: AoI for varying N.

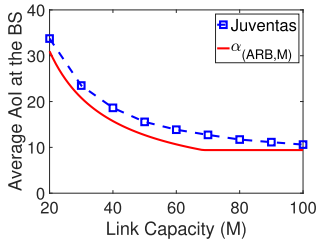


Fig. 13. Random sampling: AoI for varying M.

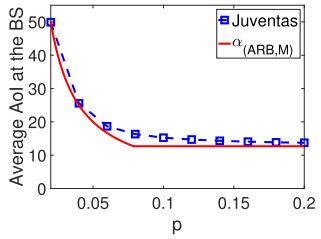


Fig. 14. Random sampling: AoI for varying p.

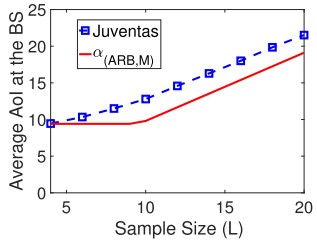


Fig. 15. Random sampling: AoI for varying L.

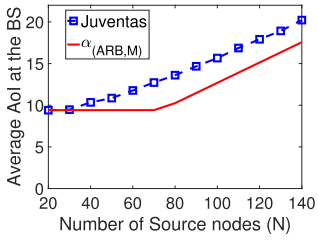


Fig. 16. Random sampling: AoI for varying N.

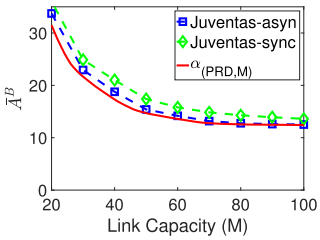


Fig. 17. Periodic sampling: weak synchronization.

sampling) case, we have to use the most general one, $\alpha_{(\mathrm{ARB},M)}$, which is valid for arbitrary sampling, as the benchmark.

(i) With the parameter settings w_i, L_i, p_i, N given in Table II, Fig. 13 shows the objective value, $\bar{A}^{\mathrm{B}}$, as a function of increasing link capacity M. We see that $\bar{A}^{\mathrm{B}}$ for Juventas decreases monotonically as M increase. Also shown in this figure is the lower bound for periodic sampling $\alpha_{(\mathrm{ARB},M)}$ derived in (25). we see that Juventas can achieve near-optimal performance.

(ii) With the parameter settings w_i, L_i, M, N given in Table II, Fig. 14 shows the objective value, $\bar{A}^{\mathrm{B}}$, as a function of increasing sampling cycle p. Here, all source nodes use the same sampling cycle p. The lower bound $\alpha_{(\mathrm{ARB},M)}$ is also shown in this figure, and we see that Juventas can achieve near-optimal performance.

(iii) With the parameter settings w_i, p_i, M, N given in Table II, Fig. 15 shows the objective value, $\bar{A}^{\mathrm{B}}$, as a function of increasing sample size L. Here, all source nodes have the same sample size L. We see that $\bar{A}^{\mathrm{B}}$ for Juventas increases monotonically as L increases. The lower bound $\alpha_{(\mathrm{ARB},M)}$ is also shown in this figure, and we see that Juventas can achieve near-optimal performance.

(iv) With the parameter settings w_i, L_i, p_i, M given in Table II, Fig. 16 shows the objective value, $\bar{A}^{\mathrm{B}}$, as a function of increasing number of source nodes N. We see that $\bar{A}^{\mathrm{B}}$ for Juventas increases monotonically as N increases. The lower bound $\alpha_{(\mathrm{ARB},M)}$ is also shown in this figure, and we see that Juventas can achieve near-optimal performance.

Compared to the cases of per time slot sampling and periodic sampling, in the case of random sampling the gap between the performance of Juventas and the lower bound is larger. The reason is that for per time slot sampling and periodic sampling, we have specifically derived tight lower bounds $\alpha_{(\mathrm{PTS},M)}$ and $\alpha_{(\mathrm{PRD},M)}$. However, we don't have specific lower bound for random sampling, and we use the lower bound for arbitrary sampling $\alpha_{(\mathrm{ARB},M)}$ as the benchmark, which is not very tight.

D. *Synchronization for Periodic Sampling*

Finally, we explore the impact of synchronization in sampling on the performance of Juventas. If two source nodes have the same sampling cycle T_i and the same initial state, $A_i^{\mathrm{s}}(0)$, we say they are synchronized. In all the simulations in Section VIII-B, the source nodes are not synchronized, either with different sampling rates or different initial states. We now study the impact of synchronization. In the first scenario, we consider synchronization only within each type of nodes (weak synchronization). In the second scenario, we consider synchronization among all source nodes (strong synchronization).

Fig. 17 shows the results under weak synchronization (with the same parameter settings as in Fig. 9). We see that the

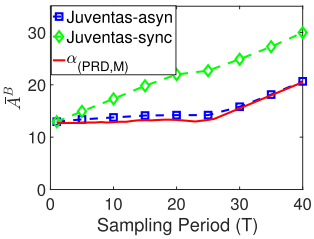


Fig. 18. Periodic sampling: strong synchronization.

objective $\bar{A}^{\mathrm{B}}$ under weak synchronization is slightly larger than that under no synchronization. Fig. 18 shows the results under strong synchronization (with the same parameter settings as in Fig. 10). We see that the objective $\bar{A}^{\mathrm{B}}$ under strong synchronization is considerably larger than that under no synchronization. Based on the results in Figs. 17 and 18, we conclude that synchronization is harmful to AoI performance and should be avoided or minimized when we initialize the source nodes.

Note that synchronization only happens under periodic sampling when there are multiple source nodes having an identical sampling cycle. Under per time slot sampling, each source node samples information at every time slot, so we don't need to consider the initial states of the source nodes. Under random sampling, since each source node samples information independently, we can safely say after a few time slots all source nodes are not synchronized in general, even if some source nodes have an identical p_i.

IX. Conclusions and Future Research

Minimizing AoI is an important objective in IoT data collection. However, most of existing research on minimizing the AoI is based on overly simplified models that are hardly useful for real world IoT applications. In this paper, we addressed this important issue by generalizing three key aspects in AoI research: sampling behavior, sample size, and transmission capacity. Under these three generalizations, we developed two fundamental properties to reduce the search space and derived tight lower bounds for an optimal solution. Further, we developed a low-complexity scheduling algorithm called Juventas, that was shown to offer near-optimal performance when there is no synchronization among the source nodes and have a guaranteed performance (within a factor of 3). The results in this paper made significant advance in bridging the gap between AoI theory and IoT data collection in practice.

Although we have successfully generalized three important aspects in AoI scheduling, there remains other aspects for further research. Most notably, in this paper, we assumed each transmission unit carries the same amount of information over all time slots. However, in real-world 4G LTE [33] or 5G NR [34] networks, due to channel diversity in time and frequency domains, different transmission units carry different amount of information. This diversity in transmission capacity adds further complexity to AoI scheduling. We will investigate its impact in our future research.

References

[1] C. Li, S. Li, and Y.T. Hou, "A general model for minimizing age of information at network edge," in *Proc. IEEE Int. Conf. Comput. Commun.*, Paris, France, Apr./May, 2019. [Online]. Available: https://www.cnsr.ictas.vt.edu/publication/general-model.pdf, Accessed: Mar. 31, 2019.

[2] L. Kong, M. K. Khan, F. Wu, G. Chen, and P. Zeng, "Millimeter-wave wireless communications for IoT-cloud supported autonomous vehicles: Overview, design, and challenges," *IEEE Commun. Mag.*, vol. 55, no. 1, pp. 62–68, Jan. 2017.

[3] N.H. Motlagh, M. Bagga, T. Taleb, "UAV-based IoT Platform: A crowd surveillance use case," *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 128–134, Feb. 2017.

[4] P. Schulz *et al.*, "Latency critical IoT applications in 5G: Perspective on the design of radio interface and network architecture," *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 70–78, Feb. 2017.

[5] V. C. Gungor *et al.*, "A survey on smart grid potential applications and communication requirements," *IEEE Trans. Ind. Inform.*, vol. 9, no. 1, pp. 28–42, Feb. 2013.

[6] S. Kaul, M. Gruteser, V. Rai, and J. Kenney, "Minimizing age of information in vehicular networks," in *Proc. 8th Annu. IEEE Commun. Soc. Conf. Sensor, Mesh Ad Hoc Commun. Netw.,* Salt Lake City, UT, USA, Jun. 27–30, 2011, pp. 350–358.

[7] S. Kaul, R. Yates, and M. Gruteser, "Real-time status: How often should one update?" in *Proc. IEEE Int. Conf. Comput. Commun.,* Orlando, FL, USA, Mar. 25–30, 2012, pp. 2731–2735.

[8] A. Kosta, N. Pappas, and V. Angelakis, "Age of information: A new concept, metric, and tool," *Found. Trends Netw.*, vol. 12, no. 3, pp. 162–259, Nov. 2017.

[9] Y. Sun, "A collection of recent papers on the age of information," [Online]. Available: http://www.auburn.edu/%7eyzs0078/, Accessed: Mar. 31, 2019.

[10] Y. Hsu, E. Modiano, and L. Duan, "Age of information: Design and analysis of optimal scheduling algorithms," in *Proc. IEEE Int. Symp. Inf. Theory*, Archen, Germany, Jun. 25–30, 2017, pp. 561–565.

[11] I. Kadota, E. Uysal-Biyikoglu, R. Singh, and E. Modiano, "Minimizing the age of information in broadcast wireless networks," in *Proc. Allerton Conf.*, Monticello, IL, USA, Sep. 27–30, 2016, pp. 844–851.

[12] I. Kadota, A. Sinha, and E. Modiano, "Optimizing age of information in wireless networks with throughput constraints," in *Proc. IEEE Int. Conf. Comput. Commun.* , Honolulu, HI, USA, Apr. 16–18, 2018, pp. 1844–1852.

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

[14] R. D. Yates, P. Ciblat, A. Yener, and M. Wigger, "Age-optimal constrained cache updating," in *Proc. IEEE Int. Symp. Inf. Theory*, Archen, Germany, Jun. 25–30, 2017, pp. 141–145.

[15] J. Zhong, R. D. Yates, and E. Soljanin, "Two freshness metrics for local cache refresh," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, June 17–22, 2018, pp. 1924–1928.

[16] M. Costa, M. Codreanu, and A. Ephremides, "Age of information with packet management," in *Proc. IEEE Int. Symp. Inf. Theory,* Honolulu, HI, USA, Jun./Jul., 2014, pp. 1583–1587.

[17] M. Costa, M. Codreanu, and A. Ephremides, "On the age of information in status update systems with packet management," *IEEE Trans. Inf. Theory*, vol. 62, no. 4, pp. 1897–1910, Apr. 2016.

[18] L. Huang and E. Modiano, "Optimizing age-of-information in a multi-class queueing system," in *Proc. IEEE Int. Symp. Inf. Theory*, Hong Kong, China, Jun. 14–19, 2015, pp. 1681–1685.

[19] Y. Inoue, H. Masuyama, T. Takine, and T. Tanaka, "The stationary distribution of the age of information in FCFS single-server queues," in *Proc. IEEE Int. Symp. Inf. Theory,* Archen, Germany, Jun. 25–30, 2017, pp. 571–575.

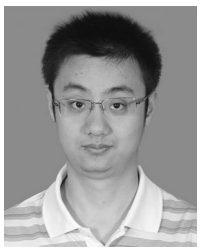
[20] C. Kam, S. Kompella, G. D. Nguyen, and A. Ephremides, "Effect of message transmission path diversity on status age," *IEEE Trans. Inf. Theory,* vol. 62, no. 3, pp. 1360–1374, Mar. 2016.

[21] A. Kosta, N. Pappas, A. Ephremides, and V. Angelakis, "Age and value of information: Non-linear age case," in *Proc. IEEE Int. Symp. Inf. Theory,* Archen, Germany, Jun. 25–30, 2017, pp. 326–330.

[22] E. Najm and E. Telatar, "Status updates in a multi-stream M/G/1/1 preemptive queue," in *Proc. IEEE INFOCOM Workshops Age Inf. Workshop*, Honolulu, HI, USA, Apr. 15–19, 2018, pp. 124–129.

[23] R. D. Yates and S. Kaul, "Real-time status updating: Multiple sources," in *Proc. IEEE Int. Symp. Inf. Theory,* Cambridge, MA, USA, Jul. 1–6, 2012, pp. 2666–2670.

[24] R. D. Yates, "Status updates through networks of parallel servers," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, Jun. 17–22, 2018, pp. 2281–2285.
[25] Q. He, D. Yuan, and A. Ephremides, "Optimal link scheduling for age minimization in wireless systems," *IEEE Trans. Inf. Theory*, vol. 64, no. 7, pp. 5381–5394, Jul. 2018.
[26] C. Joo and A. Eryilmaz, "Wireless scheduling for information freshness and synchrony: Drift-based design and heavy-traffic analysis," in *Proc. WiOpt,* Paris, France, May 15–19, 2017, pp. 1–8.
[27] N. Lu, B. Ji, and B. Li, "Age-based scheduling: improving data freshness for wireless real-time traffic," in *Proc. ACM MobiHoc,* Los Angeles, CA, USA, Jun., 2018, pp. 191–200.
[28] R. Talak, S. Karaman, and E. Modiano, "Optimizing information freshness in wireless networks under general interference constraints," in *Proc. ACM MobiHoc,* Los Angeles, CA, USA, Jun. 26–29, 2018, pp. 61–70.
[29] R. Talak, S. Karaman, and E. Modiano, "Distributed scheduling algorithms for optimizing information freshness in wireless networks," in *Proc. IEEE Int. Workshop Signal Process. Adv. Wireless Commun.,* Kalamata, Greece, Jun., 2018, pp. 1–5.
[30] R. Talak, S. Karaman, and E. Modiano, "Optimizing age of information in wireless networks with perfect channel state information," in *Proc. WiOpt,* Shanghai, China, May 7–11, 2018, pp. 1–8.
[31] A. M. Bedewy, Y. Sun, and N. B. Shroff, "Age-optimal information updates in multihop networks," in *Proc. IEEE Int. Symp. Inf. Theory,* Archen, Germany, Jun. 25–30, 2017, pp. 576–580.
[32] R. Talak, S. Karaman, and E. Modiano, "Minimizing age-of-information in multi-hop wireless networks," in *Proc. Allerton Conf.,* Monticello, IL, USA, Oct. 3–6, 2017, pp. 486–493.
[33] 3GPP TR 21.101 version 7.0.0, "Physical layer aspects for evolved UTRA," 2006. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=1247, Accessed: Dec., 2019.
[34] 3GPP TS 38.300 version 15.0.0, "NR; NR and NG-RAN overall description," 2019. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3191, Accessed: Dec. 9, 2019.
[35] S. K. Kaul and R. D. Yates, "Age of information: Updates with priority," in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, Jun. 17–22, 2018, pp. 2644–2648.
[36] J. Zhong, R. D. Yates, and E. Soljanin, "Multicast with prioritized delivery: How fresh is your data?" in *Proc. IEEE Int. Workshop Signal Process. Adv. Wireless Commun.,* Kalamata, Greece, Jun. 25–28, 2018, pp. 1–5.
[37] C. Kam, S. Kompella, and A. Ephremides, "Age of information under random updates," in *Proc. IEEE Int. Symp. Inf. Theory,* Istanbul, Turkey, Jul. 7–12, 2013, pp. 66–70.
[38] C. Kam, S. Kompella, G. D. Nguyen, J. E. Wieselthier and A. Ephremides, "Controlling the age of information: Buffer size, deadline, and packet replacement," in *Proc. IEEE Mil. Commun. Conf.,* Baltimore, MD, USA, Nov. 1–3, 2016, pp. 301–306.
[39] S. K. Kaul and R. D. Yates, "Status updates over unreliable multiaccess channels," in *Proc. IEEE Int. Symp. Inf. Theory,* Archen, Germany, Jun. 25–30, 2017, pp. 331–335.
[40] S. Farazi, A. G. Klein, and D. R. Brown III, "Average age of information for status update systems with an energy harvesting server," in *Proc. IEEE INFOCOM Workshops Age Inf. Workshop,* Honolulu, HI, USA, Apr. 16, 2018, pp. 112–117.
[41] S. Farazi, A.G. Klein, and D. R. Brown III, "Age of information in energy harvesting status update systems: When to preempt in service?" in *Proc. IEEE Int. Symp. Inf. Theory,* Vail, CO, USA, Jun. 17–22, 2018, pp. 2436–2440.
[42] A. M. Bedewy, Y. Sun, and N. B. Shroff, "Optimizing data freshness, throughput, and delay in multi-server information-update systems," in *Proc. IEEE Int. Symp. Inf. Theory,* Barcelona, Spain, Jul. 10–15, 2016, pp. 2569–2573.
[43] Y. Sun, E. Uysal-Biyikoglu, and S. Kompella, "Age-optimal updates of multiple information flows," in *Proc. IEEE INFOCOM Workshops Age Inf. Workshop*, Honolulu, HI, USA, Apr. 15–19, 2018, pp. 136–141.



Chengzhang Li (S'17) received the B.S. degree in electronics engineering from Tsinghua University, Beijing, China, in 2017. He is currently working toward the Ph.D. degree with the Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA, USA.

His current research interests include modeling, analysis and algorithm design for wireless networks, with a focus on Age of Information (AoI) and latency research.

Shaoran Li (S'17) received the B.S. degree from Southeast University, Nanjing, China, in 2014 and the M.S. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2017. He is a currently working toward the Ph.D. degree with Virginia Tech, Blacksburg, VA, USA. His research interests include algorithm design and implementation for wireless networks.

Yongce Chen (S'16) received the B.S. and M.S. degrees in electrical engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2013 and 2016, respectively. He is currently working toward the Ph.D. degree with the Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA, USA. His current research interests include optimization and algorithm design for wireless networks.

Y. Thomas Hou (F'14) received the Ph.D. degree from the New York University Tandon School of Engineering, Brooklyn, NY, USA, in 1998. He is Bradley Distinguished Professor of Electrical and Computer Engineering with Virginia Tech, Blacksburg, VA, USA, which he joined in 2002. During 1997 to 2002, he was a Member of Research Staff with Fujitsu Laboratories of America, Sunnyvale, CA, USA. His current research interests include developing innovative solutions to complex science and engineering problems arising from wireless and mobile networks. He has more than 250 papers published in IEEE/ACM journals and conferences. His papers were recognized by five best paper awards from the IEEE and two paper awards from the ACM. He holds five U.S. patents. He authored/coauthored two graduate textbooks. He was/is on the editorial boards of a number of IEEE and ACM transactions and journals. He is the Steering Committee Chair of IEEE INFOCOM conference and was a member of the IEEE Communications Society Board of Governors.

Wenjing Lou (F'15) received the Ph.D. degree in electrical and computer engineering from the University of Florida, Gainesville, FL, USA, in 2003. From 2003 to 2011, she was a Faculty Member with the Worcester Polytechnic Institute, Worcester, Massachusetts. She has been with Virginia Tech since 2011, where she is now W.C. English Professor of Computer Science. During 2014–2017, she served as a Program Director at the U.S. National Science Foundation, where she was involved in the Networking Technology and Systems program and the Secure and Trustworthy Cyberspace program. Her current research interests include privacy protection techniques in networked information systems and cross-layer security enhancement in wireless networks, by exploiting intrinsic wireless networking and communication properties.