The Power of Waiting for More than One Response in Minimizing the Age-of-Information

Yu Sang, Bin Li, and Bo Ji

***Abstract*—The Age-of-Information (AoI) has recently been proposed as an important metric for investigating the timeliness performance in information-update systems. Prior studies on AoI optimization often consider a *Push* model, which is concerned about when and how to "push" (i.e., generate and transmit) the updated information to the user. In stark contrast, in this paper we introduce a new *Pull* model, which is more relevant for certain applications (such as the real-time stock quotes service), where a user sends requests to the servers to proactively "pull" the information of interest. Moreover, we propose to employ request replication to reduce the AoI. Interestingly, we find that under this new Pull model, replication schemes capture a novel tradeoff between different levels of information freshness and different response times across the servers, which can be exploited to minimize the expected AoI at the user's side. Specifically, assuming Poisson updating process at the servers and exponentially distributed response time, we derive a closed-form formula for computing the expected AoI and obtain the optimal number of responses to wait for to minimize the expected AoI. Finally, we conduct numerical simulations to elucidate our theoretical results. Our findings show that waiting for more than one response can significantly reduce the AoI in most scenarios.**

I. Introduction

The last decades have witnessed the prevalence of smart devices and significant advances in ubiquitous computing and the Internet of things. This trend is forecasted to continue in the years to come [1]. The development of this trend has spawned a plethora of real-time services that require timely information/status updates. One practically important example of such services is vehicular networks and intelligent transportation systems [2], [3], where accurate status information (position, speed, acceleration, tire pressure, etc.) of a vehicle needs to be shared with other nearby vehicles and road-side facilities in a timely manner in order to avoid collisions and ensure substantially improved road safety. More such examples include sensor networks for environment/health monitoring [4], [5], wireless channel feedback [6], news feeds, weather updates, online social networks, fare aggregating sites (e.g., Google Shopping), and stock quotes service.

For systems providing such real-time services, those commonly used performance metrics, such as throughput and delay, exhibit significant limitations in measuring the system performance [7]. Instead, *the timeliness of information updates becomes a major concern*. To that end, a new metric called the *Age-of-Information (AoI)* has been proposed as an important metric for studying the timeliness performance [2]. The AoI is defined as the time elapsed since the most recent update occurred (see Eq. (1) for a formal definition). Using the AoI metric introduced in [2] for vehicular networks, the work of [7] employs a simple system model to analyze and optimize the timeliness performance of an information-update system. This seminal work has recently aroused dramatic interests from the research community and has inspired a series of interesting studies on the AoI analysis and optimization (see [8] and references therein).

While all prior studies consider a *Push* model, which is concerned about when and how to "push" (i.e., generate and transmit) the updated information to the user, in this paper we introduce a new *Pull* model, under which a user sends requests to the servers to proactively "pull" the information of interest. This Pull model is more relevant for many important applications where the user's interest is in the freshness of information at the point when the user requests it rather than in continuously monitoring the freshness of information. One application of the Pull model is in the real-time stock quotes service, where a customer (i.e., user) submits a query to multiple stock quotes providers (i.e., servers) and each provider responds with the most up-to-date information it has.

To the best of our knowledge, however, none of the existing work on the timeliness optimization has considered such a Pull model. In stark contrast, we focus on the Pull model and propose to employ request replication to minimize the expected AoI at the user's side. Although a similar Pull model is considered for data synchronization in [9], [10], the problems are quite different and request replication is not exploited. Note that the concept of replication is not new and has been extensively studied for various applications (e.g., cloud computing and datacenters [11], [12], storage clouds [13], parallel computing [14], [15], and databases [16], [17]). *However, for the AoI minimization problem under the Pull model, replication schemes exhibit a unique property and capture a novel tradeoff between different levels of information freshness and different response times across the servers. This tradeoff reveals the power of waiting for more than one response and can be exploited to minimize the expected AoI at the user's side.*

Next, we explain the above key tradeoff through a comparison with cloud computing systems. It has been observed that in a cloud or a datacenter, the processing time of a same

This work was supported in part by the NSF under Grants CCF-1657162, CNS-1651947, and CNS-1717108.

Yu Sang (yu.sang@temple.edu) and Bo Ji (boji@temple.edu) are with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA, and Bin Li (binli@uri.edu) is with the Department of Electrical, Computer and Biomedical Engineering, University of Rhode Island, Kingston, Rhode Island.

job can be highly variable on different servers [12]. Due to this important fact, replicating a job on multiple servers and waiting for the first finished copy can help reduce latency [11], [12]. Apparently, in such a system it is *not* beneficial to wait for more copies of the job to finish, as all the copies would give the same outcome. In contrast, in the information-update system we consider, although the servers may possess the same type of information (weather forecast, stock prices, etc.), they could have different versions of the information with different levels of freshness due to the random updating processes. Hence, the first response may come from a server with stale information; waiting for more than one response has the potential of receiving fresher information and thus helps reduce the AoI. *Hence, it is no longer the best to stop after receiving the first response (as in the other aforementioned applications).* On the other hand, waiting for too many responses will lead to a longer total waiting time and thus, also incurs a larger AoI at the user's side. *Therefore, it is challenging to determine the optimal number of responses to wait for in order to minimize the expected AoI at the user's side.*

In what follows, we summarize the key contributions of this paper. *First*, for the first time we introduce the Pull model for studying the timeliness optimization problem and propose to employ request replication to reduce the AoI. *Second*, assuming Poisson updating process at the servers and exponentially distributed response time, we derive a closed-form formula for computing the expected AoI and obtain the optimal number of responses to wait for to minimize the expected AoI. Some extensions are also discussed. *Third*, we conduct extensive numerical simulations to elucidate our theoretical results. We also investigate the impact of the system parameters (the updating rate, the mean response time, and the total number of servers) on the achieved gain in the AoI reduction. Simulation results for other types of response time distribution are also provided. Our findings show that waiting for more than one response can significantly reduce the AoI in most scenarios.

The remainder of this paper is organized as follows. We first describe our new Pull model in Section II. Then, we analyze the expected AoI under replication schemes in Section III, obtain the optimal number of responses for minimizing the expected AoI in Section IV, and briefly discuss some extensions of our work in Section V. Section VI presents the simulation results. Finally, we conclude the paper in Section VII.

II. System Model

We consider an information-update system where a user pulls time-sensitive information from n servers. These n servers are connected to a common information source and update their data *asynchronously*. We call such a model the *Pull* model (see Fig. 1). Let $i \in \{1, 2, \ldots, n\}$ be the server index. We assume that the information updating process at server i is Poisson with rate $\lambda > 0$ and is independent and identically distributed (*i.i.d.*) across the servers. This implies that the inter-update time (i.e., the time duration between two successive updates) at each server follows an exponential

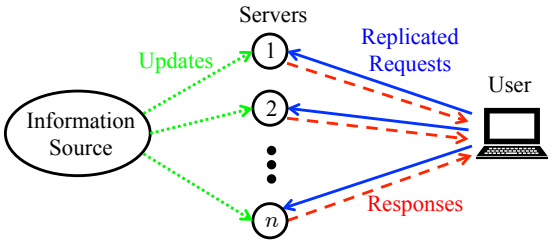


Fig. 1: The Pull model of information-update systems. Note that the arrows in the figure denote logical links rather than physical connections. The updates, requests, and responses are all transmitted through (wired or wireless) networks.

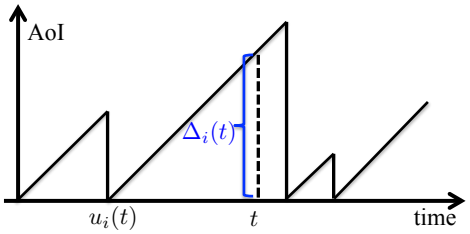


Fig. 2: An illustration of the AoI evolution at server i.

distribution with mean $1/\lambda$. Here, the inter-update time at a server can be interpreted as the time required for the server to receive information updates from the source. Let $u_i(t)$ denote the time when the most recent update at server i occurs, and let $\Delta_i(t)$ denote the AoI at server i, which is defined as the time elapsed since the most recent update at this server:

$$\Delta_i(t) \triangleq t - u_i(t). \tag{1}$$

Therefore, if an update occurs at a server, then the AoI at this server drops to zero; otherwise, the AoI increases linearly as time goes by until the next update occurs. Fig. 2 provides an illustration of the AoI evolution at server i.

In this work, we consider the (n, k) replication scheme, under which the user sends the replicated copies of the request to all n servers and waits for the first k responses. Let R_i denote the response time for server i. Note that each server may have a different response time, which is the time elapsed since the request is sent out by the user until the user receives the response from this server. We assume that the time for the requests to reach the servers is negligible compared to the time for the user to download the data from the servers. Hence, the response time can be interpreted as the downloading time. Let s denote the downloading start time, which is the same for all the servers, and let f_i denote the downloading finish time for server i. Then, the response time for server i is $R_i = f_i - s$. We assume that the response time is exponentially distributed with mean $1/\mu$ and is *i.i.d.* across the servers. Note that the model we consider above is simple, but it suffices to capture the key aspects and novelty of the problem we study.

Under the (n, k) replication scheme, when the user receives the first k responses, it uses the freshest information among

these k responses to make certain decisions (e.g., stock trading decisions based on the received stock price information). Let (j) denote the index of the server corresponding to the j-th response received by the user. Then, set $K = \{(1), (2), \ldots, (k)\}$ contains the indices of the servers that return the first k responses, and the following is satisfied: $f_{(1)} \leq f_{(2)} \leq \cdots \leq f_{(k)}$ and $R_{(1)} \leq R_{(2)} \leq \cdots \leq R_{(k)}$. Let server i^* be the one that contains the freshest information (i.e., that has the smallest AoI) among these k responses when downloading starts at time s, i.e., $i^* = \operatorname{argmin}_{i\in K} \Delta_i(s)$ (or $i^* = \operatorname{argmax}_{i\in K} u_i(s)$ due to Eq. (1)). Here, we are interested in the AoI at the user's side when it receives the k-th response, denoted by $\Delta(k)$, which is the time difference between when the k-th response is received and when the information at server i^* is updated, i.e.,

$$\Delta(k) \triangleq f_{(k)} - u_{i^*}(s). \tag{2}$$

Then, there are two natural questions of interest. *First, for a given k, can one obtain a closed-form formula for computing the expected AoI at the user's side, $\mathbb{E}[\Delta(k)]$? Second, how to determine the optimal number of responses to wait for, such that $\mathbb{E}[\Delta(k)]$ is minimized?* The second question can be formulated as the following optimization problem:

$$\min_{k\in\{1,2,\ldots,n\}} \mathbb{E}\left[\Delta(k)\right]. \tag{3}$$

We will answer these two questions in the following two sections, respectively.

III. Expected AoI

In this section, we focus on answering the first question and derive a closed-form formula for computing the expected AoI at the user's side under the (n, k) replication scheme.

We begin with the definition of $\Delta(k)$ and rewrite Eq. (2) as follows:

$$\begin{aligned}
\Delta(k) =& f_{(k)} - u_{i^*}(s) \\
=& f_{(k)} - s + s - u_{i^*}(s) \\
\overset{(a)}{=}& R_{(k)} + s - \max_{i\in K} u_i(s) \\
=& R_{(k)} + \min_{i\in K}\{s - u_i(s)\} \\
\overset{(b)}{=}& R_{(k)} + \min_{i\in K} \Delta_i(s),
\end{aligned} \tag{4}$$

where (a) is from the definition of R_i and i^* and (b) is from the definition of $\Delta_i(t)$ (i.e., Eq. (1)). As can be seen from the above expression, under the (n, k) replication scheme the AoI at the user's side consists of two terms: (i) $R_{(k)}$, the total waiting time for receiving the first k responses, and (ii) $\min_{i\in K} \Delta_i(s)$ (or $\Delta_{i^*}(s)$), the AoI of the freshest information among these k responses when downloading starts at time s. An illustration of these two terms and $\Delta(k)$ is shown in Fig. 3.

Taking the expectation of both sides of Eq. (4), we have

$$\mathbb{E}[\Delta(k)] = \mathbb{E}\left[R_{(k)}\right] + \mathbb{E}\left[\min_{i\in K} \Delta_i(s)\right]. \tag{5}$$

Intuitively, as k increases, i.e., waiting for more responses, the expected total waiting time (i.e., the first term) increases. On the other hand, upon receiving more responses, the expected AoI of the freshest information among these k responses (i.e., the second term) decreases. Hence, there is a natural tradeoff between these two terms, which is a unique property of our newly introduced Pull model.

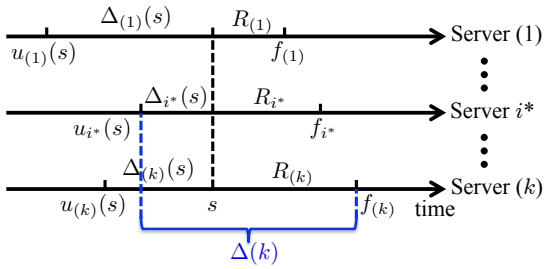


Fig. 3: An illustration of the AoI at the user's side and its two terms under the (n, k) replication scheme.

Next, we formalize this tradeoff by deriving the closed-form expressions of the above two terms as well as the expected AoI. We state the main result of this section in Theorem 1.

Theorem 1. *Under the (n, k) replication scheme, the expected AoI at the user's side can be expressed as:*

$$\mathbb{E}[\Delta(k)] = \frac{1}{\mu}(\mathbf{H}(n) - \mathbf{H}(n-k)) + \frac{1}{k\lambda}, \tag{6}$$

where $\mathbf{H}(n) = \sum_{l=1}^{n} \frac{1}{l}$ is the n-th partial sum of the diverging harmonic series.

Proof. We first analyze the the first term of the right-hand side of Eq. (5) and want to show $\mathbb{E}[R_{(k)}] = \frac{1}{\mu}(\mathbf{H}(n) - \mathbf{H}(n-k))$. Note that the response time is exponentially distributed with mean $1/\mu$ and is *i.i.d.* across the servers. Hence, random variable $R_{(k)}$ is the k-th smallest value of n *i.i.d.* exponential random variables with mean $1/\mu$. The order statistics results of exponential random variables give that $R_{(1)}$ is an exponential random variable with mean $\frac{1}{n\mu}$ and that $(R_{(j)} - R_{(j-1)})$ is an exponential random variable with mean $\frac{1}{(n+1-j)\mu}$ for any $j \in \{2, 3, \ldots, n\}$ [18]. Hence, we have the following:

$$\begin{aligned}
\mathbb{E}\left[R_{(k)}\right] &= \mathbb{E}\left[R_{(1)} + \sum_{j=2}^{k}(R_{(j)} - R_{(j-1)})\right] \\
&= \mathbb{E}[R_{(1)}] + \sum_{j=2}^{k} \mathbb{E}\left[R_{(j)} - R_{(j-1)}\right] \\
&= \sum_{j=1}^{k} \frac{1}{(n+1-j)\mu} \\
&= \frac{1}{\mu}(\mathbf{H}(n) - \mathbf{H}(n-k)).
\end{aligned} \tag{7}$$

Next, we analyze the second term of the right-hand side of Eq. (5) and want to show the following:

$$\mathbb{E}\left[\min_{i\in K} \Delta_i(s)\right] = \frac{1}{k\lambda}. \tag{8}$$

Note that the updating process at each server is a Poisson process with rate λ and is *i.i.d.* across the servers. Hence, the inter-update time for each server is exponentially distributed with mean $1/\lambda$. Due to the memoryless property of the exponential distribution, the AoI at each server has the same distribution as the inter-update time, i.e., random variable $\Delta_i(s)$ is also exponentially distributed with mean $1/\lambda$ and is *i.i.d.* across the servers [19]. Therefore, random variable $\min_{i\in K}\Delta_i(s)$ is the minimum of k *i.i.d.* exponential random variables with mean $1/\lambda$, which is also exponentially distributed with mean $\frac{1}{k\lambda}$. This implies Eq. (8).

Combining Eqs. (7) and (8), we complete the proof. □

Remark. The above analysis indeed agrees with our intuition: while the expected total waiting time for receiving the first k responses (i.e., Eq. (7)) is a monotonically increasing function of k, the expected AoI of the freshest information among these k responses (i.e., Eq. (8)) is a monotonically decreasing function of k.

IV. Optimal Replication Scheme

In this section, we will exploit the aforementioned tradeoff and focus on answering the second question we discussed at the end of Section II. Specifically, we aim to find the optimal number of responses to wait for in order to minimize the expected AoI at the user's side.

Using the analytical result of Theorem 1, we rewrite the optimization problem in Eq. (3) as:

$$\min_{k\in\{1,2,\ldots,n\}} \mathbb{E}[\Delta(k)] = \frac{1}{\mu}(\mathbf{H}(n)-\mathbf{H}(n-k)) + \frac{1}{k\lambda}. \quad (9)$$

Let k^* be an optimal solution to Eq. (9). We state the main result of this section in Theorem 2.

Theorem 2. *An optimal solution k^* can be computed as:*

$$k^* = \min\left\{\left\lceil \frac{2\mu n}{\sqrt{(\lambda+\mu)^2+4\lambda\mu n}+\lambda+\mu}\right\rceil, n\right\}. \quad (10)$$

Proof. We first define $D(k)$ as the difference of the expected AoI between the $(n,k+1)$ and (n,k) replication schemes, i.e., $D(k)\triangleq\Delta(k+1)-\Delta(k)$ for any $k\in\{1,2,\ldots,n-1\}$. From Eq. (6), we have that for any $k\in\{1,2,\ldots,n-1\}$,

$$D(k) = \frac{1}{(n-k)\mu} - \frac{1}{k(k+1)\lambda}. \quad (11)$$

It is easy to see that $D(k)$ is a monotonically increasing function of k.

We now extend the domain of $D(k)$ to the set of positive real numbers and want to find k' such that $D(k')=0$. With some standard calculations and dropping the negative solution, we derive the following:

$$k' = \frac{2\mu n}{\sqrt{(\lambda+\mu)^2+4\lambda\mu n}+\lambda+\mu}. \quad (12)$$

Next, we discuss two cases: (i) $k'>n$ and (ii) $0<k'\le n$.

In Case (i), we have $k'>n$. This implies that $D(k)=\Delta_i(k+1)-\Delta_i(k)<0$ for all $k\in\{1,2,\ldots,n\}$ due to the fact that $D(k)$ is monotonically increasing. Hence, the expected AoI $\Delta(k)$ is a monotonically decreasing function for $k\in\{1,2,\ldots,n\}$. Therefore, $k^*=n$ must be an optimal solution.

In Case (ii), we have $0<k'\le n$. We consider two subcases: k' is an integer in $\{1,2,\ldots,n\}$ and k' is not an integer.

If k' is an integer in $\{1,2,\ldots,n\}$, we have $D(k)=\Delta(k+1)-\Delta(k)<0$ for $k\in\{1,2,\ldots,k'-1\}$ and $D(k)=\Delta(k+1)-\Delta(k)>0$ for $k\in\{k'+1,\ldots,n\}$. Hence, the expected AoI $\Delta(k)$ is first decreasing (for $k\in\{1,2,\ldots,k'-1\}$) and then increasing (for $k\in\{k'+1,\ldots,n\}$). Therefore, there are two optimal solutions: $k^*=k'$ and $k^*=k'+1$ since $\Delta(k'+1)=\Delta(k')$ (due to $D(k')=0$).

If k' is not an integer, we have $D(k)=\Delta(k+1)-\Delta(k)<0$ for $k\in\{1,2,\ldots,\lfloor k'\rfloor\}$ and $D(k)=\Delta(k+1)-\Delta(k)>0$ for $k\in\{\lceil k'\rceil,\ldots,n\}$. Hence, the expected AoI $\Delta(k)$ is first decreasing (for $k\in\{1,2,\ldots,\lfloor k'\rfloor\}$) and then increasing (for $k\in\{\lceil k'\rceil,\ldots,n\}$). Therefore, $k^*=\lceil k'\rceil$ must be an optimal solution.

Combining two subcases, we have $k^*=\lceil k'\rceil$ in Case (ii). Then, combining Cases (i) and (ii), we have $k^*=\min\{\lceil k'\rceil,n\}=\min\left\{\left\lceil\frac{2\mu n}{\sqrt{(\lambda+\mu)^2+4\lambda\mu n}+\lambda+\mu}\right\rceil,n\right\}$. □

Remark. There are two special cases that are of particular interest: waiting for the first response only (i.e., $k^*=1$) and waiting for all the responses (i.e., $k^*=n$). In Corollary 1, we provide a sufficient and necessary condition for each of these two special cases.

Corollary 1. *(i) $k^*=1$ is an optimal solution if and only if $\lambda\ge\frac{\mu(n-1)}{2}$; (ii) $k^*=n$ is an optimal solution if and only if $\lambda\le\frac{\mu}{n(n-1)}$.*

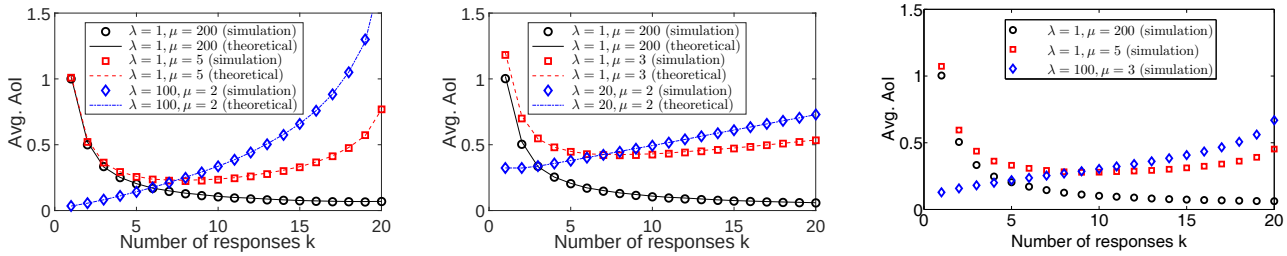
Proof. The proof follows straightforwardly from Theorem 2. A little thought gives the following: $k^*=1$ is an optimal solution if and only if $D(1)\ge 0$. Solving $D(1)=\frac{1}{(n-1)\mu}-\frac{1}{2\lambda}\ge 0$ gives $\lambda\ge\frac{\mu(n-1)}{2}$. Similarly, $k^*=n$ is an optimal solution if and only if $D(n-1)\le 0$. Solving $D(n-1)=\frac{1}{\mu}-\frac{1}{n(n-1)\lambda}\le 0$ gives $\lambda\le\frac{\mu}{n(n-1)}$. □

Remark. The above results agree well with the intuition. For a given number of servers, if the mean inter-update time is much smaller than the mean response time (i.e., $\lambda\gg\mu$), then all the servers have frequent updates and thus, the difference of the freshness levels among the servers is small. In this case, it is not beneficial to wait for more responses. On the other hand, if the mean inter-update time is much larger than the mean response time (i.e., $\lambda\ll\mu$), then one server may possess fresher information than another server. In this case, it is worth waiting for more responses, which leads to a significant gain in the AoI reduction.

V. Extensions

In this section, we discuss some extensions of our work.

Replication scheme. So far, we have only considered the (n,k) replication scheme. One limitation of this scheme is that it requires the user to send a replicated request to every



(a) Exponential response time (b) Uniform response time (c) Gamma response time

Fig. 4: Simulation results of average AoI vs. the number of responses k for three different types of response time distributions.

server, which may incur a large overhead when there are a large number of servers (i.e., when n is large). Instead, a more practical scheme would be to send the replicated requests to a subset of servers. Hence, we consider the (n, m, k) replication schemes, under which the user sends a replicated request to each of the m servers that are randomly and uniformly chosen from the n servers, and waits for the first k responses, where $m \in \{1, 2, \ldots, n\}$ and $k \in \{1, 2, \ldots, m\}$. Making the same assumptions as in Section II, we can derive the expected AoI at the user's side in a similar manner. Specifically, reusing the proof of Theorem 1 and replacing n with m in the proof, we can show the following:

$$\mathbb{E}[\Delta(k)] = \frac{1}{\mu}(\mathbf{H}(m) - \mathbf{H}(m-k)) + \frac{1}{k\lambda}. \tag{13}$$

Uniformly distributed response time. Note that our current analysis requires the memoryless property of the Poisson updating process. However, the analysis can be extended to the uniformly distributed response time. We make the same assumptions as in Section II, except that the response time is now uniformly distributed in the range of $[a, a+h]$ with $a \geq 0$ and $h \geq 0$. In this case, we have $\mathbb{E}[R_{(k)}] = \frac{kh}{n+1} + a$ [18]. Since Eq. (8) still holds, from Eq. (5) we have

$$\mathbb{E}[\Delta(k)] = \frac{kh}{n+1} + a + \frac{1}{k\lambda}. \tag{14}$$

Following a similar line of analysis to that in the proof of Theorem 2, we can show that an optimal solution k^* can be computed as:

$$k^* = \min\left\{\left\lceil \frac{2(n+1)}{\sqrt{h^2\lambda^2 + 4h\lambda(n+1)} + h\lambda} \right\rceil, n\right\}. \tag{15}$$

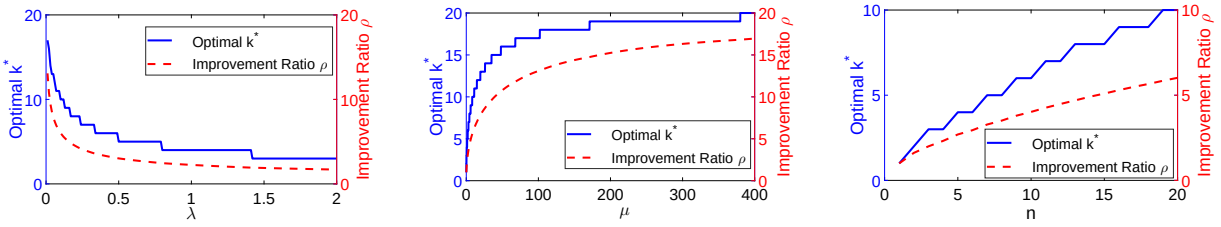
VI. Numerical Results

In this section, we perform extensive simulations to evaluate the AoI performance in an information-update system with 20 servers under the (n, k) replication scheme. We first describe our simulation settings. Throughout the simulations, the updating process at each server is assumed to be Poisson with rate λ and is *i.i.d.* across the servers. The user's request for the information is generated at time s, which is uniformly selected from the time interval $[0, T]$, where we set $T = 10^6/\lambda$. This implies that each server has a total of 10^6 updates on average.

Next, we evaluate the AoI performance through simulations for three types of response time distribution: *exponential*, *uniform*, and *Gamma*. First, we assume that the response time is exponentially distributed with mean $1/\mu$. Fig. 4a presents how the average AoI changes as the number of responses k varies in three representative setups, where each point represents an average of 10^3 simulation runs. We also include plots of our theoretical results (i.e., Eq. (6)) for comparison. A crucial observation from Fig. 4a is that the simulation results match perfectly with our theoretical results. In addition, we observe three different behaviors of the average AoI performance: (i) If the inter-update time is much smaller than the response time (i.e., $\lambda = 100$, $\mu = 2$), then the average AoI increases as k increases and thus, it is not beneficial to wait for more than one response. (ii) In contrast, if the inter-update time is much larger than the response time (i.e., $\lambda = 1$, $\mu = 200$), then the average AoI decreases as k increases and thus, it is worth waiting for all the responses so as to achieve a smaller average AoI. (iii) When the inter-update time is comparable to the response time (i.e., $\lambda = 1$, $\mu = 5$), then as k increases, the AoI would first decrease and then increase. On the one hand, when k is small, the freshness of the data at the servers dominates and thus, waiting for more responses helps reduce the average AoI. On the other hand, when k becomes large, the total waiting time becomes dominant and thus, the average AoI increases as k further increases.

In Section V, we discussed the extension of our theoretical results to the case of uniformly distributed response time. Hence, we also perform simulations for the response time uniformly distributed in the range of $[\frac{1}{2\mu}, \frac{3}{2\mu}]$ with mean $1/\mu$. Fig. 4b presents the average AoI as the number of responses k changes. In this scenario, the simulation results also match perfectly with the theoretical results (i.e., Eq. (14)). Also, we observe a very similar phenomenon to that in Fig. 4a on how the average AoI varies as k increases in three different simulation setups.

In addition, Fig. 4c presents the simulation results for the response time with Gamma distribution, which can be used to model the response time in relay networks [20]. Specifically, we consider a special class of the Gamma(r, θ) distribution that is the sum of r *i.i.d.* exponential random variables with mean θ (which is also called the Erlang distribution). Then, the mean response time $1/\mu$ is equal to $r\theta$. We fix $r = 5$ in



(a) Impact of updating rate λ. (b) Impact of mean response time $1/\mu$. (c) Impact of total number of servers n.

Fig. 5: Impact of the system parameters on the optimal k^* and the corresponding improvement ratio. We consider the exponential distribution for the response time. In (a), we fix $\mu = 1, n = 20$; in (b), we fix $\lambda = 1, n = 20$; in (c), we fix $\lambda = 1, \mu = 10$.

the simulations. Although we are unable to derive analytical results in this case, the observations are similar to that under the exponential and uniform distributions.

Finally, we investigate the impact of the system parameters (the updating rate, the mean response time, and the total number of servers) on the optimal number of responses k^* and the *improvement ratio*, defined as $\rho \triangleq \mathbb{E}[\Delta(1)]/\mathbb{E}[\Delta(k^*)]$. The improvement ratio captures the gain in the AoI reduction under the optimal scheme compared to a naive scheme of waiting for the first response only.

Fig. 5a shows the impact of the updating rate λ. We observe that the optimal number of responses k^* decreases as λ increases. This is because when the updating rate is large, the AoI diversity at the servers is small. In this case, waiting for more responses is unlikely to receive a response with much fresher information. Therefore, the optimal scheme will simply be a naive scheme that waits only for the first response when the updating rate is relatively large (e.g., $\lambda = 2$). Fig. 5b shows the impact of the mean response time $1/\mu$. We observe that the optimal number of responses k^* increases as μ increases. This is because when μ is large (i.e., when the mean response time is small), the cost of waiting for additional responses becomes marginal and thus, waiting for more responses is likely to lead to the reception of a response with fresher information. Fig. 5c shows the impact of the total number of servers n. We observe that both the optimal number of responses k^* and the improvement ratio increase with n. This is because an increased number of servers leads to more diversity gains both in the AoI at the servers and in the response time.

VII. Conclusion

In this paper, we introduced a new Pull model for studying the AoI minimization problem under the replication schemes. Assuming Poisson updating process and exponentially distributed response time, we derived the closed-form expression of the expected AoI at the user's side and provided a formula for computing the optimal solution. Not only did our work reveal a novel tradeoff between different levels of information freshness and different response times across the servers, but we also demonstrated the power of waiting for more than one response in minimizing the expected AoI at the user's side. An interesting direction for future work would be to develop dynamic replication schemes that do not require the knowledge of the updating process and the response time distribution.

References

[1] "Cisco visual networking index: Global mobile data traffic forecast update, 2015-2020," February 2016, http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.pdf.

[2] S. Kaul, M. Gruteser, V. Rai, and J. Kenney, "Minimizing age of information in vehicular networks," in *Proceedings of IEEE SECON*, 2011, pp. 350–358.

[3] S. Kaul, R. Yates, and M. Gruteser, "On piggybacking in vehicular networks," in *Proceedings of IEEE GLOBECOM*, 2011, pp. 1–5.

[4] J. Ko, C. Lu, M. B. Srivastava, J. A. Stankovic, A. Terzis, and M. Welsh, "Wireless sensor networks for healthcare," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1947–1960, 2010.

[5] P. Corke, T. Wark, R. Jurdak, W. Hu, P. Valencia, and D. Moore, "Environmental wireless sensor networks," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1903–1917, 2010.

[6] M. Costa, S. Valentin, and A. Ephremides, "On the age of channel state information for non-reciprocal wireless links," in *Proceedings of IEEE ISIT*, 2015, pp. 2356–2360.

[7] S. Kaul, R. Yates, and M. Gruteser, "Real-time status: How often should one update?" in *Proceedings of IEEE INFOCOM*, 2012, pp. 2731–2735.

[8] 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 Transactions on Information Theory*, in press, 2017.

[9] L. Bright, A. Gal, and L. Raschid, "Adaptive pull-based data freshness policies for diverse update patterns," University of Maryland, Tech. Rep., 2004. [Online]. Available: http://drum.lib.umd.edu/handle/1903/1334

[10] ——, "Adaptive pull-based policies for wide area data delivery," *ACM Transactions on Database Systems*, vol. 31, no. 2, pp. 631–671, 2006.

[11] K. Gardner, S. Zbarsky, S. Doroudi, M. Harchol-Balter, and E. Hyytia, "Reducing latency via redundant requests: Exact analysis," *ACM SIGMETRICS Performance Evaluation Review*, 2015.

[12] G. Ananthanarayanan, A. Ghodsi, S. Shenker, and I. Stoica, "Why let resources idle? Aggressive cloning of jobs with dolly," in *The 4th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)*, 2012.

[13] B. Li, A. Ramamoorthy, and R. Srikant, "Mean-field-analysis of coding versus replication in cloud storage systems," in *Proceedings of IEEE INFOCOM*, 2016, pp. 1–9.

[14] D. Wang, G. Joshi, and G. Wornell, "Efficient task replication for fast response times in parallel computation," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 42, no. 1, 2014, pp. 599–600.

[15] ——, "Using straggler replication to reduce latency in large-scale parallel computing," *ACM SIGMETRICS Performance Evaluation Review*, vol. 43, no. 3, pp. 7–11, 2015.

[16] E. Pacitti, "Improving data freshness in replicated databases," Ph.D. dissertation, INRIA, 1999.

[17] J. Pereira and M. Araújo, "Evaluating data freshness in large scale replicated databases," *INForum 2010-II Simpósio de Informática*, pp. 231–242, 2010.

[18] B. C. Arnold, N. Balakrishnan, and H. N. Nagaraja, *A first course in order statistics*. SIAM, 2008.

[19] R. Nelson, *Probability, stochastic processes, and queueing theory: the mathematics of computer performance modeling*. Springer Science & Business Media, 2013.

[20] E. Najm and R. Nasser, "Age of information: The gamma awakening," in *Proceedings of IEEE ISIT*, 2016.