# Coded Wireless Distributed Computing With Packet Losses and Retransmissions

Dong-Jun Han, *Graduate Student Member, IEEE*, Jy-Yong Sohn, *Member, IEEE*, and Jaekyun Moon, *Fellow, IEEE*

***Abstract*— In wireless distributed computing systems, mobile devices that are connected wirelessly to the Fog (e.g., small base stations) collaboratively solve a given computational task. Unfortunately, wireless distributed computing systems suffer from packet losses due to severe channel fading. Moreover, a wireless device can drop out of the system when leaving the coverage of a master node in the Fog layer. We model this unreliability between a device and a master node as a packet erasure channel. When a packet fails to be detected at the receiver, the corresponding packet is retransmitted, which would significantly increase the overall run-time to finish the task. We take a coding-theoretic approach to tackle this straggler-like problem in wireless distributed computing. We first investigate the expected latency using an $(n, k)$ maximum-distance separable (MDS) code. We obtain the lower and upper bounds on the latency in closed forms and provide guidelines to design MDS codes depending on the *channel* condition characterized by packet erasure probability. Then, we introduce another important performance metric called *minimum latency*, and also provide guidelines on designing optimal codes. Based on optimal codes, we obtain the performance curves of achievable minimum latency and achievable workload as functions of packet erasure probability.**

***Index Terms*— Coded computation, wireless distributed computing, packet loss, straggler.**

## I. Introduction

THE number of mobile and Internet of Things (IoT) devices is currently growing at an explosive pace. Billions of these devices residing at the wireless edge are valuable potential resources for solving large-scale computational tasks. Thanks to the proliferation of these computing devices at the wireless edge, there has been increased interest in wireless distributed computing systems [2]–[7]. Mobile and IoT devices that are connected wirelessly to the Fog (e.g., small base stations) can collaboratively solve large-scale computational tasks. After finishing computing, these computing devices send their computational results to the master node in the Fog layer. More recently, mobile devices at the wireless edge are utilized for training neural networks in federated learning [8]–[10].

Although wireless distributed computing is receiving significant attention nowadays, there still exist some bottlenecks to overcome. First, there are packet losses due to channel fading of wireless networks, which require retransmission of the packet until successful reception. This can cause significant delay in communication time, resulting in a severe form of straggling. Moreover, a mobile node can drop out of the system when leaving the service area of the Fog or due to blackout, which also causes unpredictable delay in finishing the task.

Regarding this straggler issue, many researchers addressed the problem in cloud computing scenarios. The authors of [11] proposed a new framework called *coded computation*, which uses maximum-distance-separable (MDS) codes to provide redundancy in distributed matrix multiplications. The $(n, k)$ MDS code has the property that any $k$ out of $n$ encoded results can recover the original $k$ data. This property of MDS code has a significant advantage in handling stragglers; even though any information regarding the straggler nodes are not known (e.g., the locations of stragglers), the master node can simply decode the result after receiving any $k$ results from the fastest $k$ workers. By applying the MDS code, it was shown that total computation time of the coded scheme can achieve an order-wise improvement over the uncoded scheme.

Since then, coded computation has been applied to various other distributed computing scenarios. Coded distributed computing schemes for high-dimensional matrix multiplication are proposed in [12], [13], and the use of short dot product for computing large linear transforms is proposed in [14]. In [15]–[20], novel coding schemes are proposed targeting distributed gradient descent. Codes for convolution and Fourier transform are studied in [21] and [22], respectively. The use of rateless codes are studied in [23], [24] for improving load balancing in distributed computing systems. In [25], [26], the authors proposed a scheme to exploit the work completed by stragglers, by sequentially allocating multiple encoded tasks to each worker.

Mitigating the effect of stragglers in more practical settings is also being studied in recent years. The authors of

Manuscript received July 10, 2020; revised March 3, 2021; accepted June 14, 2021. Date of publication June 29, 2021; date of current version December 10, 2021. This work was supported by the National Research Foundation of Korea under Grant 2019R1I1A2A02061135 and in part by the IITP funds from MSIT of Korea under Grant 2020-0-00626. This article was presented in part at the 2019 IEEE International Symposium on Information Theory (ISIT) [1]. The associate editor coordinating the review of this article and approving it for publication was L. Musavian. *(Corresponding author: Dong-Jun Han.)*

Dong-Jun Han and Jaekyun Moon are with the School of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon 34141, South Korea (e-mail: djhan93@kaist.ac.kr; jmoon@kaist.edu).

Jy-Yong Sohn is with the Department of Electrical and Computer Engineering, University of Wisconsin–Madison, Madison, WI 53706 USA (e-mail: sohn9@wisc.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TWC.2021.3091465.

Digital Object Identifier 10.1109/TWC.2021.3091465

[27], [28] proposed schemes for speeding up distributed matrix multiplication over heterogeneous networks, where a variety of computing machines coexist with different capabilities. In [29], a hierarchical coding scheme is proposed to combat stragglers in practical distributed computing systems with multiple racks. Codes are also introduced in various studies targeting data shuffling applications [3], [30]–[35]. However, the previous works listed above do not consider the packet losses caused by channel fading in practical wireless networks. This limits the previous works on coded computation to be applied in wireless distributed computing systems that are receiving significant attention nowadays.

### A. Main Contributions

In this paper, we consider a straggler problem for computing matrix-vector multiplication in wireless distributed computing setup. Compared to all existing works, our model reflects packet losses, a critical issue in wireless networks with channel fading. The transmission failure at a specific worker necessitates packet retransmission, which would in turn increase the overall run-time to finish the task. Even when a worker finishes its computation early, it can still become a straggler by communication time delay. Moreover, with limited number of retransmissions at the computing devices, the master would face an even bigger challenge to collect timely results from the workers as retransmissions for overcoming packet losses are limited. A reliable solution to deal with these problems is in great need for wireless distributed computing.

Our approach is to first separate the run-time at each worker into computation and communication time. In analyzing the communication time, we model the links between the master node and worker nodes as packet erasure channels, as justified by the packet losses frequently observed in wireless networks. We analyze the *expected latency* in this setting using an $(n, k)$ MDS code, which has significant advantage in handling stragglers due to the MDS property. We obtain the lower and upper bounds on expected latency in closed-forms. Based on the bounds, we provide guidelines to design the MDS code depending on the channel condition characterized by packet erasure probability $\epsilon$. We consider two scenarios here. The first is to find the appropriate load at each worker (optimize $k$) when the number of available workers $n$ is given, as in the previous works. The second scenario especially targets environments of the wireless edge with a massive number of IoT devices with small storage space. In this setup, the number of available workers are sufficiently large and we find the best $n$ when there is a limit on the storage space at each worker. For the special case with $k$ set to maximum (when each node computes a single inner product), we show that the expected overall run-time can be obtained by analyzing the continuous-time Markov chain with its transition rate given as a function of packet erasure probability $\epsilon$.

Then, we introduce another performance metric *minimum latency*, which can be a telling metric for wireless networks where packet losses can create a long tail in the probability of completion time. Based on the minimum latency, we provide guidelines on designing the optimal MDS codes. We also consider a setup where the number of retransmissions is limited to avoid unacceptable time delay or due to bandwidth constraint at the devices. In this setup, we obtain achievable performance curves of minimum latency as a function of packet erasure probability. Moreover, for given constraints on minimum latency, we find the maximum amount workload that the current system can handle.

### B. Related Works

We first note that the authors of [36] considered packet losses in the context of distributed storage systems. For distributed computing, the initial work on coded computation [11] optimized $k$ of an $(n, k)$ MDS code to minimize the expected latency. However, in the delay model of [11], the computation time and communication time are not separated, and thus could not reflect the packet losses that are a fact of life in practical wireless networks with channel fading.

Only a few prior works on coded computation [37], [38] did consider both the computation time and communication time. However, the works of [37], [38] assume a constant communication time, and thus do not reflect wireless environments with packet losses. The slowdown during communication is not considered at all. Compared to the settings in [37], [38], in our work, packet losses and retransmissions are taken into account. Our unique contribution is the design of codes that is tailored to the *channel* condition, i.e., packet erasure probability $\epsilon$. Analysis on the effect of limited number of retransmissions at the computing devices is also important and is unique to our work.

*Organization:* The rest of this paper is organized as follows. In Section II, we describe the master-worker setup in wireless networks with packet losses. Expected latency analysis is performed in Section III. Then in Section IV, we analyze minimum latency under the setting of limited number of retransmissions. Finally, concluding remarks are drawn in Section V.

*Notation:* We denote the set $\{1, 2, \dots n\}$ by $[n]$ for $n \in \mathbb{N}$, where $\mathbb{N}$ is the set of positive integers. We also denote the $k^{\text{th}}$ order statistics of $n$ independent random variables $(X_1, X_2, \dots X_n)$ by $X_{(k)}$. We denote $f(n) = \Theta(g(n))$ if there exist $c_1, c_2, n_0 > 0$ such that $\forall n > n_0$, $c_1 g(n) \leq f(n) \leq c_2 g(n)$.

## II. Problem Statement

### A. Wireless MDS Coded Computation

We study a matrix-vector multiplication problem, which is a key computation in many machine learning algorithms. Given a matrix $\mathbf{A} \in \mathbb{R}^{m \times d}$, the goal is to obtain $\mathbf{y} = \mathbf{A}\mathbf{x}$ for an arbitrary input vector $\mathbf{x} \in \mathbb{R}^{d \times 1}$, using $n$ distributed worker nodes and a single master node. We apply coding to combat stragglers in distributed computing; the matrix $\mathbf{A}$ is first divided into $k$ submatrices to obtain $\mathbf{A}_i \in \mathbb{R}^{\frac{m}{k} \times d}$ for $i \in [k]$. Then, an $(n, k)$ MDS code is applied to construct $\tilde{\mathbf{A}}_i \in \mathbb{R}^{\frac{m}{k} \times d}$ for $i \in [n]$, which are distributed across the $n$ worker nodes ($n > k$). Now given an arbitrary input $\mathbf{x}$, the master node broadcasts $\mathbf{x}$ to all workers, where each worker $i$ computes $\tilde{\mathbf{A}}_i \mathbf{x}$ and sends the computed result to the master. By receiving any $k$ out of $n$ results from $\{\tilde{\mathbf{A}}_i \mathbf{x}\}_{i=1}^{n}$, the master node is able to recover the desired result $\mathbf{A}\mathbf{x}$.

We especially focus on wireless networks where mobile/IoT devices serving as worker nodes are connected wirelessly to a master node in the Fog layer. The rationale for this setting comes from the recent proliferation of mobile/IoT devices at the wireless edge, which are regarded as valuable computing resources. The workers use orthogonal channels for transmissions so that the signals sent from the workers can be detected at the master without interference. Due to the channel fading in wireless networks, the communication time delay caused by packet losses should be taken into account.

### B. Computation and Communication Time Model

We assume that the required time at each worker to compute a single inner product of vectors of size $d$ follows a shifted-exponential distribution with rate $\mu_1$ and shift $\theta_1$. Since $m/k$ inner products are computed at each worker (for computing $\tilde{\mathbf{A}}_i\mathbf{x}$ at worker $i$), similar to the models in [27], [37], we assume that $X_i$, the computation time of worker $i$ follows a shifted-exponential distribution with rate $k\mu_1/m$ and shift $m\theta_1/k$:

$$\Pr[X_i \leq t] = 1 - e^{-\frac{k}{m}\mu_1(t-\frac{m}{k}\theta_1)}. \tag{1}$$

After computing the given task $\tilde{\mathbf{A}}_i\mathbf{x}$, each worker $i$ transmits its computed result to the master using packets. We assume that each packet contains $\beta$ inner products, which means that there are $\frac{m}{k\beta}$ packets to be transmitted at each worker. The master successfully obtains $\tilde{\mathbf{A}}_i\mathbf{x}$ when all packets corresponding to worker $i$ are received at the master. Channel coding is applied at each packet before each worker $i$ sends the packets for $\tilde{\mathbf{A}}_i\mathbf{x}$ to the master. Here, although the workers see independent channels, the channel-state information is assumed to be not known at the worker-side. Thus, in implementing the channel codes, the code lengths (or packet lengths) are fixed for all workers to a same specific value. Due to the channel coding, a specific packet is either successfully decoded at the master or totally dropped by decoding failure. We model such decoding failures of packets as packet erasure channels with a fixed erasure probability $\epsilon$. When a packet fails to be detected at the master (with probability $\epsilon$), the corresponding packet is retransmitted. Otherwise, the worker transmits the next packet to the master. The work is finished when the master successfully detects the results of $k$ out of $n$ workers. The system model is described in Fig. 1 with $n = 3$, $k = 2$. The communication time[1] for the $j^{\text{th}}$ transmission of the $r^{\text{th}}$ packet at each worker, $Y_{j,r}$, is modeled as a shifted-exponential distribution (as the same as the communication time model in [29]) with rate $\mu_2$ and shift parameter $\theta_2$ as follows:

$$\Pr[Y_{j,r} \leq t] = 1 - e^{-\mu_2(t-\theta_2)}. \tag{2}$$

Since each worker transmits $\frac{m}{k\beta}$ packets, the total communication time at worker $i$, denoted by $S_i$, is written as

$$S_i = \sum_{r=1}^{\frac{m}{k\beta}} \sum_{j=1}^{N_r} Y_{j,r} \tag{3}$$

[1]The communication time also includes the delay for decoding each packet at the master.

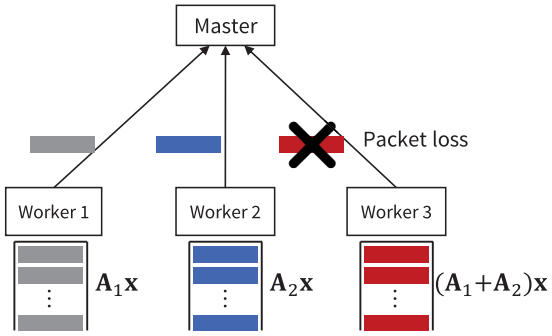



Fig. 1. The master-worker framework with $n = 3$ and $k = 2$. Each square represents a packet, which are transmitted through a packet erasure channel where the links are independent for each user. For an arbitrary input $\mathbf{x}$, the master can obtain $\mathbf{A}\mathbf{x}$ by receiving any $k = 2$ out of $n = 3$ results from $\{\tilde{\mathbf{A}}_i\mathbf{x}\}_{i=1}^3$, where $\tilde{\mathbf{A}}_1 = \mathbf{A}_1$, $\tilde{\mathbf{A}}_2 = \mathbf{A}_2$, $\tilde{\mathbf{A}}_3 = \mathbf{A}_1 + \mathbf{A}_2$.

where $N_r$ is the number of transmissions for packet $r$ which follows geometric distribution with success probability $1 - \epsilon$, i.e., $\Pr[N_r = l] = (1 - \epsilon)\epsilon^{l-1}$. Here, $\sum_{j=1}^{N_r} Y_{j,r}$ is the communication time for retransmitting a specific packet until success. The communication time at each worker is obtained by summing up this value for all different packets, which results in (3). This model fully reflects the communication time delay caused by packet losses in wireless networks.

We note that instead of discarding the failed packet at the master, one can also think of a scheme that stores the received packets and combine them with the newly retransmitted packet for decoding. This means that the retransmission probability is no longer equal to the packet erasure probability $\epsilon$. The retransmission probability now decreases as the number of retransmissions for a specific packet increases, which leads to a totally different problem setup. Considering this kind of setup is an interesting topic for future research.

### C. Problem Formulation

We define the overall run-time of the task as the sum of computation and communication time to finish the task. Now the overall run-time using an $(n, k)$ MDS code is written as

$$T = \underset{i\in[n]}{k^{\text{th}}\min}(X_i + S_i), \tag{4}$$

which represents the $k^{\text{th}}$ smallest value among $n$ values of $\{X_i + S_i\}_{i=1}^n$. For a given set of system model and channel parameter values $(\mu_1, \mu_2, \theta_1, \theta_2, \epsilon)$, we are interested in the following questions. For a given system workload $m$, what are the best values of $(n, k)$? As in many previous works, the number of workers $n$ could be a given system parameter, in which case the quest is to find the best $k$ value. However, in wireless networks where a number of low-cost mobile/IoT devices coexist, each computing device can have limit on the storage space but the number of available devices is sufficiently large. Hence, we also consider the case for finding the best $n$, for a given constraint on $m/k$ (i.e., constraint on $k$ with fixed $m$).

A natural metric is to minimize the expected latency $\mathbb{E}[T]$ as done in [27], [37]. The expected latency can also be

normalized by $(m/k)n$, the total amount of computational burden of the system. If we increase $k$, the computational burden at each node $m/k$ decreases which decreases $(m/k)n$. Moreover, $(m/k)n$ increases as the number of workers $n$ increases. For a given workload $m$, this value is minimized when $k = n$, which is the case for the uncoded scheme. This value happens to be the inverse of the code rate $R = k/n$, and one may sometimes wish to minimize the *R-normalized* expected latency $\mathbb{E}[T]/R$ instead of $\mathbb{E}[T]$.

While mathematical analysis is often possible on $\mathbb{E}[T]$, making it a convenient choice as the key performance metric, the expected latency alone may not be a sufficiently telling metric, especially for wireless networks where packet losses can create a long tail in the probability distribution of $T$. We thus introduce another metric called minimum latency beyond which the system would experience with only a small probability. Especially in scenarios of having limited number of retransmissions at the computing devices, it is inevitable to consider the minimum latency. These issues will be discussed later in Section IV.

## III. Expected Latency Analysis

We analyze the expected latency $\mathbb{E}[T]$ here. We first review some useful results on the order statistics. The expected value of the $k^{\text{th}}$ order statistic of $n$ exponential random variables of rate $\mu$ is $\frac{H_n - H_{n-k}}{\mu}$, where $H_k = \sum_{i=1}^{k} 1/i$. For $k = n$, the $k^{\text{th}}$ order statistic becomes $H_n/\mu$. As in [11], [37], $H_n$ can be approximated as $H_n \simeq \log(n) + \eta$ for large $n$ where $\eta$ is a fixed constant. For the ease of representation, in the sequel we assume that $\theta_2 = 0$. For notational simplicity, we also assume that each packet contains a single inner product, i.e., $\beta = 1$, although our analysis can be easily generalized to any packet size.

### A. Lower and Upper Bounds on $\mathbb{E}[T]$

We first state the following result on the communication time at each worker $S_i$.

*Lemma 1: The random variable $S_i$ follows an erlang distribution with shape parameter $\frac{m}{k}$ and rate parameter $(1-\epsilon)\mu_2$, i.e., $Pr[S_i \le t] = 1 - \sum_{p=0}^{\frac{m}{k}-1} \frac{1}{p!} e^{-(1-\epsilon)\mu_2 t} \{(1-\epsilon)\mu_2 t\}^p$.*

*Proof:* See Appendix A. ∎

Utilizing Lemma 1, we provide the following theorem, which shows the lower and upper bounds of $\mathbb{E}[T]$.

*Theorem 1: Let $G_i$ be an erlang random variable with shape parameter $\frac{m}{k}$ and rate parameter 1, i.e., $Pr[G_i \le t] = 1 - \sum_{p=0}^{\frac{m}{k}-1} \frac{1}{p!} e^{-t} t^p$. Then, we have $\mathcal{L} \le \mathbb{E}[T] \le \mathcal{U}$ where*

$$\mathcal{L} = \max_{i \in [k]} \left( \frac{m}{k}\theta_1 + \frac{m(H_n - H_{n-k+i-1})}{k\mu_1} + \frac{\mathbb{E}[G_{(i)}]}{(1-\epsilon)\mu_2} \right) \quad (5)$$

$$\mathcal{U} = \min_{i \in [n] \setminus [k-1]} \left( \frac{m}{k}\theta_1 + \frac{m(H_n - H_{i-k})}{k\mu_1} + \frac{\mathbb{E}[G_{(i)}]}{(1-\epsilon)\mu_2} \right). \quad (6)$$

*Proof:* See Appendix B. ∎

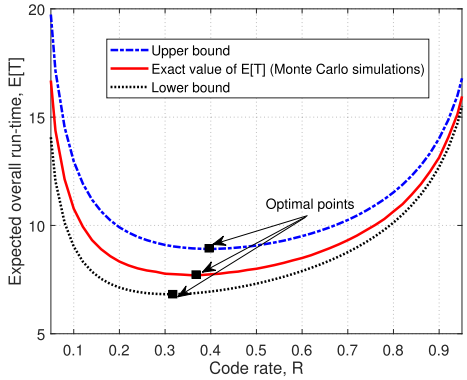



Fig. 2. Expected latency and its bounds versus code rate $R = k/n$ with $n = 100$, $m = 500$, $\mu_1 = 1$, $\mu_2 = 10$, $\epsilon = 0.1$.

As illustrated in [41], the expected value of the $i^{\text{th}}$ order statistic $\mathbb{E}[G_{(i)}]$ can be written as (7) shown at the bottom of the page, where $\Gamma$ is a gamma function[2] and $a_q(x, y)$ is the coefficient of $t^q$ in the expansion of $(\sum_{j=0}^{x-1} \frac{t^j}{j!})^y$ for any integer $x$, $y$. From (5) and (6), it can be seen that the lower and upper bounds of $\mathbb{E}[T]$ are decreasing functions of $\epsilon$. Note that $\epsilon$ only decreases the communication time (not the computation time) by the factor $\frac{1}{1-\epsilon}$ as in (5) and (6). It can be also seen that the lower and upper bounds decrease with increasing $\mu_1$, which determines the computation speed at each worker.

For the uncoded scheme, the overall workload are equally distributed to $n$ workers without replication. Therefore, the computational load at each worker are reduced by the factor $k/n$ compared to the MDS-coded scheme. We have $\mathcal{L}_{\text{uncoded}} \le \mathbb{E}[T_{\text{uncoded}}] \le \mathcal{U}_{\text{uncoded}}$ and

$$\mathcal{L}_{\text{uncoded}} = \max_{i \in [n]} \left( \frac{m}{n}\theta_1 + \frac{m(H_n - H_{i-1})}{n\mu_1} + \frac{\mathbb{E}[G'_{(i)}]}{(1-\epsilon)\mu_2} \right) \quad (8)$$

$$\mathcal{U}_{\text{uncoded}} = \frac{m}{n}\theta_1 + \frac{m(H_n - H_{n-k})}{n\mu_1} + \frac{\mathbb{E}[G'_{(i)}]}{(1-\epsilon)\mu_2} \quad (9)$$

which are obtained by inserting $k = n$ in (5) and (6), respectively. Here, $G'_i$ is an erlang random variable with shape parameter $m/n$ and rate parameter 1, where $\mathbb{E}[G'_{(i)}]$ can be obtained by inserting $k = n$ in (7).

### B. Code Rate Design

Now based on the derived bounds on expected latency, we provide guidelines on designing $(n, k)$ MDS codes. We first consider the case where the number of workers $n$ is a given parameter. Workload $m$ is also given. The goal is to find the best $k$ for given $n$, $m$. We observe Fig. 2, which shows the expected latency and the bounds as functions of code rate $R = k/n$ with $n = 100$, $m = 500$, $\mu_1 = 1$, $\mu_2 = 10$. For simplicity, we assumed $\theta_1 = 0$. It is observed that the bounds

[2]For a positive integer $z$, $\Gamma(z) = (z-1)!$.

$$\mathbb{E}[G_{(i)}] = \frac{n!}{(i-1)!(n-i)!\Gamma(\frac{m}{k})} \sum_{x=0}^{i-1} (-1)^x \binom{i-1}{x} \sum_{q=0}^{(\frac{m}{k}-1)(n-i+x)} a_q(\frac{m}{k}, n-i+x) \frac{\Gamma(\frac{m}{k}+q+1)}{(n-i+x+1)^{\frac{m}{k}+q+1}} \quad (7)$$

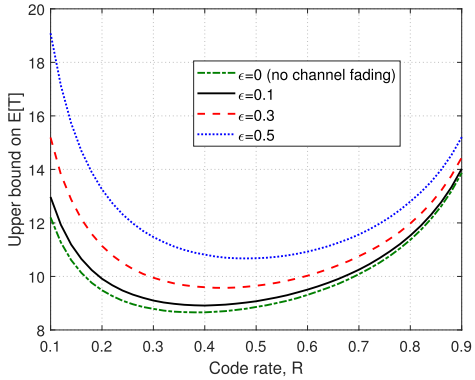



Fig. 3. Upper bound $\mathcal{U}$ on $\mathbb{E}[T]$ versus code rate $R = k/n$ with $n = 100$, $m = 500$, $\mu_1 = 1$, $\mu_2 = 10$. One can design MDS code to minimize $\mathcal{U}$ depending on the packet erasure probability $\epsilon$.

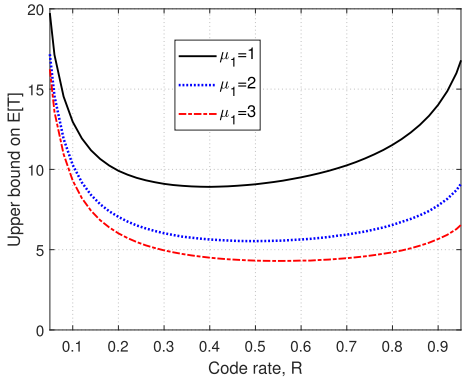



Fig. 5. Upper bound $\mathcal{U}$ on $\mathbb{E}[T]$ versus code rate $R = k/n$ with $n = 100$, $m = 500$, $\epsilon = 0.1$, $\mu_2 = 10$. One can design MDS code to minimize $\mathcal{U}$ depending on the computing speed $\mu_1$ at the workers.

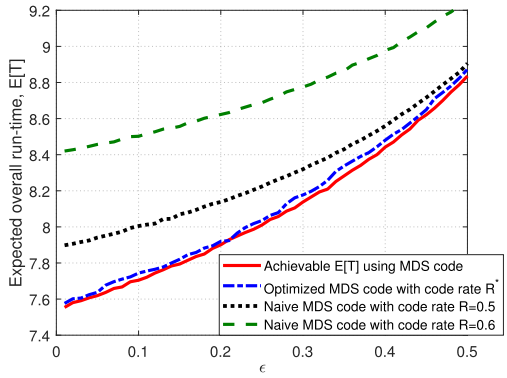



Fig. 4. Latency performance on the design targeting upper bound minimization, assuming $n = 100$, $m = 500$, $\mu_1 = 1$, $\mu_2 = 10$.

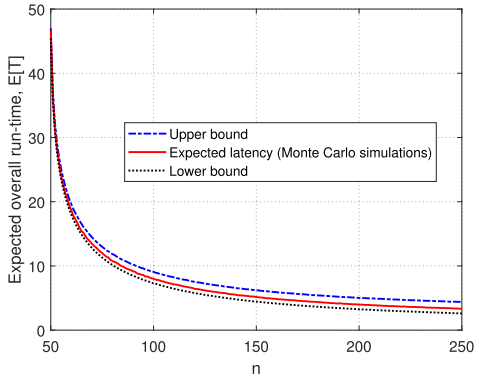



Fig. 6. Expected latency and its bounds as a function of $n$, assuming $k = 50$, $m = 500$, $\theta_1 = 0.2$, $\mu_1 = 1$, $\mu_2 = 10$, $\epsilon = 0.1$.

and the exact expected latency have very similar behaviors. With small $R$ (small $k$ for a given $n$), the size of subtask given at each worker is relatively large, resulting in an increased latency due to large computation and communication time. With large $R$, although the size of subtask at each worker is small, we observe relatively large run-time since the master should receive results from relatively large portion of worker nodes. The code rate $R$ that minimizes the exact expected latency is very close to the code rate that minimizes the upper bound $\mathcal{U}$.

In Fig. 3, we plot the upper bound $\mathcal{U}$ versus code rate $R = k/n$ with different $\epsilon$, when $n = 100$, $\mu_1 = 1$, $\mu_2 = 10$. As packet erasure probability $\epsilon$ increases, the optimal code rate minimizing $\mathcal{U}$ increases. This is because with larger $\epsilon$, reducing the number of packets to be transmitted at each worker (by increasing $k$) is more effective in reducing the latency compared to decreasing the number of workers that should successfully transmit their results to the master (by decreasing $k$). Depending on $\epsilon$, the optimal code rate $R^*$ can be designed from Fig. 3 to minimize the upper bound $\mathcal{U}$ in (5). Now the question is, how good is the performance of the MDS code which uses a code rate that minimizes $\mathcal{U}$? Fig. 4 compares the performance between the achievable $\mathbb{E}[T]$ (obtained by Monte Carlo simulations) and the MDS code with rate $R^*$. The performances of the naive MDS codes with fixed code rates are also shown. It can be seen that the performance of the MDS code with rate $R^*$ is far better than the naive schemes, and is also very close to the achievable $\mathbb{E}[T]$, showing the effectiveness of the proposed design. Note that one can also design MDS code depending on $\mu_1$, which determines the computing speed at each worker. Fig. 5 shows the details, indicating that a larger code rate is required with a larger $\mu_1$.

Now we consider another practical scenario in a wireless setup. Assume that the number of available computing devices is sufficiently large, but each device has limit on storage space. This is equivalent to considering the case where $\frac{m}{k}$ is given for a fixed workload $m$, and $n$ is a design parameter; we would like to find the best $n$ for a given $k$, $m$. Fig. 6 shows the expected latency $\mathbb{E}[T]$ and its bounds as a function of $n$, where $k = 50$, $m = 500$. Obviously, latency and its bounds decrease as $n$ increases since we are utilizing more computing resources with larger $n$ when $k$, $m$ are given. Hence, the optimal solution here is to utilize as much as workers as possible.

Now we introduce the notion of *R-normalized* expected latency $\mathbb{E}[T]/R$. Based on the derived bounds on expected latency in Theorem 1, we plot the $R$-normalized latency performance in Fig. 7. Compared to Fig. 6 which does not consider normalization of $R$, simply increasing $n$ is not an appropriate solution here. With small $n$, we have small $1/R$ but $\mathbb{E}[T]/R$ is large since we do not have enough coding

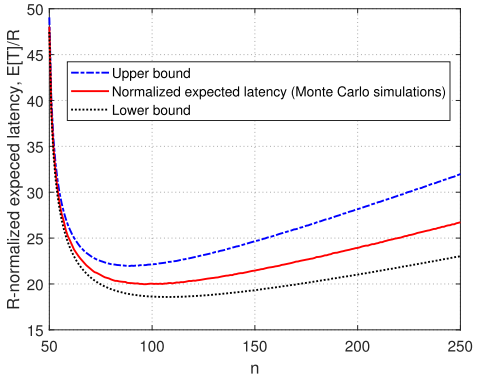



Fig. 7. $R$-normalized expected latency and its bounds versus $n$, assuming $k = 50$, $m = 500$, $\theta_1 = 0.2$, $\mu_1 = 1$, $\mu_2 = 10$, $\epsilon = 0.1$.

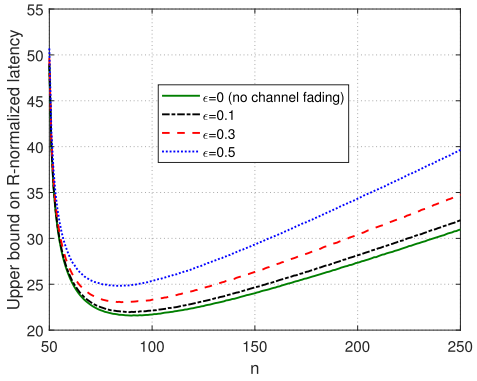



Fig. 8. Upper bound on $R$-normalized expected latency depending on $\epsilon$, where $k = 50$, $m = 500$, $\theta_1 = 0.2$, $\mu_1 = 1$, $\mu_2 = 10$.

gain to reduce the expected latency $\mathbb{E}[T]$. With very large $n$, we have small $\mathbb{E}[T]$ but again $\mathbb{E}[T]/R$ is relatively large since we have large computational burden by utilizing too many worker nodes. One can find the optimal $n$ minimizing the lower or upper bound between these two extreme cases.

In Fig. 8, we plot the upper bound on $R$-normalized expected latency as a function of $n$, depending on the channel condition $\epsilon$. An interesting observation is that optimal $n$ minimizing the upper bound decreases as $\epsilon$ increases. This is because with larger $\epsilon$, decreasing $1/R$ (by decreasing $n$) is more effective in reducing $\mathbb{E}[T]/R$ compared to decreasing $\mathbb{E}[T]$ (by increasing $n$) in this scenario with $k = 50$, $m = 500$, $\mu_1 = 1$, $\mu_2 = 10$. From Fig. 8, optimal $n$ minimizing the upper bound can be designed depending on the packet erasure probability $\epsilon$.

### C. Special Case: $k = m$

In this subsection, we especially focus on the case with $k = m$ to provide more detailed analysis with more insights. Each worker is capable of computing only a single inner product, a befitting assumption when low-cost IoT devices having very small storage size are utilized for distributed computing.

With $k = m$, the communication time $S_i$ follows an exponential distribution with rate $(1-\epsilon)\mu_2$, (i.e., $\Pr[S_i \leq t] = 1 - e^{-(1-\epsilon)\mu_2 t}$), which can be seen from the proof of Lemma 1. Now we define a continuous-time Markov chain $\mathbb{C}$ defined over 2-dimensional state space $(u, v) \in \{0, 1, \ldots, n\} \times \{0, 1, \ldots, k\}$, where the transition rates are defined as follows. The transition rate from state $(u, v)$ to state $(u+1, v)$ is $(n-u)\mu_1$ if $v \leq u < n$, and 0 otherwise. The transition rate from state $(u, v)$ to state $(u, v+1)$ is $(u-v)(1-\epsilon)\mu_2$ if $0 \leq v < \min(u, k)$, and 0 otherwise. Based on the definition of $\mathbb{C}$, we arrive at the following theorem which shows that $\mathbb{E}[T]$ can be obtained from the analysis of the hitting time of a continuous-time Markov chain. The following theorem enables us to design a more precise code minimizing the exact expected latency, not the lower or upper bounds.

*Theorem 2: Let $\tilde{T}$ be the expected hitting time of $\mathbb{C}$ from state $(0, 0)$ to the set of states $(u, v)$ with $v = k$. Then with $k = m$, the expected overall run-time becomes $\mathbb{E}[T] = \tilde{T} + \theta_1$.*

*Proof:* For a given time $t$, define two functions:

$$u(t) = |\{j \in [n] : X_j - \theta_1 \leq t\}|, \tag{10}$$

$$v(t) = |\{i \in [n] : X_i - \theta_1 + S_i \leq t\}|. \tag{11}$$

Here, $u(t)$ can be viewed as the number of workers that finished computation by time $t + \theta_1$, and $v(t)$ is the number of workers that have successfully transmitted the computation results to the master by time $t + \theta_1$. From these definitions, $v(t)$ can be rewritten as

$$v(t) = |\{i \in [u] : X_i - \theta_1 + S_i \leq t\}|. \tag{12}$$

Now we omit index $t$ in $u(t)$, $v(t)$ for notational simplicity. From $\mathbb{E}[T] = \mathbb{E}[k^{\text{th}}\min_{i\in[n]}(X_i + S_i)]$, it can be seen that $\mathbb{E}[T] - \theta_1$ is equal to the expected arrival time from $(0, 0)$ to any $(u, v)$ with $v = k$. Thus, we define the state of Markov chain $(u, v)$ over space $\{0, 1, \ldots, n\} \times \{0, 1, \ldots, k\}$.

Now the transition rates can be described as follows. By (10), for a given $t$, there are $n - u$ integers of $j \in [n]$ satisfying $X_j - \theta_1 > t$ at state $(u, v)$. Since $X_j - \theta_1$ follows an exponential distribution with rate $\mu_1$, state transition from $(u, v)$ to $(u+1, v)$ occurs with rate $(n-u)\mu_1$. The transition from $(u, v)$ from $(u+1, v)$ is possible for $u \geq v$, by the definitions (10), (11).

Next, consider the transition rate from $(u, v)$ to $(u, v+1)$ for a fixed $t$. Among the $u$ integers of $i \in [n]$ satisfying $X_i - \theta_1 \leq t$, there are $v$ of them satisfying $X_i - \theta_1 + S_i \leq t$ at state $(u, v)$. Recall that $S_i$ follows the exponential distribution with rate $(1-\epsilon)\mu_2$ from Lemma 1. Thus, state transition from $(u, v)$ to $(u, v+1)$ occurs with rate $(u-v)(1-\epsilon)\mu_2$. Note that $v \in \{0, 1, \ldots, \min(u, k) - 1\}$. Finally, we have $\mathbb{E}[T] = \tilde{T} + \theta_1$, which completes the proof. ■

For an arbitrary state $(u, v)$, the first component $u$ can be viewed as the number of workers who finished computation, and the second component $v$ is the number of workers that successfully transmitted their results to the master. Fig. 9 shows an example of the state diagram with $n = 5$, $k = 2$. From the initial state $(0, 0)$, the given task is finished when the Markov chain visits one of the states with $v = 2$ for the first time. Since the computational results of a worker are transmitted after the worker finishes the whole computation, $u \geq v$ always holds. The expected hitting time of the Markov chain can be obtained performing the first-step analysis [39].

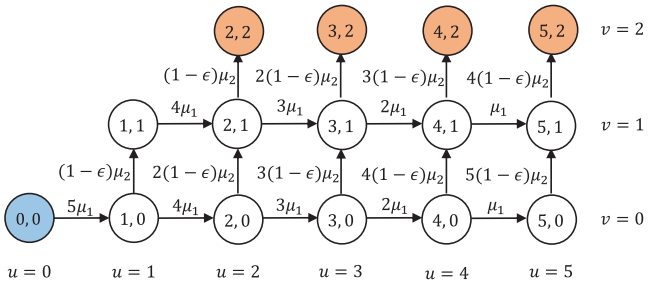



Fig. 9. State transition diagram example with $n = 5$, $k = 2$.

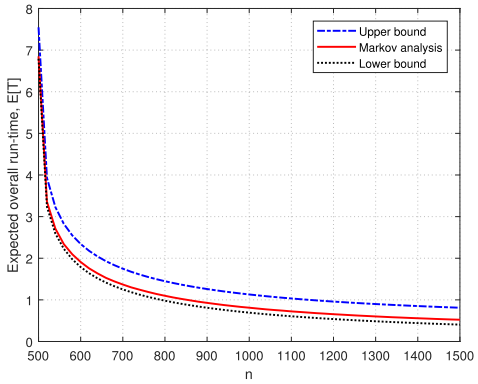



Fig. 10. Bounds of $\mathbb{E}[T]$ for $k = m = 500$, $\mu_1 = 1$, $\mu_2 = 10$, $\epsilon = 0.1$.

It can be seen from the transition rates $(n-u)\mu_1$ and $(u-v)(1-\epsilon)\mu_2$ that the expected hitting time of the Markov chain decreases with increasing $\mu_1$ (faster computing) or decreasing $\epsilon$ (lower retransmission probability).

While the exact $\mathbb{E}[T]$ can be obtained via Markov chain analysis, we additionally provide closed-form expressions for the lower and upper bounds of $\mathbb{E}[T]$ in the following theorem, to provide more insights and guidelines on designing the code.

*Theorem 3: Assuming $k = m$, we have $\mathcal{L} \leq \mathbb{E}[T] \leq \mathcal{U}$ where*

$$\mathcal{L} = \max_{i\in[k]}\left(\theta_1 + \frac{H_n - H_{n-k+i-1}}{\mu_1} + \frac{H_n - H_{n-i}}{(1-\epsilon)\mu_2}\right) \tag{13}$$

$$= \begin{cases} \theta_1 + \dfrac{1}{n\mu_1} + \dfrac{H_n - H_{n-k}}{(1-\epsilon)\mu_2}, & \epsilon \geq 1 - \dfrac{\mu_1}{\mu_2} \\ \theta_1 + \dfrac{H_n - H_{n-k}}{\mu_1} + \dfrac{1}{n(1-\epsilon)\mu_2}, & otherwise, \end{cases} \tag{14}$$

$$\mathcal{U} = \min_{i\in[n]\setminus[k-1]}\left(\theta_1 + \frac{H_n - H_{i-k}}{\mu_1} + \frac{H_n - H_{n-i}}{(1-\epsilon)\mu_2}\right). \tag{15}$$

*Proof:* See Appendix C. ∎

The lower bound in (13) can be rewritten as (14) with two regimes classified by equation $\epsilon = 1 - \frac{\mu_1}{\mu_2}$, which is equivalent to

$$\frac{1}{\mu_1} = \frac{1}{(1-\epsilon)\mu_2}, \tag{16}$$

i.e., when the computation time $X_i$ in (1) and the communication time $S_i$ in Lemma 1 have the same rate.

For $\epsilon \geq 1 - \frac{\mu_1}{\mu_2}$, the sum of the first two terms of the lower bound in (14), $\theta_1 + \frac{1}{n\mu_1}$, is the expected value of minimum computation time among $n$ workers. The last term is the expected value of the $k^{\text{th}}$ smallest communication time among $n$ workers. This is the case where all the workers finish the computation by time $\frac{1}{n\mu_1}$ and then start communication simultaneously, which is an intuitive lower bound. We show in Appendix C that the derived lower bound (14) is the tightest among the $k$ candidates of the bounds we obtained in (13). For the upper bound, by defining $H_0 = 0$, we have $\mathcal{U} = \theta_1 + \frac{H_n}{\mu_1} + \frac{H_n - H_{n-k}}{(1-\epsilon)\mu_2}$ by inserting $i = k$ at (15). Here, $\theta_1 + \frac{H_n}{\mu_1}$ is the expected value of maximum computation time among $n$ workers and the last term $\frac{H_n - H_{n-k}}{(1-\epsilon)\mu_2}$ is the expected value of the $k^{\text{th}}$ smallest communication time among $n$ workers. Again, this is an obvious upper bound as it can be interpreted as the case where all $n$ workers starts communication simultaneously right after the slowest worker finishes its computation. The derived upper bound includes this kind of scenarios. It can be also seen from the bounds that the lower and upper bounds decrease as $\mu_1$ increases or $\epsilon$ decreases.

For a given workload $m = k$, here we would like to find the best $n$. It can be easily seen from Theorem 3 that both $\mathcal{L}$ and $\mathcal{U}$ are decreasing functions of $n$. Fig. 10 shows the lower and upper bounds along with the hitting time of the Markov chain, with a varying number of workers $n$. The number of inner products to be computed is assumed to be $m = 500$. It can be seen that the lower bound is tighter than the upper bound. It can be also seen that not only the lower and upper bounds but also the exact expected latency decreases as $n$ increases. Therefore, increasing $n$ as much as possible is the optimal solution in minimizing $\mathbb{E}[T]$. In reducing the $R$-normalized expected latency $\mathbb{E}[T]/R$, we have the following theorem.

*Theorem 4: For a given workload $m = k$, $\mathbb{E}[T]/R$ is minimized by setting $n = k/R^*$, where*

$$R^* = \begin{cases} 1 + \dfrac{1}{W_{-1}\left(-e^{-\theta_1(1-\epsilon)\mu_2 - 1}\right)}, & \epsilon \geq 1 - \dfrac{\mu_1}{\mu_2} \\ 1 + \dfrac{1}{W_{-1}\left(-e^{-\theta_1\mu_1 - 1}\right)}, & otherwise, \end{cases} \tag{17}$$

*and $W_{-1}(\cdot)$ is the lower branch of Lambert $W$ function.*[3]

*Proof:* We first consider the case $\epsilon \geq 1 - \mu_1/\mu_2$. By approximating $\mathbb{E}[T]$ to the lower bound $\mathcal{L}$, we have

$$\frac{\mathbb{E}[T]}{R} \approx \frac{\mathcal{L}}{R} = \frac{1}{k}\left(n\theta_1 + \frac{1}{\mu_1} + \frac{n}{(1-\epsilon)\mu_2}\log\left(\frac{n}{n-k}\right)\right) \tag{18}$$

from Theorem 3. For a fixed $k = m$, we would like to find the minimizer of $\mathcal{L}/R$ with respect to $n$. From $\frac{\partial}{\partial n}(\frac{\mathcal{L}}{R}) = 0$, we have

$$\frac{n}{n-k} - \log\left(\frac{n}{n-k}\right) = 1 + \theta_1(1-\epsilon)\mu_2. \tag{19}$$

By setting $k = R^*n$, we have $\frac{1}{1-R^*} - \log(\frac{1}{1-R^*}) = 1 + \theta_1(1-\epsilon)\mu_2$. By defining $\beta = \frac{1}{1-R^*}$ and taking the exponential at both sides, we obtain $e^\beta/\beta = \exp(1 + \theta_1(1-\epsilon)\mu_2)$. Then by the definition of the Lambert $W$ function, we have

$$\beta = -W_{-1}(-e^{-\theta_1(1-\epsilon)\mu_2 - 1}). \tag{20}$$

[3] $W_{-1}(x)$ is the unique solution of $te^t = x$, $t \leq -1$.

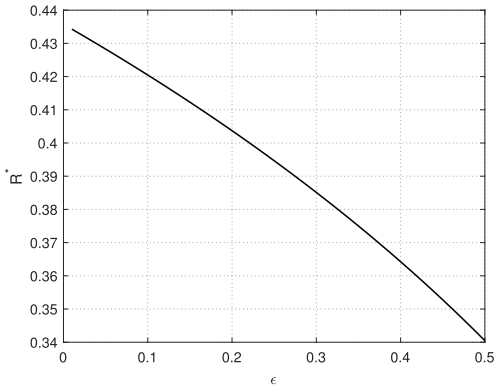



Fig. 11. $R^*$ obtained from Theorem 4, with $\theta_1 = 0.2$, $\mu_1 = 1$, $\mu_2 = 1$.

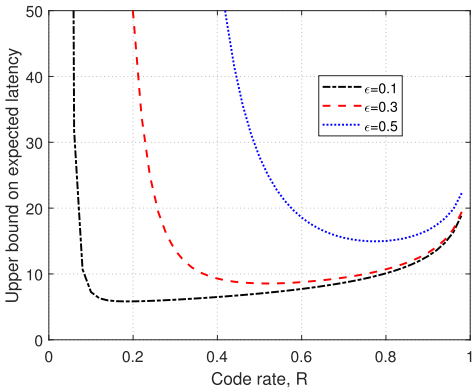



Fig. 12. The scenario where each worker transmits the entire result by a single packet. Upper bound on expected latency is shown for $n = 100$, $m = 500$, $\mu_1 = 1$, $\mu_2 = 10$.

Now by inserting this result to $\beta = \frac{1}{1-R^*}$, we have $R^* = 1 + \frac{1}{W_{-1}\left(-e^{-\theta_1(1-\epsilon)\mu_2 - 1}\right)}$ which completes the proof. We can similarly prove the case for $\epsilon < 1 - \mu_1/\mu_2$ from (14). ■

It is observed that the ratio $R^* = k/n^*$ obtained from Theorem 4 does not depend on $k = m$. Moreover, it can be seen from Theorem 4 that $R^*$ decreases as packet erasure probability $\epsilon$ increases, or $\mu_1$ and $\mu_2$ decrease. In Fig. 11, we plot $R^*$ with varying channel condition $\epsilon$. As expected, optimal $R$ decreases as $\epsilon$ increases, which indicates that it is more beneficial to increase $n$ in reducing $\mathbb{E}[T]/R$ for a given $m = k$ with $\mu_1 = \mu_2 = 1$ and a large $\epsilon$.

### D. Transmitting the Entire Result by a Single Packet

While our work considers transmitting multiple packets one by one, it is also possible to send the entire result by a single packet with a larger size. In this case, the computation time model is the same as in (1) and the communication time becomes $S_i = \sum_{j=1}^{N} Y_j$, where $Y_j$ follows an exponential distribution with rate $\frac{k}{m}\mu_2$ considering the increased packet length by the factor $\frac{m}{k}$. Here, $N$ follows a geometric distribution with success probability $1-\epsilon'$, where $\epsilon'$ is the new erasure probability of the packet containing the whole computational result at each worker. For analyzing $\epsilon'$, we assume that 1) each bit becomes erroneous independently according to the fixed bit-error rate (BER) of the system $\epsilon_b$, and 2) the packet is lost if at least one bit is erroneous [40]. Then, the packet erasure probability $\epsilon$ considered so far can be written as $\epsilon = 1-(1-\epsilon_b)^l$ where $l$ is the packet length for a single inner product. Now considering the entire result as a single packet, we have $\epsilon' = 1-(1-\epsilon_b)^{\frac{m}{k}l} = 1-(1-\epsilon)^{\frac{m}{k}}$. By following exactly the same procedure of the proof of Theorem 1, we can easily obtain the lower and upper bounds on expected latency as follows: $\mathcal{L} = \max_{i\in[k]}\left(\frac{m}{k}\theta_1 + \frac{H_n - H_{n-k+i-1}}{\mu_1} + \frac{m(H_n - H_{n-i})}{k(1-\epsilon')\mu_2}\right)$, $\mathcal{U} = \min_{i\in[n]\setminus[k-1]}\left(\frac{m}{k}\theta_1 + \frac{H_n - H_{i-k}}{\mu_1} + \frac{m(H_n - H_{n-i})}{k(1-\epsilon')\mu_2}\right)$. Fig. 12 shows the upper bound as a function of code rate $R$, when $n$ is given. The result is consistent with the trend in Fig. 3, indicating that a larger code rate is required in minimizing the upper bound with larger $\epsilon$.

## IV. Minimum Latency Analysis

In the previous section, we focused on analyzing the expected latency $\mathbb{E}[T]$, a performance metric that has been studied in many previous works on coded computation [11], [12], [24], [38], [42]. However, the expected latency alone may not be a sufficiently telling metric, especially for wireless networks where packet losses can create a long tail in the probability distribution of $T$.

### A. Minimum Latency

In this section, we introduce a new metric *minimum latency* $T^{(\alpha)}$, the minimum overall run-time that we can guarantee with probability $1-\alpha$:

$$T^{(\alpha)} = \min\{\tau : \Pr[T \le \tau] \ge 1-\alpha\}. \tag{21}$$

Here, $T$ is a random variable defined in (4) and $\alpha$ is the acceptable probability that the overall computation does not finish until $T^{(\alpha)}$. Since $T$ is the $k$-th order statistic of $\{X_i + S_i\}_{i=1}^n$, the probability $\Pr[T \le \tau]$ can be written as

$$\Pr[T \le \tau] = \sum_{j=k}^{n} \binom{n}{j} F(\tau)^j (1-F(\tau))^{n-j}, \tag{22}$$

where $F(\tau) = \int_{-\infty}^{\tau} f(z)dz$ and $f(z) = \int_{-\infty}^{\infty} f_X(y)\, f_S(z-y)dy$. Here,

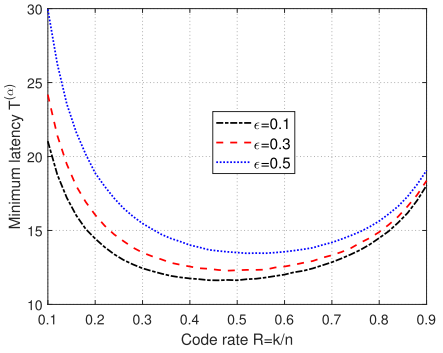
$$f_X(x) = \frac{k}{m}\mu_1 e^{-\frac{k}{m}\mu_1 x} \tag{23}$$

$$f_S(s) = \frac{\{(1-\epsilon)\mu_2\}^{\frac{m}{k}} s^{\frac{m}{k}-1} e^{-(1-\epsilon)\mu_2 s}}{(\frac{m}{k}-1)!} \tag{24}$$

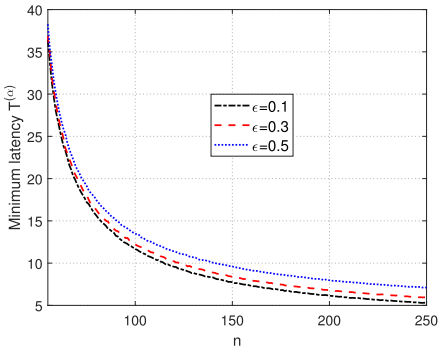
are the probability density functions of $X_i$ and $S_i$ defined in (1) and (3), respectively.

Fig. 13 shows the behavior of minimum latency $T^{(\alpha)}$ assuming $\alpha = 0.01$, $\mu_1 = 1$, $\mu_2 = 5$, $m = 500$. For a given $n = 100$, it can be seen from Fig. 13a that optimal code rate minimizing $T^{(\alpha)}$ increases as packet erasure probability $\epsilon$ increases, which is consistent with the result in Fig. 3. From Fig. 13b, it can be seen that the minimum latency decreases as we increase the number of workers $n$ for a given storage space at each node (for given $k$, $m$ as in Fig. 7). As channel becomes worse, more workers are required to achieve the same performance.

Similar to the $R$-normalized expected latency we considered in Section III-B, we observe the behavior of $R$-normalized

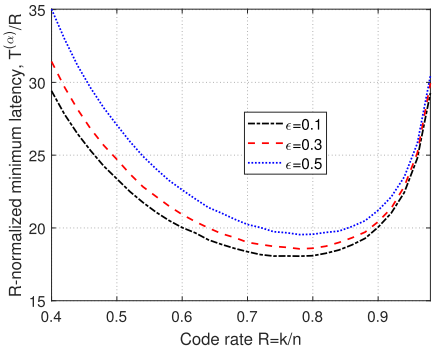



(a) Varying $k$, fixed $n = 100$.

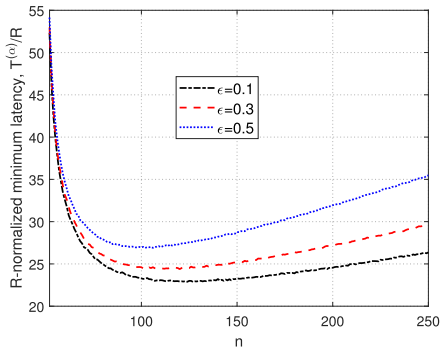



(b) Varying $n$, fixed $k = 50$.

Fig. 13. Performance of minimum latency $T^{(\alpha)}$.




(a) Varying $k$, fixed $n = 100$.




(b) Varying $n$, fixed $k = 50$.

Fig. 14. Performance of $R$-normalized minimum latency $T^{(\alpha)}/R$.

minimum latency $T^{(\alpha)}/R$ in Fig. 14. An important observation is that optimal code rate $R = k/n$ that minimizes $T^{(\alpha)}/R$ is significantly larger than the code rate that minimizes $T^{(\alpha)}$, which can be seen from Figs. 14a and 13a. This implies that reducing the total amount of computational burden of the system (increasing $k$) is effective in reducing the $R$-normalized minimum latency. Finally from Fig. 14b, it can be seen that simply increasing $n$ is not an appropriate solution in reducing $T^{(\alpha)}/R$. One can find the optimal number of workers $n$ depending on the channel condition $\epsilon$. Both Figs. 13 and 14 indicate that optimal $k$ or $n$ should be precisely chosen depending on the performance metric $T^{(\alpha)}$ or $T^{(\alpha)}/R$, and the channel condition $\epsilon$.

### *B. Analysis Under the Setting of Limited Number of Retransmissions*

So far, we focused on latency analysis over packet erasure channels under the assumption that there is no limit on the number of transmissions at each worker. The number of transmissions can be potentially infinite on the analysis above. However, in practical systems, the number of transmissions at each worker is often limited to avoid unacceptable time delay or due to bandwidth constraint at the computing devices. Let $\gamma$ be the maximum number of packets that is allowed to be transmitted at each worker. That is, the total number of transmissions of the system is limited to $n\gamma$.

When the number of packet transmissions allowed at each worker is given by $\gamma$, the minimum latency is rewritten as

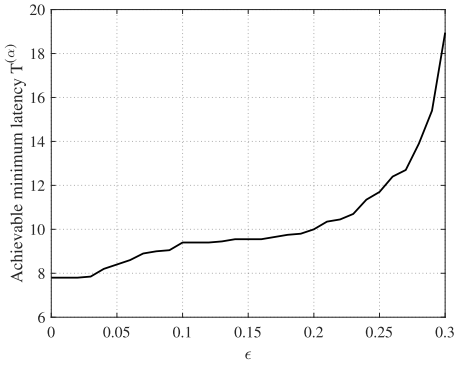
$$T^{(\alpha)} = \min\{\tau : \Pr[T' \leq \tau] \geq 1 - \alpha\}, \tag{25}$$

where $T' = \underset{i\in[n]}{k^{\text{th}}\min}(X_i + S_i')$ is the overall run-time until successful computation for a given $\gamma$. Here, we have $S_i' = \sum_{j=1}^{C_i} Y_j$ where $Y_j$ follows an exponential distribution with rate $\mu_2$ and $C_i$ is the number of transmissions at a specific worker until a successful transmission of $\frac{m}{k}$ packets, which are all independent for different workers. Let $p$ be the probability of success at a specific worker, i.e., the probability that a master node successfully receive $\frac{m}{k}$ out of $\gamma$ packets from a specific worker. Then, $p$ can be written as

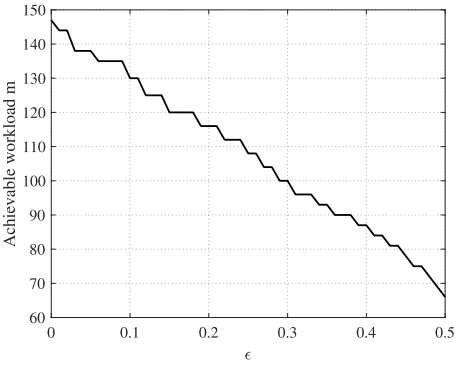
$$\begin{aligned} p &= \underbrace{(1-\epsilon)^{\frac{m}{k}}}_{0 \text{ failure}} + \underbrace{\binom{\frac{m}{k}}{1}(1-\epsilon)^{\frac{m}{k}}\epsilon}_{1 \text{ failure}} \\ &\quad + \ldots \underbrace{\binom{\gamma-1}{\gamma-\frac{m}{k}}(1-\epsilon)^{\frac{m}{k}}\epsilon^{\gamma-\frac{m}{k}}}_{\gamma-\frac{m}{k} \text{ failures}} \\ &= \sum_{i=0}^{\gamma-\frac{m}{k}} \binom{\frac{m}{k}+i-1}{i}(1-\epsilon)^{\frac{m}{k}}\epsilon^{i}. \end{aligned} \tag{26}$$

With probability $1 - p$, the worker fails to transmit the $\frac{m}{k}$ packets so that $C_i$ is undefined and $S_i'$ becomes infinity. Thus, we have random variable $C_i$ given by:

$$C_i = \begin{cases} \frac{m}{k}, & \text{w.p. } (1-\epsilon)^{\frac{m}{k}} \\ \frac{m}{k}+1, & \text{w.p. } \binom{\frac{m}{k}}{1}(1-\epsilon)^{\frac{m}{k}}\epsilon \\ \quad\vdots & \\ \gamma, & \text{w.p. } \binom{\gamma-1}{\gamma-\frac{m}{k}}(1-\epsilon)^{\frac{m}{k}}\epsilon^{\gamma-\frac{m}{k}} \\ \text{undefined}, & \text{w.p. } 1-p. \end{cases} \tag{27}$$




(a) Achievable $T^{(\alpha)}$ vs $\epsilon$ with $m = 120$.




(b) Achievable $m$ vs $\epsilon$ with $\tau_t = 10$.

Fig. 15. Achievable performance curves assuming $n = 40$, $\mu_1 = 1$, $\mu_2 = 5$, $\gamma = 7$, $\alpha = 0.01$.

Then, random variable $S_i'$ becomes

$$S_i' \begin{cases} \sim \text{erlang}(\frac{m}{k}, \mu_2) \text{ w.p. } (1-\epsilon)^{\frac{m}{k}} \\ \sim \text{erlang}(\frac{m}{k}+1, \mu_2) \text{ w.p. } \binom{\frac{m}{k}}{1}(1-\epsilon)^{\frac{m}{k}}\epsilon \\ \quad \vdots \\ \sim \text{erlang}(\gamma, \mu_2) \text{ w.p. } \binom{\gamma-1}{\gamma-\frac{m}{k}}(1-\epsilon)^{\frac{m}{k}}\epsilon^{\gamma-\frac{m}{k}} \\ = \infty \text{ w.p. } 1-p. \end{cases} \tag{28}$$

### C. Achievable Performance Curves

Now we consider two practical optimization problems. First is to minimize $T^{(\alpha)}$ for given $\gamma$ and $n$, where the achievable curve (as a function of $\epsilon$) is shown in Fig. 15a. The parameters are set as $n = 40$, $\mu_1 = 1$, $\mu_2 = 5$, $\gamma = 7$, $\alpha = 0.01$. We observe that the achievable $T^{(\alpha)}$ increases as the packet erasure probability $\epsilon$ grows but that the rate of increase varies. Finally, we consider another important problem which is to find the achievable workload $m$ of the current system under the constraint $T^{(\alpha)} \leq \tau_t$ (or $\Pr[T' \leq \tau_t] \geq 1-\alpha$) and for given $\gamma$, $n$. From Fig. 15b which shows the achievable workload as a function of channel condition $\epsilon$, it can be seen that achievable workload $m$ decreases more or less linearly as $\epsilon$ grows.

### D. Discussion on Extension to Other Codes

Although we utilized the MDS code for analysis, our result can be also applied to other recent codes targeting stragglers in distributed computing; a recent line of work on coded distributed computing focused on developing *polynomial codes* [13] rather than MDS codes. The authors consider recovery threshold as a performance metric, which is defined as the minimum number of workers that need to successfully send their results to the master in order to finish the overall task. Based on the recovery threshold, we can define a new metric called *minimum expected latency* to finish the task, and directly apply our analysis to extend the polynomial codes to a wireless setting. Given a recovery threshold $K_{\text{poly}}$ of a polynomial code, the minimum expected latency is equal to the expected latency for receiving any $K_{\text{poly}}$ out of $n$ results and thus we can directly apply our analytical tools.

## V. Conclusion

We studied the problem of coded computation assuming packet losses, which are a fact of life in wireless networks. Closed-form expressions of lower and upper bounds on the expected latency have been derived. We showed that the performance of the coding scheme minimizing the upper bound nearly achieved the optimal run-time, which showed the effectiveness of the proposed design. Considering the special case with $k = m$, we showed that the expected latency was equivalent to analyzing a continuous-time Markov chain. Then, we analyzed minimum latency, that has not been considered in previous works on coded computation. Based on the minimum latency, we provided guidelines on designing the optimal coding schemes. Finally, under the setting of a limited number of retransmissions at the computing devices, we obtained achievable performance curves of minimum latency and workload, as functions of packet erasure probability $\epsilon$. Extending our result to hierarchical computing systems [29], secure distributed computing systems [42] or to other variants of codes are interesting and important topics for future research.

## Appendix A
## Proof of Lemma 1

For $\beta = 1$, the communication time $S_i$ in (3) can be written as $S_i = \sum_{r=1}^{\frac{m}{k}} Z_r$ where $Z_r = \sum_{j=1}^{N_r} Y_{j,r}$. We first observe the distribution of $Z_r$. For a fixed integer $q$, $\sum_{j=1}^{q} Y_{j,1}$, the sum of $q$ exponential random variables with rate $\mu_2$, follows the erlang distribution with shape parameter $q$ and rate parameter $\mu_2$:

$$\Pr[\sum_{j=1}^{q} Y_{j,r} \leq t] = 1 - \sum_{p=0}^{q-1} \frac{1}{p!} e^{-\mu_2 t}(\mu_2 t)^p. \tag{29}$$

Now we can write

$$\begin{aligned} \Pr[Z_r \leq t] &= \sum_{q=1}^{\infty} \Pr[N_r = q] \Pr[\sum_{j=1}^{q} Y_{j,r} \leq t] \\ &= \sum_{q=1}^{\infty} ((1-\epsilon)\epsilon^{q-1})(1 - \sum_{p=0}^{q-1} \frac{1}{p!} e^{-\mu_2 t}(\mu_2 t)^p) \\ &= 1 - e^{-\mu_2 t} \sum_{q=1}^{\infty} \sum_{p=0}^{q-1} ((1-\epsilon)\epsilon^{q-1}) \frac{1}{p!}(\mu_2 t)^p \\ &= 1 - e^{-\mu_2 t} \sum_{p=0}^{\infty} \sum_{q=p+1}^{\infty} ((1-\epsilon)\epsilon^{q-1}) \frac{1}{p!}(\mu_2 t)^p \\ &= 1 - e^{-\mu_2 t} \sum_{p=0}^{\infty} \frac{1}{p!}(\epsilon \mu_2 t)^p \\ &= 1 - e^{-(1-\epsilon)\mu_2 t}. \end{aligned}$$

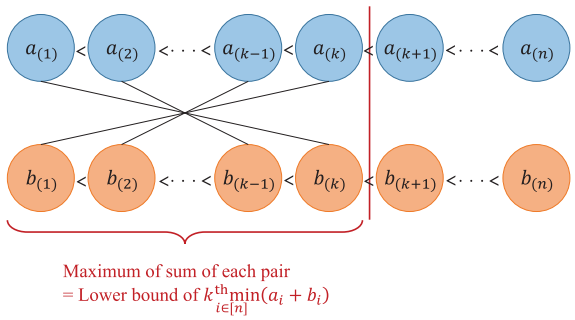



Fig. 16. Illustration of Proposition 1.

This implies that $Z_r$ follows an exponential distribution with rate $(1-\epsilon)\mu_2$. Thus, from $S_i = \sum_{r=1}^{\frac{m}{k}} Z_r$, the communication time $S_i$ follows an erlang distribution with shape parameter $\frac{m}{k}$ and rate parameter $(1-\epsilon)\mu_2$, which completes the proof.

## APPENDIX B
## PROOF OF THEOREM 1

### A. Lower Bound

We first prove the following proposition, which is illustrated as in Fig. 16.

*Proposition 1: For any two sequences* $\{a_i\}_{i\in[n]}$, $\{b_i\}_{i\in[n]}$,

$$k^{th}\underset{i\in[n]}{min}(a_i + b_i) \geq \underset{i\in[k]}{max}(a_{(k-i+1)} + b_{(i)}). \tag{30}$$

*Proof:* Define the ordering vector $\boldsymbol{\pi} \in \Pi$, where $\Pi$ is the set of permutation of $[n]$, i.e.,

$$\Pi = \{\boldsymbol{\pi} = [\pi_1, \ldots, \pi_n] : \pi_i \in [n]\ \forall i \in [n],\ \pi_i \neq \pi_j \quad for\ i \neq j\}. \tag{31}$$

Note that the $k^{\text{th}}$ smallest value out of $n$ can be always rewritten as the maximum of $k$ smallest values. Define $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_n] \in \Pi$ to make the set of values $\{a_{\pi_i} + b_{\pi_i}\}_{i=1}^{k}$ be the $k$ smallest among $\{a_i + b_i\}_{i=1}^{n}$. In addition, define $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 \in \Pi$ to make $a_{\pi_i} = a_{(\pi_{i,1})}$ and $b_{\pi_i} = b_{(\pi_{i,1})}$ be satisfied $\forall i \in [k]$, where $\boldsymbol{\pi}_1 = [\pi_{1,1}, \ldots, \pi_{n,1}]$, $\boldsymbol{\pi}_2 = [\pi_{1,2}, \ldots, \pi_{n,2}]$. Then, $k^{\text{th}}\min_{i\in[n]}(a_i + b_i)$ can be rewritten as

$$k^{\text{th}}\min_{i\in[n]}(a_i + b_i) = \max_{i\in[k]}(a_{\pi_i} + b_{\pi_i}) \tag{32}$$

$$= \max_{i\in[k]}(a_{(\pi_{i,1})} + b_{(\pi_{i,2})}). \tag{33}$$

Now define a new set of ordering vectors $\Pi^* \subseteq \Pi$, as

$$\Pi^* = \{\boldsymbol{\pi} \in \Pi : \pi_i \in [k] \quad \forall i \in [k]\}. \tag{34}$$

Then we can always construct $\boldsymbol{\pi}_1^*, \boldsymbol{\pi}_2^* \in \Pi^*$ such that $\pi_{i,1} \geq \pi_{i,1}^*$, $\pi_{i,2} \geq \pi_{i,2}^*$ for all $i \in [k]$, which is easy to prove. Then by adequately selecting $\boldsymbol{\pi}^* \in \Pi^*$, (33) is lower bounded as

$$\max_{i\in[k]}(a_{(\pi_{i,1})} + b_{(\pi_{i,2})}) \geq \max_{i\in[k]}(a_{(\pi_{i,1}^*)} + b_{(\pi_{i,2}^*)}) \tag{35}$$

$$= \max_{i\in[k]}(a_{(\pi_i^*)} + b_{(i)}). \tag{36}$$

Equation (35) implies that the lower bound of $\max_{i\in[k]}(a_{(\pi_{i,1})} + b_{(\pi_{i,2})})$ can be obtained by properly selecting the $k$ pairs $\{a_{(\pi_{i,1}^*)} + b_{(\pi_{i,2}^*)}\}_{i=1}^{k}$ all in the left region of Fig. 16, and taking the maximum. This value $\max_{i\in[k]}(a_{(\pi_{i,1}^*)} + b_{(\pi_{i,2}^*)})$ can be simply rewritten as (36). Now we want to show that for any $\boldsymbol{\pi}^* \in \Pi^*$, (36) cannot be smaller than $\max_{i\in[k]}(a_{(k-i+1)} + b_{(i)})$, which is the lower bound illustrated in Fig. 16.

Define $L$ as

$$L = \max_{i\in[k]}(a_{(k-i+1)} + b_{(i)}) = a_{(k-j+1)} + b_{(j)} \tag{37}$$

for some $j \in [k]$. Suppose there exist $L'$ and an ordering vector $\boldsymbol{\pi}' = [\pi_1', \ldots, \pi_n'] \in \Pi^*$ such that

$$\max_{i\in[k]}(a_{(\pi_i')} + b_{(i)}) = L' < L. \tag{38}$$

We will show that the assumption of (38) is a contradiction, which would complete the proof. By (38), $a_{(\pi_{j'}')} + b_{(j')} < a_{(k-j+1)} + b_{(j)}$ should be satisfied for all $j' \in [k]$. This implies $\pi_{j'}' < k - j + 1$ for all $j' \in [k] \setminus [j-1]$, which is a contradiction since one-to-one mapping between $j'$ and $\pi_{j'}'$ cannot be constructed for $j' \in [k]$. ∎

Since $\Pr[X_i \leq t] = 1 - e^{-\frac{k}{m}\mu_1(t - \frac{m}{k}\theta_1)}$ for general $k$, we have

$$\mathbb{E}[X_{(i)}] = \frac{m}{k}\theta_1 + \frac{m(H_n - H_{n-i})}{k\mu_1} \tag{39}$$

for all $i \in [n]$. For communication time, we can rewrite $S_i$ as

$$S_i = \frac{G_i}{(1-\epsilon)\mu_2} \tag{40}$$

where $G_i$ is an erlang random variable with shape parameter $\frac{m}{k}$ and rate parameter 1. We take this approach since a closed-form expression of $\mathbb{E}[G_{(i)}]$ exists as (7). Now the lower bound is derived as

$$\begin{aligned}\mathbb{E}[T] &= \mathbb{E}[k^{\text{th}}\min_{i\in[n]}(X_i + S_i)] \\ &\underset{(a)}{\geq} \mathbb{E}[\max_{i\in[k]}(X_{(k-i+1)} + S_{(i)})] \\ &\underset{(b)}{\geq} \max_{i\in[k]}\left(\frac{m}{k}\theta_1 + \frac{m(H_n - H_{n-k+i-1})}{k\mu_1} + \frac{\mathbb{E}[G_{(i)}]}{(1-\epsilon)\mu_2}\right)\end{aligned}$$

where $(a)$ holds from Proposition 1 and $(b)$ holds from the Jensen's inequality.

### B. Upper Bound

For the upper bound, we use the following proposition which is illustrated in Fig. 17.

*Proposition 2: For any two sequences* $\{a_i\}_{i\in[n]}$, $\{b_i\}_{i\in[n]}$,

$$k^{th}\underset{i\in[n]}{min}(a_i + b_i) \leq \underset{i\in[n]\setminus[k-1]}{min}(a_{(n+k-i)} + b_{(i)}). \tag{41}$$

*Proof:* We use the similar idea to that in the proof of Proposition 1. Consider a set of ordering vectors $\Pi$ defined in (31). Note that the $k^{\text{th}}$ smallest value out of $n$ can always be rewritten as the minimum of $n - k + 1$ largest values. We define $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_n] \in \Pi$ to make the set of values $\{a_{\pi_i} + b_{\pi_i}\}_{i=k}^{n}$ be the $n - k + 1$ largest among $\{a_i + b_i\}_{i=1}^{n}$. In addition, define $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 \in \Pi$ to make $a_{\pi_i} = a_{(\pi_{i,1})}$ and $b_{\pi_i} = b_{(\pi_{i,1})}$ be satisfied $\forall i \in [n] \setminus [k-1]$, where

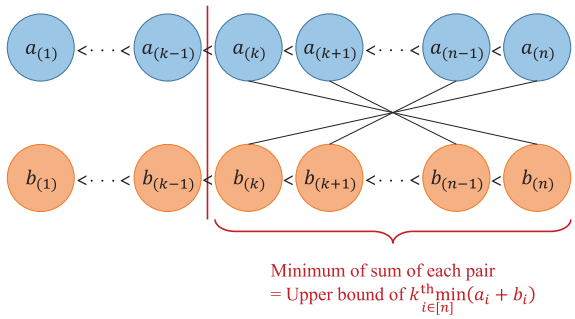



Fig. 17. Illustration of Proposition 2.

$\boldsymbol{\pi}_1 = [\pi_{1,1},\ldots,\pi_{n,1}]$, $\boldsymbol{\pi}_2 = [\pi_{1,2},\ldots,\pi_{n,2}]$. Then, $k^{\text{th}}\min_{i\in[n]}(a_i+b_i)$ can be rewritten as

$$k^{\text{th}}\min_{i\in[n]}(a_i+b_i) = \min_{i\in[n]\setminus[k-1]}(a_{\pi_i}+b_{\pi_i}) \quad (42)$$

$$= \min_{i\in[n]\setminus[k-1]}(a_{(\pi_{i,1})}+b_{(\pi_{i,2})}). \quad (43)$$

Now define another set of ordering vectors $\Pi^* \subseteq \Pi$ as

$$\Pi^* = \{\boldsymbol{\pi}\in\Pi : \pi_i \in [n]\setminus[k-1] \quad \forall i \in [n]\setminus[k-1]\}. \quad (44)$$

Then we can always construct $\boldsymbol{\pi}_1^*, \boldsymbol{\pi}_2^* \in \Pi^*$ such that $\pi_{i,1} \le \pi_{i,1}^*$, $\pi_{i,2} \le \pi_{i,2}^*$ for all $i \in [n]\setminus[k-1]$, which can be proved easily. Then by adequately selecting $\boldsymbol{\pi}^* \in \Pi^*$, (43) is upper bounded as

$$\min_{i\in[n]\setminus[k-1]}(a_{(\pi_{i,1})}+b_{(\pi_{i,2})}) \le \min_{i\in[n]\setminus[k-1]}(a_{(\pi_{i,1}^*)}+b_{(\pi_{i,2}^*)}) \quad (45)$$

$$= \min_{i\in[n]\setminus[k-1]}(a_{(\pi_i^*)}+b_{(i)}). \quad (46)$$

Equation (45) implies that the upper bound of $\min_{i\in[n]\setminus[k-1]}(a_{(\pi_{i,1})}+b_{(\pi_{i,2})})$ can be obtained by properly selecting the $n-k+1$ pairs $\{a_{(\pi_{i,1}^*)}+b_{(\pi_{i,2}^*)}\}_{i=k}^n$ all in the right region of Fig. 17, and taking the minimum. This value $\min_{i\in[n]\setminus[k-1]}(a_{(\pi_{i,1}^*)}+b_{(\pi_{i,2}^*)})$ can be simply rewritten as (46). Now we want to show that for any $\boldsymbol{\pi}^* \in \Pi^*$, (46) cannot be larger than $\min_{i\in[n]\setminus[k-1]}(a_{(n+k-i)}+b_{(i)})$, which is the upper bound illustrated in Fig. 17.

Define $U$ as

$$U = \min_{i\in[n]\setminus[k-1]}(a_{(n+k-i)}+b_{(i)}) = a_{(n+k-j)}+b_{(j)} \quad (47)$$

for some $j \in [n]\setminus[k-1]$. Suppose there exist $U'$ and an ordering vector $\boldsymbol{\pi'} = [\pi_1',\ldots,\pi_n'] \in \Pi^*$ such that

$$\min_{i\in[n]\setminus[k-1]}(a_{(\pi_i')}+b_{(i)}) = U' > U. \quad (48)$$

We show that the assumption of (48) is a contradiction. By (48), $a_{(\pi_{j'}')}+b_{(j')} > a_{(n+k-j)}+b_{(j)}$ should be satisfied for all $j' \in [n]\setminus[k-1]$. This implies $\pi_{j'}' > n+k-j$ for all $j' \in [j]\setminus[k-1]$, which is a contradiction since one-to-one mapping between $j'$ and $\pi_{j'}'$ cannot be constructed for $j' \in [n]\setminus[k-1]$. ■

Now from (39) and (39), the upper bound is derived as

$$\mathbb{E}[T] = \mathbb{E}[k^{\text{th}}\min_{i\in[n]}(X_i+S_i)] \quad (49)$$

$$\underset{(d)}{\le} \mathbb{E}[\min_{i\in[n]\setminus[k-1]}(X_{(n+k-i)}+S_{(i)})] \quad (50)$$

$$\underset{(e)}{\le} \min_{i\in[n]\setminus[k-1]}\Big(\frac{m}{k}\theta_1 + \frac{m(H_n-H_{i-k})}{k\mu_1} + \frac{\mathbb{E}[G_{(i)}]}{(1-\epsilon)\mu_2}\Big) \quad (51)$$

where $(d)$ holds from Proposition 2 and $(e)$ holds from the Jensen's inequality.

## APPENDIX C
## PROOF OF THEOREM 3

Since $\Pr[X_i \le t] = 1-e^{-\mu_1(t-\theta_1)}$ and $\Pr[S_i \le t] = 1-e^{-(1-\epsilon)\mu_2 t}$ for $k=m$, we have

$$\mathbb{E}[X_{(i)}] = \theta_1 + \frac{H_n-H_{n-i}}{\mu_1} \quad (52)$$

$$\mathbb{E}[S_{(i)}] = \frac{H_n-H_{n-i}}{(1-\epsilon)\mu_2}. \quad (53)$$

for all $i \in [n]$. Then, the lower bound is derived as

$$\mathbb{E}[T] = \mathbb{E}[k^{\text{th}}\min_{i\in[n]}(X_i+S_i)]$$

$$\underset{(a)}{\ge} \mathbb{E}[\max_{i\in[k]}(X_{(k-i+1)}+S_{(i)})]$$

$$\underset{(b)}{\ge} \max_{i\in[k]}\Big(\theta_1 + \frac{H_n-H_{n-k+i-1}}{\mu_1} + \frac{H_n-H_{n-i}}{(1-\epsilon)\mu_2}\Big)$$

$$\underset{(c)}{=} \begin{cases} \theta_1 + \dfrac{H_n-H_{n-1}}{\mu_1} + \dfrac{H_n-H_{n-k}}{(1-\epsilon)\mu_2}, & \epsilon \ge 1-\dfrac{\mu_1}{\mu_2} \\ \theta_1 + \dfrac{H_n-H_{n-k}}{\mu_1} + \dfrac{H_n-H_{n-1}}{(1-\epsilon)\mu_2}, & otherwise \end{cases}$$

where $(a)$ holds from Proposition 1 and $(b)$ holds from the Jensen's inequality. Now we prove $(c)$. Let

$$f_L = \theta_1 + \frac{H_n-H_{n-k+i-1}}{\mu_1} + \frac{H_n-H_{n-i}}{(1-\epsilon)\mu_2}. \quad (54)$$

Assuming $n$ is large and $k$ is linear in $n$, we can approximate $H_n \simeq \log(n)$, $H_{n-k} \simeq \log(n-k)$, and $H_{n-k+i-1} \simeq \log(n-k+i-1)$. Then,

$$\frac{\partial^2 f_L}{\partial i^2} = \frac{1}{(n-k+i-1)^2\mu_1} + \frac{1}{(1-\epsilon)(n-i)^2\mu_2} > 0 \quad (55)$$

which implies $f_L$ is a convex function of $i$. Since $i \in [k]$, the integer $i$ maximizing $f_L$ is either $i=1$ or $i=k$. Thus, $(c)$ is satisfied, which completes the proof.

For the upper bound, we have

$$\mathbb{E}[T] = \mathbb{E}[k^{\text{th}}\min_{i\in[n]}(X_i+S_i)] \quad (56)$$

$$\underset{(d)}{\le} \mathbb{E}[\min_{i\in[n]\setminus[k-1]}(X_{(n+k-i)}+S_{(i)})] \quad (57)$$

$$\underset{(e)}{\le} \min_{i\in[n]\setminus[k-1]}\Big(\theta_1 + \frac{H_n-H_{i-k}}{\mu_1} + \frac{H_n-H_{n-i}}{(1-\epsilon)\mu_2}\Big) \quad (58)$$

where $(d)$ holds from Proposition 2 and $(e)$ holds from the Jensen's inequality.

## REFERENCES


[1] D.-J. Han, J.-Y. Sohn, and J. Moon, "Coded distributed computing over packet erasure channels," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2019, pp. 717–721.

[2] D. Datla *et al.*, "Wireless distributed computing: A survey of research challenges," *IEEE Commun. Mag.*, vol. 50, no. 1, pp. 144–152, Jan. 2012.

[3] F. Li, J. Chen, and Z. Wang, "Wireless MapReduce distributed computing," *IEEE Trans. Inf. Theory*, vol. 65, no. 10, pp. 6101–6114, Oct. 2019.

[4] Y. He, J. Ren, G. Yu, and Y. Cai, "D2D communications meet mobile edge computing for enhanced computation capacity in cellular networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 3, pp. 1750–1763, Mar. 2019.

[5] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[6] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 1784–1797, Mar. 2018.

[7] J. Ren, G. Yu, Y. Cai, and Y. He, "Latency optimization for resource allocation in mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5506–5519, Aug. 2018.

[8] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2022–2035, Mar. 2020.

[9] M. M. Amiri and D. Gunduz, "Federated learning over wireless fading channels," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3546–3557, May 2020.

[10] G. Zhu, Y. Wang, and K. Huang, "Broadband analog aggregation for low-latency federated edge learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 491–506, Jan. 2020.

[11] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Trans. Inf. Theory*, vol. 64, no. 3, pp. 1514–1529, Mar. 2018.

[12] K. Lee, C. Suh, and K. Ramchandran, "High-dimensional coded matrix multiplication," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 2418–2422.

[13] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Polynomial codes: An optimal design for high-dimensional coded matrix multiplication," in *Proc. Neural Inf. Process. Syst.*, Long Beach, CA, USA, Dec. 2017, pp. 4406–4416.

[14] S. Dutta, V. Cadambe, and P. Grover, "'Short-Dot': Computing large linear transforms distributedly using coded short dot products," in *Proc. Neural Inf. Process. Syst.*, 2016, pp. 2092–2100.

[15] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding: Avoiding stragglers in distributed learning," in *Proc. Int. Conf. Mach. Learn.* 2017, pp. 3368–3376.

[16] L. Chen *et al.*, "DRACO: Byzantine-resilient distributed training via redundant gradients," in *Proc. ICML*, 2018, pp. 903–912.

[17] M. Ye and E. Abbe, "Communication-computation efficient gradient coding," in *Proc. Int. Conf. Mach. Learn.* 2018, pp. 5610–5619.

[18] A. Reisizadeh, S. Prakash, R. Pedarsani, and A. Salman Avestimehr, "CodedReduce: A fast and robust framework for gradient aggregation in distributed learning," 2019, *arXiv:1902.01981*. [Online]. Available: http://arxiv.org/abs/1902.01981

[19] Q. Yu, S. Li, N. Raviv, M. Kalan, M. Soltanolkotabi, and A. S. Avestimehr, "Lagrange coded computing: Optimal design for resiliency, security, and privacy," in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, 2019, pp. 1215–1225.

[20] J. Sohn, D.-J. Han, and J. Moon, "Election coding for distributed learning: Protecting SignSGD against byzantine attacks," in *Proc. Neural Inf. Process. Syst.*, 2020.

[21] S. Dutta, V. Cadambe, and P. Grover, "Coded convolution for parallel and distributed computing within a deadline," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 2403–2407.

[22] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Coded Fourier transform," in *Proc. 55th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Oct. 2017, pp. 494–501.

[23] A. Mallick and G. Joshi, "Rateless codes for distributed computations with sparse compressed matrices," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2019, pp. 2793–2797.

[24] A. Mallick, M. Chaudhari, U. Sheth, G. Palanikumar, and G. Joshi, "Rateless codes for near-perfect load balancing in distributed matrix-vector multiplication," in *Proc. Abstr. SIGMETRICS/Perform. Joint Int. Conf. Meas. Model. Comput. Syst.*, Jun. 2020, pp. 95–96.

[25] N. Ferdinand and S. C. Draper, "Hierarchical coded computation," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 1620–1624.

[26] S. Kiani, N. Ferdinand, and S. C. Draper, "Exploitation of stragglers in coded computation," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 1988–1992.

[27] A. Reisizadeh, S. Prakash, R. Pedarsani, and A. S. Avestimehr, "Coded computation over heterogeneous clusters," *IEEE Trans. Inf. Theory*, vol. 65, no. 7, pp. 4227–4242, Jul. 2019.

[28] M. Kim, J.-Y. Sohn, and J. Moon, "Coded matrix multiplication on a group-based model," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2019, pp. 722–726.

[29] H. Park, K. Lee, J.-Y. Sohn, C. Suh, and J. Moon, "Hierarchical coding for distributed computing," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 1630–1634.

[30] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "Coded MapReduce," in *Proc. 53rd Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2015, pp. 964–971.

[31] S. Li, M. A. Maddah-Ali, Q. Yu, and A. S. Avestimehr, "A fundamental tradeoff between computation and communication in distributed computing," *IEEE Trans. Inf. Theory*, vol. 64, no. 1, pp. 109–128, Jan. 2018.

[32] M. A. Attia and R. Tandon, "Information theoretic limits of data shuffling for distributed learning," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2016, pp. 1–6.

[33] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "A unified coding framework for distributed computing with straggling servers," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2016, pp. 1–6.

[34] S. Li, Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "A scalable framework for wireless distributed computing," *IEEE/ACM Trans. Netw.*, vol. 25, no. 5, pp. 2643–2654, Oct. 2017.

[35] M. Attia and R. Tandon, "Information theoretic limits of data shuffling for distributed learning," 2016, *arXiv:1609.05181*. [Online]. Available: http://arxiv.org/abs/1609.05181

[36] M. Gerami, M. Xiao, J. Li, C. Fischione, and Z. Lin, "Repair for distributed storage systems with packet erasure channels and dedicated nodes for repair," *IEEE Trans. Commun.*, vol. 64, no. 4, pp. 1367–1383, Apr. 2016.

[37] A. Reisizadeh and R. Pedarsani, "Latency analysis of coded computation schemes over wireless networks," in *Proc. 55th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Oct. 2017, pp. 1256–1263.

[38] K. Lee, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Coded computation for multicore setups," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 2413–2417.

[39] P. Bremaud, *Markov Chains, Gibbs Fields, Monte Carlo Simulation, and Queues*. Berlin, Germany: Springer, 1999.

[40] J. Korhonen and Y. Wang, "Effect of packet size on loss rate and delay in wireless links," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Mar. 2005, pp. 1608–1613.

[41] S. S. Gupta, "Order statistics from the gamma distribution," *Technometrics*, vol. 2, no. 2, pp. 243–262, May 1960.

[42] R. Bitar, P. Parag, and S. E. Rouayheb, "Minimizing latency for secure distributed computing," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 2900–2904.


**Dong-Jun Han** (Graduate Student Member, IEEE) received the B.S. degree in mathematics and electrical engineering and the M.S. degree in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST) in 2016 and 2018, respectively, where he is currently pursuing the Ph.D. degree. His research interests include distributed machine learning and information theory.

**Jy-Yong Sohn** (Member, IEEE) received the Ph.D. degree from the School of Electrical Engineering (EE), KAIST, in 2020. He is currently a Post-Doctoral Researcher with the Department of Electrical and Computer Engineering (ECE), University of Wisconsin–Madison. He is interested in the intersection of machine learning, information theory, and distributed systems. Especially, he focuses on research problems in distributed/federated learning and trustworthy machine learning. He was a recipient of the IEEE ICC 2017 Best Paper Award, the Qualcomm-KAIST Innovation Awards, the KAIST Global Leader Fellowship, the KAIST EE Best Research Achievement Award, and the NRF Korea Post-Doctoral Fellowship.

**Jaekyun Moon** (Fellow, IEEE) received the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA. From 1990 to 2009, he was a Faculty Member with the School of Electrical and Computer Engineering, University of Minnesota, Twin Cities. He consulted as the Chief Scientist of DSPG, Inc. from 2004 to 2007. He worked as the Chief Technology Officer with Link-A-Media Devices Corporation. He is currently a Professor of electrical engineering with KAIST. His research interests include channel characterization, signal processing and coding for data storage, and digital communication. He received the McKnight Land-Grant Professorship from the University of Minnesota. He also received the IBM Faculty Development Awards and the IBM Partnership Awards. He was awarded the National Storage Industry Consortium (NSIC) Technical Achievement Award for the invention of the maximum transition run (MTR) code, a widely used error-control/modulation code in commercial storage systems. He has served as the Program Chair for the 1997 IEEE Magnetic Recording Conference. He is also the Past Chair of the Signal Processing for Storage Technical Committee of the IEEE Communications Society. He has also served as a Guest Editor for the 2001 IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS (JSAC) Special Issue on Signal Processing for High Density Recording. He has also served as an Editor for IEEE TRANSACTIONS ON MAGNETICS in the area of signal processing and coding for 2001–2006.