Minimizing Latency for Secure Coded Computing Using Secret Sharing via Staircase Codes

Rawad Bitar, *Member, IEEE*, Parimal Parag, *Member, IEEE*, Salim El Rouayheb, *Member, IEEE*

***Abstract*—We consider the setting of a Master server, M, who possesses confidential data and wants to run intensive computations on it, as part of a machine learning algorithm for example. The Master wants to distribute these computations to untrusted workers who volunteered to help with this task. However, the data must be kept private in an information theoretic sense. Some of the workers may be stragglers, e.g., slow or busy. We are interested in reducing the delays experienced by the Master. We focus on linear computations as an essential operation in many iterative algorithms. We propose a solution based on new codes, called Staircase codes, introduced previously by two of the authors. Staircase codes allow flexibility in the number of stragglers up to a given maximum, and universally achieve the information theoretic limit on the download cost by the Master, leading to latency reduction. We find upper and lower bounds on the Master's mean waiting time. We derive the distribution of the Master's waiting time, and its mean, for systems with up to two stragglers. We show that Staircase codes always outperform existing solutions based on classical secret sharing codes. We validate our results with extensive implementation on Amazon EC2.**

I. Introduction

We consider the setting of distributed computing in which a server M, referred to as Master, possesses *confidential* data (e.g., personal, genomic or medical data) and wants to perform intensive computations on it. M wants to divide these computations into smaller computational tasks and distribute them to n *untrusted* worker machines that can perform these smaller tasks in parallel. The workers then return their results to the Master, who can process them to obtain the result of its original task.

R. Bitar and S. El Rouayheb are with the ECE department of Rutgers University. P. Parag is with the ECE department of the Indian Institute of Science.

Emails: rawad.bitar@rutgers.edu, parimal@iisc.ac.in, salim.elrouayheb@rutgers.edu.

Part of the work was presented at ISIT, 2017 [1]. Proofs omitted from this manuscript can be found in [2].

The work of the first and last authors was supported in part by NSF Grants CCF 1817635 and CNS 1801630 and by ARL Grant W911NF-17-1-0032. The work of the second author was supported in part by the Science and Engineering Research Board (SERB) under Grant No. DSTO-1677, the Department of Telecommunications, Government of India, under Grant DOTC-0001, the Robert Bosch Center for Cyber-Physical Systems, the Centre for Networked Intelligence (a Cisco CSR initiative) of the Indian Institute of Science, Bangalore. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

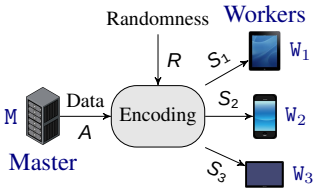
In this paper, we are interested in applications in which the worker machines do not belong to the same system or cluster as the Master. Rather, the workers are online computing machines that can be hired or can volunteer to help the Master in its computations, e.g., crowdsourcing platforms like the SETI@home [3] and folding@home [4] projects. The additional constraint, which we worry about here, is that the workers cannot be trusted with the sensitive data, which must remain hidden from them. Privacy could be achieved using fully homomorphic encryption that allows computing over encrypted data. However, homomorphic encryption incurs high computation and storage overheads [5], which may not be feasible in certain applications.

We propose information theoretic security to achieve the privacy requirement. Information theoretic security is typically used to provide privacy with no constraints on the computational power of the adversary (compromised workers). Our main motivation for information theoretic security is the low complexity of the resulting schemes (compared to homomorphic encryption). The assumption that we have to make here is a limit on the number of workers colluding against the Master.

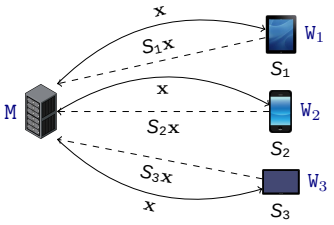
We focus on linear computations (matrix multiplication) since they form a building block of many iterative algorithms, such as principal component analysis, support vector machines and other gradient-descent based algorithms [6], [7]. The workers introduce random delays due to the difference of their workloads or network congestion. This causes the Master to wait for the slowest workers, referred to as stragglers in the distributed computing community [8], [9]. Our goal is to reduce the aggregate delay experienced by the Master.

Privacy can be achieved by encoding the data, with random keys, using linear secret sharing codes [10] as illustrated in Example 1. However, these codes are not specifically designed to minimize latency as we will highlight later.

Example 1. *Let the matrix A denote the data set owned by M and let $\mathbf{x}$ be a given vector. M wants to compute $A\mathbf{x}$. Suppose that M gets the help of 3 workers out of which at most 1 may be a straggler. M generates a random matrix R of same dimensions as A with entries drawn over the same alphabet as the entries of A. M encodes A and R into 3 shares $S_1 = R$, $S_2 = R + A$ and $S_3 = R + 2A$*



(a) The Master M encodes its data A with a random matrix R into 3 secret shares S_1, S_2, S_3. Any two shares can decode A. For example, $S_1 = R$, $S_2 = A + R$, and $S_3 = A + 2R$. M sends the share S_i to worker W_i. The randomness R is used to ensure privacy.



(b) To compute $A\mathbf{x}$, M sends $\mathbf{x}$ to all the workers. Each worker W_i computes $S_i\mathbf{x}$ and sends the result to M. M can decode $A\mathbf{x}$ after receiving any two responses, e.g., $A\mathbf{x} = S_2\mathbf{x} \quad S_1\mathbf{x} = (A+R)\mathbf{x} \quad R\mathbf{x}$.

Fig. 1: Secure distributed matrix multiplication with 3 workers. The Master encodes its data using a linear secret sharing code, e.g., Shamir's codes (given in the caption) [11], [12] or Staircase codes (given in Table I) [13], [14]. Decoding $A\mathbf{x}$ follows from the linearity of the code.

S_1	S_2	S_3
$A_1 + A_2 + R_1$	$A_1 + 2A_2 + 4R_1$	$A_1 + 3A_2 + 4R_1$
$R_1 + R_2$	$R_1 + 2R_2$	$R_1 + 3R_2$

TABLE I: The shares sent by M to each worker when using Staircase codes. In this example, each share is divided into two sub-shares. The operations shown are in $GF(5)$.

using a secret sharing scheme [11], [12]. M sends share S_i to worker W_i (Figure 1a) and then sends $\mathbf{x}$ to all the workers. Each worker computes $S_i\mathbf{x}$ and sends it back to M (Figure 1b). M can decode $A\mathbf{x}$ after receiving any 2 responses. For instance, if the first two workers respond, M can obtain $A\mathbf{x} = S_2\mathbf{x} \quad S_1\mathbf{x}$. No information about A is revealed to the workers, because A is one-time padded by R.

In the previous example, even if there were no stragglers, M still has to wait for the full responses of two workers, and the response of the third one will not be used for decoding. In addition, M always has to decode $R\mathbf{x}$ in order to decode $A\mathbf{x}$. Hence, more delays are incurred by spending communication and computation resources on decoding $R\mathbf{x}$, which is only needed for privacy. We overcome those limitations by using Staircase codes introduced in [13], [14] which do not always require decoding $R\mathbf{x}$. Thus, possibly reducing the computation load at the workers and the communication cost at the Master. In addition, Staircase codes allow more flexibility in the number of responses needed for decoding $A\mathbf{x}$, as explained in the next example.

Example 2 (Staircase codes)**.** *Consider the same setting as Example 1. Instead of using a classical secret sharing code, M now encodes A and R using the Staircase code given in Table I. The Staircase code requires M to divide the matrices A and R into $A = \begin{bmatrix} A_1 & A_2 \end{bmatrix}^T$ and $R = \begin{bmatrix} R_1 & R_2 \end{bmatrix}^T$. In this setting, M sends two sub-shares to each worker, hence each task consists of 2 sub-tasks. The Master sends $\mathbf{x}$ to all the workers. Each worker multiplies the sub-shares by $\mathbf{x}$ (going from top to bottom) and sends each multiplication back to M independently. Now, M has two possibilities for decoding: 1) M receives the first sub-task from all the workers, i.e., receives $(A_1 + A_2 + R_1)\mathbf{x}$, $(A_1 + 2A_2 + 4R_1)\mathbf{x}$ and $(A_1 + 3A_2 + 4R_1)\mathbf{x}$ and decodes $A\mathbf{x}$ which is the concatenation of $A_1\mathbf{x}$ and $A_2\mathbf{x}$. Note that here M decodes only $R_1\mathbf{x}$ and does not need to decode $R_2\mathbf{x}$. 2) M receives all the sub-tasks from any 2 workers and decodes $A\mathbf{x}$. Here M has to decode $R_1\mathbf{x}$ and $R_2\mathbf{x}$. Note that if M uses the first three sub-shares, it only decodes half of $R\mathbf{x}$, i.e., $R_1\mathbf{x}$, and does not need to decode $R_2\mathbf{x}$. This allows the master to save on communication cost. After receiving enough sub-tasks, the Master sends a message to the workers instructing them to stop computing the remaining sub-tasks. One can check that no information about A is revealed to the workers, because each sub-share is padded by a random matrix.*

A. Contributions

To the extent of our knowledge, this paper is the first work to analyze latency for private distributed coded computing under the presence of stragglers. We consider the distributed computing setting described above in which we require the workers to learn no information (in an information theoretic sense) about the Master's data. We study the waiting time of the Master caused by delays of the workers. We follow the literature, e.g., [6], [15], and model the service time at the workers as a shifted exponential random variable. This service time includes upload time, computation time and download time, i.e., computation and network latency. Finding codes that minimize the delay at the Master is still an open problem in general. In this work, we take

the download communication cost as a proxy for delay when designing the coding schemes. More precisely, we study the performance of the recently introduced Staircase codes [13], [14] that achieve the information theoretic limit on download cost [16] and compare them to classical secret sharing codes. The encoding and decoding at the Master add delays that are proportional to inverting a $k \times k$ matrix and multiplying this inverse by a vector of length k. The encoding and decoding complexities are of order $O(k \log k)$ when the generator matrix is a Vandermonde matrix. However, since the codes mentioned in this paper have same encoding and decoding complexities, we do not account for those delays in our latency analysis. Therefore, our delay analysis (and comparison to Staircase codes) is the same for codes that requires a threshold on the number of stragglers, such as [17], [18], and for classical secret sharing schemes.

Before we state our contributions, we introduce some necessary notations. We denote by n the number of workers available to help the Master, k, denote the minimum number of non stragglers and z the maximum number of colluding workers. We refer to such secure distributed computing system by a (n, k, z) system. We make the following contributions:

1) General bounds for systems with any number of stragglers: We derive an upper and a lower bound on the Master's mean waiting time when using Staircase codes (Theorem 1). Moreover, we derive the exact distribution of the Master's waiting time when using Staircase codes, in an integral form (Theorem 4). Using the upper bound, we compare the performance of Staircase codes to classical secret sharing codes and characterize the savings obtained by Staircase codes. We show that Staircase codes always outperform classical secret sharing codes.
2) Exact characterization for systems with up to 2 stragglers: We use the integral expression of Theorem 4 to find the exact distribution of the Master's waiting time for systems with up to $n - k = 1$ and up to $n - k = 2$ stragglers (Corollary 5). Moreover, we derive the exact expressions of the Master's mean waiting time for these systems (Theorem 2) and use these expressions to show the tightness of our upper bound.
3) Simulations and validation: We ran extensive MATLAB simulations for different system parameters. Our main observation is that the upper bound, based on Jensen's inequality, is a good approximation of the mean waiting time. Furthermore, we validate our results with extensive implementation on Amazon EC2 clusters. The savings obtained on EC2 clusters are within the range of the values predicted by the theoretical model. To give an example, for $n = 4$ workers, large data and high traffic regime, our implementation shows 59% (Figure 4a) savings in the mean waiting time while the theoretical model predicts 66% savings (Figure 3a).

B. Related work

The problem of stragglers has been identified and studied by the distributed computing community, see e.g., [8], [9], [19], [20]. Recently, there has been a growing research interest in studying codes for straggler mitigation and delay minimization in distributed systems with no secrecy constraints. The early body of work focused on minimizing latency of content download in distributed storage systems, see e.g., [15], [21]–[23] and later the focus has shifted to using codes for straggler mitigation in distributed computing, see e.g., [6], [7], [24]–[29].

Secure multiparty computation [30] can be used in this setting to provide privacy. However, the methods there are generic and not tailored to matrix multiplication and therefore do not have efficient communication cost and flexible straggler mitigation. The work that is closest to ours is [10] that studies the problem of distributively multiplying two private matrices under information theoretic privacy constraints using classical secret sharing codes. Subsequently to our initial result that have appeared in [1], several other works have studied different variants of this problem. In particular, [17], [18], [31]–[36] studied the problem of distributively multiplying two private matrices and [37]–[39] studied the problem of running private distributed machine learning algorithm in the presence of stragglers. In terms of delay analysis, all these works use schemes that assume a threshold on the number of stragglers and have similar delays as classical secret sharing schemes. However, the schemes used to multiply two private matrices require the master to use codes for both matrices which results in more tradeoffs between encoding complexity, straggler tolerance, upload cost and download cost see [18] for example.

In general, privacy in distributed computing is studied separately, mostly in the computer science community. Our work can also be related to the work on privacy-preserving algorithms, e.g., [40]–[43]. However, the privacy constraint in this line of work is computational privacy, and the proposed algorithms are not designed for straggler mitigation.

C. Organization

The paper is organized as follows. We formalize the problem and define the model in Section II. In Section III, we present and discuss our main results. We describe the construction of Staircase codes in Section IV. In Sections V and VI, we study the probability distribution of the Master's waiting time and derive bounds on the mean waiting time. We show, in Section VII, that the (random) number of workers that minimizes the waiting time is concentrated around its average. We evaluate the performance of Staircase codes via simulation in Section VIII. In Section IX, we give a representative sample of our implementation on Amazon EC2 clusters and compare them to our theoretical findings. We conclude the paper in Section X.

II. System Model

We consider a Master server M which wants to perform intensive computations on confidential data represented by an $m \times \ell$ matrix A (typically $m >> \ell$). M divides these computations into smaller computational tasks and assigns them to n workers W_i, $i = 1, \ldots, n$, that can perform these tasks in parallel. The division is horizontal, i.e., each worker gets a given number of rows of A with all their corresponding columns.

a) Computations model: We focus on linear computations. The motivation is that a building block in several iterative machine learning algorithms, such as gradient descent, is the multiplication of A by a sequence of $\ell \times 1$ attribute vectors $x^1, x^2, \ldots$. In the sequel, we focus on the multiplication Ax with one attribute vector x.

b) Workers model: The workers have the following properties: 1) The workers incur random delays while executing the task assigned to them by M, resulting in what is known as the straggler problem [6], [8], [9]. 2) Up to z, $z < k$, workers can collude, i.e., at most z workers can share with each other the data they receive from M. The threshold z could be thought of as a desired level of security. This has implications on the privacy constraint described later.

c) General scheme: M encodes A, using randomness, into n shares S_i sent to worker W_i, $i = 1, \ldots, n$. Any k or more shares can decode A, and any collection of z workers obtain zero information about A. For any set $\mathcal{B} \subset \{1, \ldots, n\}$, let $S_{\mathcal{B}} = \{S_i, i \in \mathcal{B}\}$ denote the collection of shares given to worker W_i for all $i \in \mathcal{B}$. The previous requirements can be expressed as,

$$H(A|S_{\mathcal{B}}) = 0, \quad \forall \mathcal{B} \subset \{1, \ldots, n\} \text{ s.t. } |\mathcal{B}| \geq k,$$
$$H(A|S_{\mathcal{Z}}) = H(A), \quad \forall \mathcal{Z} \subset \{1, \ldots, n\} \text{ s.t. } |\mathcal{Z}| \leq z.$$

At each iteration, the Master sends x to all the workers. Then, each worker computes $S_i x$ and sends it back to the Master. In the case where the share S_i consists of sub-shares, each worker multiplies the sub-shares by x and sends the result back to the master independently. Since the scheme and the computations are linear, the Master can decode Ax after receiving enough responses. After receiving enough responses the master sends a stop message to the workers instructing them to stop computing on the remaining sub-shares. We refer to such scheme as an (n, k, z) system. We note that our scheme can be generalized to the cases where the attribute vectors x contain information about A, and therefore need to be hidden from the workers. We describe the generalization of our scheme to such case in [2, Appendix].

d) Encoding: We consider classical secret sharing codes [11], [12] and universal Staircase codes [13], [14]. We describe their properties that are necessary for the delay analysis. Secret sharing codes require the division of A into $k - z$ row blocks each of dimension $\frac{m}{k-z} \times \ell$ and encodes them into n shares of identical dimension. Any k shares can decode A. Similarly, Staircase codes encode A into n shares of $\frac{m}{(k-z)} \times \ell$ each with the additional requirement that each share is divided into $b = \text{LCM}\{k - z + 1, \ldots, n - z\}$ sub-shares, where $\text{LCM}\{a, b, c\}$ denotes the least common multiple of a, b and c. Decoding A requires a fraction $\frac{d}{b}$ sub-shares, $\frac{d}{b} = \frac{(k-z)}{(d-z)}$, from any of the d shares, $d \in \{k, \ldots, n\}$. We provide a detailed explanation on the construction of Staircase codes in Section IV. We show that Staircase codes outperform classical codes in terms of incurred delays.

e) Delay model: Let T_A be the random variable representing the time spent to compute Ax at one worker. We assume a mother runtime distribution $F_{T_A}(t)$ that is shifted exponential with rate μ and a constant shift c. This is a popular model for modeling service time in a compute cluster [6], [15], and primarily motivated by its analytical tractability. Furthermore, the shifted exponential distribution captures the two parts of the service completion time: the constant part of the task-dependent service time at each server, and the stochasticity in service due to uncorrelated background processes at each server. For each $i \in \{1, \ldots, n\}$, we let T_i denote the time spent by worker W_i to execute its task. Due to the encoding, each task given to a worker W_i is z times smaller than A, or $T_i = \frac{T_A}{(k-z)}$. It follows that F_{T_i} is a scaled distribution of F_{T_A}. That is, for $t \geq c(k - z)$,

$$F_{T_i}(t) \triangleq F_{T_A}((k-z)t) = 1 - e^{-\mu(k-z)(t - \frac{c}{k-z})}. \quad (1)$$

We assume that the T_i's, $i = 1, \ldots, n$, are independent and identically distributed (iid). For an (n, k, z) system

$$E[T_{SC}] \quad \frac{c}{n \quad z} + \max_{d2f\, k;\cdots;n\, g} \sum_{i=0}^{k\ 1} \frac{n \quad X^{i} \quad i}{i} \sum_{j=0}^{} \frac{2(\quad 1)^{j}}{(2(n \quad i + j)(d \quad z) + (\ n \quad d)(n \quad d + 1))}: \tag{6}$$

$$E\,[T_{SC}(k+1\,;k;z)] = \frac{c}{k \quad z+1} + \frac{1}{-}\sum_{i=1}^{k+1}(\quad 1)^{i} \quad \frac{k+1}{i} \quad 4\frac{i\exp\ \frac{c}{k\ z}}{(k \quad z)i+1} \quad \frac{1}{(k \quad z+1)\,i}5: \tag{7}$$

$$E[T_{SC}(k+2\,;k;z)] = E[T_{SC}(k+2\,;k+1;z)] + \sum_{i=2}^{k+2}\frac{(\quad 1)^{i}}{} \quad \frac{k+2}{i} \quad \frac{i}{2} \quad 4\frac{\exp\ \frac{4\,c}{k\ z}}{(k \quad z)i+4} \quad \frac{2\exp\ \frac{3\,c}{k\ z}}{(k \quad z)i+3}5: \tag{8}$$

using Staircase codes, we assume that is evenly distributed among the sub-tasks[1]. That is, the time spent by a worker on one sub-task is equal to , and the time spent on sub-tasks is .

Let be the order statistic of the 's and be the time the Master waits until it can decode . If the aggregate wait is due to workers each nishing fraction of its sub-tasks, then the Master's waiting time is . We can write

$$T_{SC}(n;k;z) = \min_{d2f\, k;\cdots;n\, g} \quad {}_{d}T_{(d)} \quad : \tag{2}$$

It is useful for our analysis to look at as the sum of an exponential random variable and a constant offset, i.e. , where .

From this interpretation, it is easy to verify that the order statistic of can be expressed as

$$T_{(d)} = T^{0}_{(d)} + c=(k \quad z);$$

where is the order statistic of iid exponential random variables with rate . Therefore, we can write the Master's waiting time for Staircase codes as

$$T_{SC}(n;k;z) = \min_{d2f\, k;\cdots;n\, g} \quad {}_{d}\ T^{0}_{(d)} + \frac{c}{k \quad z} \quad : \tag{3}$$

For an system using classical secret sharing codes, the Master's waiting time is equal to the time spent by the fastest workers to nish their individual tasks. Hence, we can write

$$T_{SS}(n;k;z) = T_{(k)}: \tag{4}$$

[1]Therefore, we make two assumptions on the service time of the workers: (1) the distribution of service times at each worker is i.i.d., (2) at each worker, the service time of a sub-task is proportional to the fraction of the total task at the worker. Accordingly, the parameters of the sub-task distribution (shift and mean) vary linearly with the sub-task size. We observe that the service time of sub-tasks of the task computed at a given worker are proportional to their fractional size, and therefore are not independent. These assumptions make the problem more amenable to theoretical analysis. In Section IX, we compare our model to traces obtained from Amazon cloud and show that our model provides insightful engineering guidelines.

We drop the notation from and when the system parameters are clear from the context.

III. Our Results

Our results characterize the delay performance of secure coded computing when using Staircase codes and compare it to classical secret sharing codes. The performance of Staircase codes is re ected in the Master's waiting time . Towards our goal, we establish in Theorem 1 general bounds on the Master's mean waiting time when using Staircase codes for all systems, under the shifted exponential delay model.

Theorem 1 (Bounds on the Master's mean waiting time). Let be the harmonic sum de ned as , , with the notation , . The mean waiting time of the Master for an Staircase coded system is upper bounded by

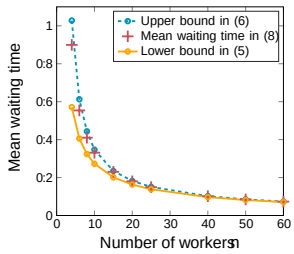
$$E[T_{SC}] \quad \min_{d2f\, k;\cdots;n\, g} \quad \frac{H_n \quad H_{n\ d}}{(d \quad z)} + \frac{c}{d \quad z} \quad : \tag{5}$$

The lower bounded is given in (6).

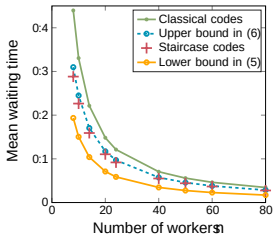
We derive in Section VI a general integral expression (19) leading to the CDF of , the waiting time of the Master for all systems. Using the general integral expression, we derive the exact expression of the CDF for systems with and as stated in the next theorem.

Theorem 2 (Exact expression of for systems with up to 2 stragglers). The mean waiting time of the Master for and systems is given in (7) and (8), respectively.

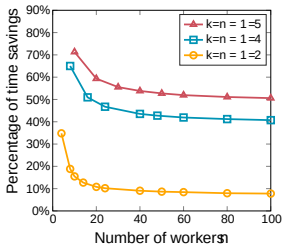
To give insights into the theoretical bounds above, we compare in Figure 2a bounds (5) and (6) for the case of to the exact expression in (8). We see that the upper bound in (5) is closer to the actual value and



(a) Systems with fixed $k = 2$.



(b) Systems with fixed $k=n = 1=2$.



(c) Savings: systems with fixed $k=n$.

Fig. 2: Theoretical upper and lower bounds for systems with rate of the exponential random variable $= 1$, shift $c = 1$ and no colluding workers, i.e. $z = 1$. Figure 2a compares the bounds derived in Theorem 1 to the theoretical mean waiting time for $(k + 2; k; 1)$ derived in Corollary 2. Observe that the upper bound in (5) is a good approximation of the mean waiting time in (8). Figure 2b compares the bounds in (5) and (6) to the simulated mean waiting time for $(n; k; z)$ systems with fixed rate $k=n = 1=2$. We obtain the mean waiting time by averaging over 10000 iterations for each value of n. Figure 2c compares the upper bound in (5) to the mean waiting time of classical secret sharing in (9). The savings are computed as the normalized difference between the waiting time of Staircase codes and classical secret sharing codes, $(E[T_{SC}] \quad E[T_{SS}]) = E[T_{SC}]$.

the gap between the two bounds closes as n increases. We also establish the comparison for fixed rate regimes, in particular rate $k=n = 1=2$. Since here $n \quad k + 2$, we compare in Figure 2b the bounds to numerical results obtained by simulation and observe the same behavior as before. We also plot in the same figure the mean waiting time for classical secret sharing codes obtained from (4) and given by

$$E[T_{SS}] = \frac{H_n \quad H_{n \quad k}}{(k \quad z)} + \frac{c}{k \quad z}: \qquad (9)$$

This allows to verify that Staircase codes always outperform classical secret sharing codes. In Figure 2c, we plot the lower bound on the relative savings brought by Staircase codes for systems with rate $k=n = 1=2; 1=4; 1=5$. For instance, for rate $1=4$, the savings are lower bounded by 40% for large n. We supplement our theoretical results in Section VIII with an extensive array of simulations in addition to measurement results obtained by implementation on Amazon EC2 clusters. The savings obtained in the implementation on Amazon cloud are within the savings predicted by the theoretical model.

IV. Staircase codes

We briefly explain the encoding and decoding of Staircase codes. Let A be an m $`$ matrix with elements drawn uniformly at random from a finite alphabet, e.g. a finite field[2] $GF(q)$. An $(n; k; z)$ Universal[3] Staircase code [13], [14] allows the Master to encode A into n

$$M_{SC} = \begin{matrix} 2 \\ 6 \\ 6 \\ 6 \\ 6 \\ 4 \end{matrix} \begin{matrix} S & D_1 & D_2 & \cdots & D_{h \quad 1} \\ & R_1 & R_2 & R_3 & \cdots & R_h \\ & 0 & 0 & \cdots & 0 \end{matrix} \begin{matrix} 3 \\ 7 \\ 7 \\ 7 \\ 7 \\ 5 \end{matrix}$$

staircase structure

$M_1 \quad M_2 \quad M_3 \quad \cdots \quad M_h$

TABLE II: The structure of the matrix M_{SC} that contains the secret and keys in the universal Staircase code construction [14].

shares and distribute them to n workers. In addition to privacy against any z colluding workers, Staircase codes enjoy the secret reconstruction with minimum communication cost property. The Master can reconstruct the secret by contacting any set of k d n, workers and downloading a part of their shares. The information theoretic lower bound on the amount of information downloaded from each worker is referred to as communication cost $CC(d)$ and is given [16] by

$$CC(d) = \frac{k \quad z}{d \quad z}: \qquad (10)$$

Encoding: Let V be an n n Vandermonde matrix defined over $GF(q)$, q n. Let M_{SC} be the matrix defined in Table II and detailed next. The encoding of Staircase codes consists of multiplying A by M_{SC} to obtain the matrix $C = V M_{SC}$. The n rows of C form the n shares.

To construct the matrix M defined in Table II, an $(n; k; z)$ Staircase code requires dividing the data matrix

[2]The computation can be carried over the reals by using unbiased quantization as in [38] and references within.

[3]We only describe Universal Staircase codes [14] and shall refer to them as Staircase codes.

A into $b(k-z)$ matrices $A_1, \ldots, A_{(k-z)b}$ each of dimension[4] $m/b \times (k-z)$, where $b = \text{LCM}\{k+1, \ldots, n-z\}$. Let $d_1 = n, d_2 = n-1, \ldots, d_h = k$ denote the number of workers contacted by the Master, with $h = n-k+1$. Let b_i, $d_i - z$ for $i = 1, \ldots, h$. The data matrices are arranged in a $bm/(k-z)b \times (k-z)b = b_1$ matrix S. To ensure secrecy, the construction uses zb matrices $R_1, \ldots, R_{zb}$ of dimension $m/b \times (k-z)$ each and with elements drawn independently and uniformly at random from $GF(q)$. The random matrices $R_1, \ldots, R_{zb}$ are partitioned into b matrices R_i, $i = 1, \ldots, b$, each of dimension $zm/(k-z)b \times (k-z)b = b_{i-1} \times b_0 = 1$. The matrix M_{SC} is the concatenation of matrices M_i, $i = 1, \ldots, b$. Each matrix M_i consists of the b_i subtasks downloaded by the Master when decoding from d_i workers, i.e., when there are $n - d_i$ stragglers.

The elements appearing in each matrix D_j are the elements of the $(n-j+1)^{th}$ row of $M_1 M_2 \cdots M_j$ rearranged to obtain the dimension of D_j as $mb_{j+1}/(k-z)b \times (k-z)b/b_{j+1}$ for $j = 1, \ldots, b-1$. The 0's are the all zero matrices used to complete M_i's to $nm/(k-z)b$ rows.

The structure of the matrix M_{SC}, called Staircase structure, allows the Master to decode the secret and achieve optimal communication and read overheads CO and RO for all $d, k \le d \le n$.

Decoding: The Master contacts any d_i workers, $i = 1, \ldots, b$ and downloads $M_1 \cdots M_i$ from each contacted worker. The Master is guaranteed [14, Theorem 2] to decode the secret.

In the setting of secure distributed computing, the Master encodes A and sends the n shares to the workers. To compute Ax, M sends x to the workers and waits for the first d, $k \le d \le n$, workers to send part of their shares multiplied by x. Since the multiplication is linear, the Master can decode Ax and part of the random matrices $R_i x$, $i = 1, \ldots, zb$.

V. Bounds on the Master's mean waiting time for all (n, k, z) systems

We derive an upper and a lower bound on the Master's mean waiting time $E[T_{SC}(n,k,z)]$ for all (n,k,z) systems, i.e., we prove Theorem 1.

A. Proof of the upper bound on the mean waiting time

Proof: We use Jensen's inequality to upper bound the mean waiting time $E[T_{SC}]$. Since min is a convex function, we can use Jensen's inequality to upper bound the mean waiting time,

$$E[T_{SC}] = E\left[\min_{d \in \{k,\ldots,n\}} \frac{d}{d-z} T^0_{(d)} + \frac{c}{d-z}\right] \le \min_{d \in \{k,\ldots,n\}} \frac{d}{d-z} E\left[T^0_{(d)}\right] + \frac{c}{d-z}. \quad (11)$$

We need the following theorem in order to derive an exact expression of the mean of the d^{th} order statistic of n iid exponential random variables.

Theorem (Renyi [44]). The d^{th} order statistic $T^0_{(d)}$ of n iid exponential random variables T^0_i is equal to the following random variable in the distribution

$$T^0_{(d)} \sim \sum_{j=1}^{d} \frac{T^0_j}{n-j+1}.$$

Using Renyi's theorem, the mean of the d^{th} order statistic $E\left[T^0_{(d)}\right]$ can be written as

$$E[T^0_{(d)}] = E[T^0_j] \sum_{j=0}^{d-1} \frac{1}{n-j} = \frac{H_n - H_{n-d}}{(k-z)}. \quad (12)$$

From equations (11) and (12), the mean waiting time is upper bounded by

$$E[T_{SC}] \le \min_{d \in \{k,\ldots,n\}} \frac{H_n - H_{n-d}}{(d-z)} + \frac{c}{d-z}.$$

We give an intuitive behavior of the upper bound. The harmonic number can be approximated by $H_n \approx \log(n) + \gamma$, where $\gamma \approx 0.577218$ is called the Euler-Mascheroni constant. Alternatively, we can use the upper and lower bounds $\log(n) < H_n < \log(n+1)$ on the Harmonic number H_n, to upper bound the mean waiting time

$$E[T_{SC}] < \min\{\alpha_1, \alpha_2\},$$
$$\alpha_1 \approx \min_{d \in \{k,\ldots,n-1\}} \frac{1}{(d-z)} \log\left(\frac{n+1}{n-d}\right) + \frac{c}{d-z},$$
$$\alpha_2 \approx \frac{1}{(n-z)} \log(n+1) + \frac{c}{n-z}.$$

■

B. Proof of the lower bound on the mean waiting time

Proof: Recall that $T_{SC} = \min\{{}_d T_{(d)} : d \in \{k, \ldots, n\}\} = \min\{{}_d T^0_{(d)} + \frac{c}{d-z} : d \in \{k, \ldots, n\}\}$. Since the minimum of the sum is greater than the sum of the minimums, we can lower bound the waiting time T_{SC}

[4] If the number of rows in A is not divisible by b, one can use zero padding or the representation of A in a smaller field $GF(q_1)$ such that $q = q_1^b$.

$$\prod_{j=d+1}^{n} \Pr\left(\frac{T'_j}{n-j+1} > \frac{t}{(k-z)}\right) = F\left(\sum_{j=d+1}^{n} (n-j+1)\,t\right) = F\left(\frac{(n-d)(n-d+1)\,t}{2}\right): \quad (14)$$

$$\Pr(C_d(t)) = \sum_{i=0}^{k-1} \binom{n}{i} \sum_{j=0}^{i} \binom{i}{j} (-1)^j F\left(t(n-i+j)(d-z) + t(n-d)(n-d+1)/2\right): \quad (17)$$

$$E[T'_{SC}] = \int_0^{\infty} \Pr\left(T'_{SC} > t\right) dt \le \int_0^{\infty} \max_{d\in\{k,\ldots,n\}} \Pr(C_d(t))\,dt \le \max_{d\in\{k,\ldots,n\}} \int_0^{\infty} \Pr(C_d(t))\,dt: \quad (18)$$

---

in terms of residual waiting times $T'_{SC} = \min\{\frac{1}{d} T'_{(d)} : d \in \{k,\ldots,n\}\}$, as

$$T_{SC} = \min_{d\in\{k,\ldots,n\}} \left\{\frac{1}{d} T'_{(d)} + \frac{c}{d-z}\right\} \le T'_{SC} + \frac{c}{(n-z)}.$$

Since the mean of a continuous random variable can be computed by integrating the tail probability, we lower bound $E[T'_{SC}]$ by lower bounding the tail probability of T'_{SC} exceeding any threshold value t. We observe that T'_{SC} is greater than t, if and only if the d^{th} order statistic $T'_{(d)}$ is greater than $\frac{t}{d}$ for each $d \in \{k,\ldots,n\}$. That is,

$$\{T'_{SC} > t\} = \bigcap_{d=k}^{n} T'_{(d)} > \frac{t}{d}.$$

Recall that $\frac{1}{d}(k-z) = t(d-z)$ is increasing in d, and so is $T'_{(d)}$. For the residual service times $T'_1,\ldots,T'_n$, we consider $C_d(t)$ defined as the following set

$$T'_{(k)} > \frac{t}{d} \bigcap_{i=d+1}^{n} T'_{(i)} \quad T'_{(i-1)} > \frac{t}{i} - \frac{t}{i-1}.$$

For each $d \in \{k,\ldots,n\}$, we observe that $C_d(t) \subseteq \{T'_{SC} > t\}$ since $\{T'_{(k)} > t\frac{1}{d}\} \cap \bigcap_{j=k}^{d} \{T'_{(j)} > t\frac{1}{j}\}$. It follows that, $\Pr\{T'_{SC} > t\} \le \max_{d\in\{k,\ldots,n\}} \Pr(C_d(t))$. Next, we evaluate $\Pr(C_d(t))$ explicitly. To this end, we first observe that $\frac{1}{j} - \frac{1}{j-1} = \frac{1}{(k-z)}$ identically for each $j \in \{2,\ldots,n\}$. Further, we apply Renyi's theorem and independence of residual times T'_js to write

$$\Pr(C_d(t)) = \Pr\left(T'_{(k)} > \frac{t}{d}\right) \prod_{j=d+1}^{n} \Pr\left(\frac{T'_j}{n-j+1} > \frac{t}{(k-z)}\right): \quad (13)$$

In the following, we would use $F(t) = 1 - e^{-t}$ for $t \ge 0$ to represent the cumulative distribution function (CDF) and $\bar{F}(t) = 1 - F(t)$ to represent the complementary cumulative distribution function (CCDF), of an exponential random variable with rate. It follows that the CCDF for the residual service time T'_j is $\Pr\{T'_j > t\} = \bar{F}((k-z)t)$. Utilizing the exponential form, we can write (14).

From definition, it follows that $\frac{1}{k} = 1$. Further, the k^{th} order statistic of n residual service times exceeds a threshold if and only if at most $k-1$ different residual service times are less than the threshold, c.f., Lemma 3. That is,

$$\Pr\left\{T'_{(k)} > t\right\} = \sum_{i=0}^{k-1} \binom{n}{i} F((k-z)t)^i \bar{F}((k-z)t)^{n-i}: \quad (15)$$

Since $\bar{F}(t) = 1 - F(t)$, using the binomial expansion, we have

$$\bar{F}((k-z)t)^i = \sum_{j=0}^{i} \binom{i}{j} (-1)^j F((k-z)t)^j: \quad (16)$$

Exploiting the exponential form of $F(t)$, aggregating results from (14), (15) and (16), we can re-write (13) as (17). The proof follows from the integral $\int_0^t e^{-xt}\,dt = \frac{1}{x}$, the linearity of integrals, and the lower bound (18). ■

Lemma 3. Marginal complementary distribution of d^{th} order statistic $T'_{(d)}$ of n iid random variables $(T'_1,\ldots,T'_n)$ with common distribution $f_{T'}(t)$ is given by

$$\Pr\{T'_{(d)} > t\} = \sum_{i=0}^{d-1} \binom{n}{i} F_{T'}(t) \bar{F}_{T'}(t)^{n-i}.$$

We note the cumulative distribution function (CDF) of f by $F_{T'}(t)$, $f_{T'}(T' < t)$ and the complementary cumulative distribution function (CCDF) of f by $\bar{F}$, $f_{T'}(T' > t) = 1 - F_{T'}(t)$.

Proof: The d^{th} order statistic is greater than t, if and only if at most $d-1$ out of n iid random variables $(T_1,\ldots,T_n)$ can be less than t, and the rest are greater than t. ■

$$F_{T_{SC}}(t) = 1 - n! \int_{(y_k,\ldots,y_n)\in A(t)} \frac{F_{T^0}(y_k)^{k-1}}{(k-1)!} dF_{T^0}(y_k)\ldots dF_{T^0}(y_n) \quad \text{for } t > 0. \tag{19}$$

$$F_{T_{SC(k+1,k,z)}}(t) = F_{T^0}(t_{k+1})^{k+1} + F_{T^0}(t_k)^k F_{T^0}(t_{k+1})(k+1). \tag{20}$$

$$F_{T_{SC(k+2,k,z)}}(t) = F_{T^0}(t_{k+2})^{k+2} + (k+2)F_{T^0}(t_{k+2})\left[F_{T^0}(t_{k+1})^{k+1} + (k+1)F_{T^0}(t_k)^k\left(F_{T^0}(t_{k+1}) - \frac{1}{2}F_{T^0}(t_{k+2})\right)\right]. \tag{21}$$

$$\Pr\{T_{SC} > t\} = \int_{y\in\hat{A}(t)} dF_{T^0_{(1)},\ldots,T^0_{(n)}}(y) = n! \int_{t_n}^{\infty}\int_{t_k}^{y_{k+1}}\cdots\int_{t_1}^{y_2}\prod_{i=k}^{n} dF_{T^0}(y_i)\int_0^{y_k}\cdots\int_0^{y_2}\prod_{i=1}^{k-1} dF_{T^0}(y_i). \tag{22}$$

VI. Distribution of the Master's waiting time for all (n,k,z) systems

Now we are ready to derive an integral expression for the probability distribution of T_{SC}, the Master's waiting time when using Staircase codes.

Theorem 4 (Integral expression leading to $F_{T_{SC}}(t)$)**.** The distribution of the Master's waiting time T_{SC} of an (n,k,z) system using Staircase codes is given in (19).

We denote the residual service time at each worker W_i, $i = 1,\ldots,n$, by the random variable $T_i^0 = T_i - \frac{c}{k-z}$, and the associated distribution by $F_{T^0}(y_i)$, $F_{T^0}(y_i) = 1 - \exp(-\mu y_i)$ for $y_i > 0$. For $i = k,\ldots,n$, we define t_i as $t_i \triangleq \max\left\{\frac{i-z}{k-z}t - \frac{c}{i-z}, 0\right\}$. We denote by $A(t)$ the set of ordered variables $(y_k,\ldots,y_n)$ defined as

$$\{0 \le y_k \le y_{k+1} \le \cdots \le y_n : t_k < y_k,\ldots,t_n < y_n\}.$$

We apply Theorem 4 to get the mean waiting time of the Master and the exact distribution of the waiting time for systems with $n = k+1$ and $n = k+2$ in Theorem 2 and Corollary 5, respectively.

Corollary 5 (Exact expression of $F_{T_{SC}}(t)$ for systems with up to 2 stragglers)**.** The distribution of the Master's waiting time for $(k+1,k,z)$ and $(k+2,k,z)$ systems is given in (20) and (21), respectively. Both distributions are defined for $t > 0$, and $F_{T^0}(t) = 1 - \exp(-(k-z)\mu t)$.

We omit the proof of Corollary 5 since it follows from simply integrating (19) and defer the proof of Theorem 2 to the technical report [2].

Proof of Theorem 4: Let T_i^0 denote the residual service time of worker i with the offset $\frac{c}{k-z}$. The sequence $(T_1^0,\ldots,T_n^0)$ of residual service times of n workers is assumed to be i.i.d and distributed exponentially with rate $(k-z)\mu$ with the tail-distribution function $F_{T^0}(t) = e^{-(k-z)\mu t}$ for $t > 0$.

Since the common distribution of residual service times is absolutely continuous with respect to the Lebesgue measure, the corresponding probability density exists and is denoted by $f_{T^0}(t) = dF_{T^0}(t)/dt = (k-z)\mu e^{-(k-z)\mu t}$ for $t > 0$. Further, we know that the order statistics $(T^0_{(1)},\ldots,T^0_{(n)})$ of residual times $(T_1^0,\ldots,T_n^0)$ is identical for all their $n!$ permutations. Hence, for any $0 \le y_1 \le \cdots \le y_n$, we can write $f_{T^0_{(1)},\ldots,T^0_{(n)}}(y_1,\ldots,y_n) = n! f_{T^0_1,\ldots,T^0_n}(y_1,\ldots,y_n) = n!\prod_{i=1}^{n} f_{T^0}(y_i)$. The product form of joint density follows from the independence of the residual service times.

In terms of $\mu_j = \frac{k-z}{j-z}\mu$, the order statistics of residual times $T^0_{(j)}$, and the offset $\frac{c}{k-z}$, we can write

$$\{T_{SC} > t\} = \bigcap_{j=k}^{n}\left\{T^0_{(j)} > \frac{t}{j} - \frac{c}{j-z}\right\}.$$

For each $k \le j \le n$, we define $t_j \triangleq \max\left\{\frac{t}{j} - \frac{c}{j-z}, 0\right\}$, $y_{n+1} \triangleq \infty$, and $\hat{A}(t) \triangleq \bigcap_{j=k}^{n+1}\{t_j < y_j \le y_{j+1}\} \cap \bigcap_{j=1}^{k-1}\{0 \le y_j \le y_{j+1}\}$. In terms of t_j, y_{n+1} and $\hat{A}(t)$, we can write the tail distribution as in (22).

First, we compute the integral with respect to ordered non-negative real variables $(y_1,\ldots,y_{k-1})$ over the region $B_{k-1} \triangleq \bigcap_{j=1}^{k-1}\{0 \le y_j \le y_{j+1}\}$, a projection of $\hat{A}(t)$ on $(k-1)$ dimensional space spanned by $(y_1,\ldots,y_{k-1})$.

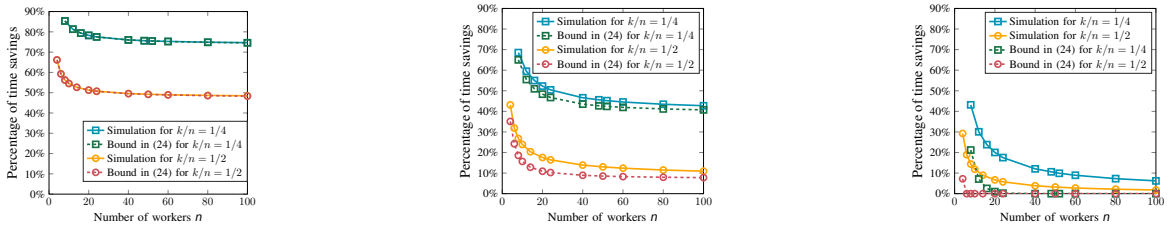
Claim 6. For each $k > 1$, we have

$$I_k \triangleq \int_{B_{k-1}} dF_{T^0}(y_{k-1})\ldots dF_{T^0}(y_1) = \int_0^{y_k}\cdots\int_0^{y_2}\prod_{i=1}^{k-1} dF_{T^0}(y_i) = \frac{F_{T^0}(y_k)^{k-1}}{(k-1)!}.$$

Since the projection of $\hat{A}(t)$ on $(n-k+1)$ dimensional space spanned by $(y_k,\ldots,y_n)$ is equal to $A(t)$, it follows that the integration of the first part is equal to $n!\int_{(y_k,\ldots,y_n)\in A(t)} dF_{T^0}(y_n)\ldots dF_{T^0}(y_k)$, giving us the result. ∎

VII. Interplay between code design and latency

Universal Staircase codes allows the master to decode Ax from any random number d of workers, $k \le d \le n$.



(a) Savings for $\lambda c = 100$. (b) Savings for $\lambda c = 1$. (c) Savings for $\lambda c = 0.001$.

Fig. 3: Savings for the fixed rate regime, $k/n = 1/2$ and $1/4$. The lower bound on the savings of Staircase codes obtained from (24) is compared to the numerical values obtained by simulations. We consider systems with no colluding workers, i.e., $z = 1$, we fix $\lambda = 1$ and vary c. For instance, for systems with rate $k/n = 1/2$ and $\lambda c = 100$ Staircase codes can provide up to 66% reduction in the mean waiting time.

The downside is that the universal construction requires a large number of sub-tasks $b = \text{LCM}(k \quad z+1, \ldots, n \quad z)$. In many applications, there may be an overhead associated with excessive divisions into sub-tasks. We show that we can reduce the number of sub-tasks at the expense of a small increase of the Master's waiting time. Using the so-called Δ-Universal Staircase codes [14] reduces the number of sub-tasks at the expense of limiting the Master to a set $\Delta \quad \{k, \ldots, n\}$ of number of workers allowing the Master to decode $A\mathbf{x}$. In other words, the Master can decode $A\mathbf{x}$ by downloading enough information from any d workers, $d \in \Delta$. The number of sub-tasks assigned to each worker is reduced from $b = \text{LCM}(k \quad z+1, \ldots, n \quad z)$ to the least common multiple of all $d_i \in \Delta$. It remains to prove that d is concentrated around its mean. Hence, restricting d to an interval Δ centered around its mean, leads to a reduction in the Master's waiting time.

Next, we prove that the number of workers d that minimize the waiting time is concentrated around its average.

Lemma 7. *For an (n, k, z) system, the probability distribution of the distance between d and its average is*

$$\Pr\{|d \quad \mathbb{E}[d]| > t\} \quad 2e^{\quad 2t^2/n(n \quad k)^2}.$$

We prove Lemma 7 by showing that the number of workers d that first finish the aggregate computation is concentrated around its mean, using McDiarmid's inequality. Recall that $d : \mathbb{R}_+^n \to \{k, \ldots, n\}$ is a function of the compute times $T_1, \ldots, T_n$.

$$d(T_1, \ldots, T_n) \quad , \quad \arg\min \quad \frac{k \quad z}{i \quad z} T_{(i)} : i \in \{k, \ldots, n\} \quad .$$

Claim 8. *The number of workers d that minimize the waiting time is a bounded difference function of compute times with constants $(n \quad k, \ldots, n \quad k)$. That is, for each $i \in [n]$ taking $t, t' \in \mathbb{R}_+^n$ such that $t_j = t'_j$ for each $j \in [n] \setminus \{i\}$ and $t_i \neq t'_i$,*

$$\sup\{|g(t) \quad g(t')| : t, t' \in \mathbb{R}_+^n\} \quad n \quad k. \qquad (23)$$

The claim follows from the fact that $d \in \{k, \ldots, n\}$. Therefore, we can apply the McDiarmid's inequality to obtain the concentration bound on d.

VIII. Simulations

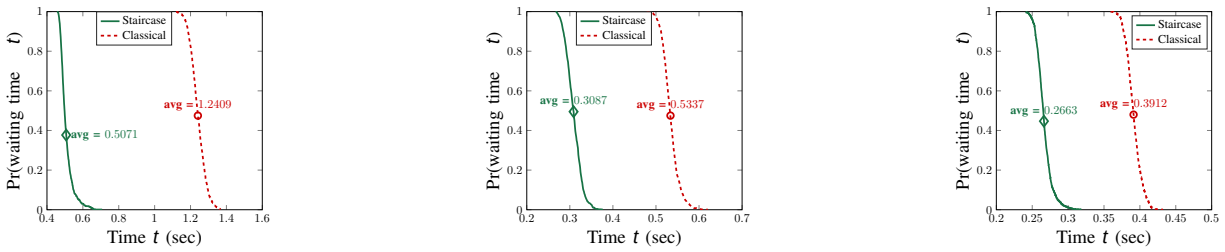
We use the normalized difference between the mean waiting time of Staircase codes and classical secret sharing codes as a performance metric for Staircase codes. We refer to this metric as the savings. Using the result of Theorem 1, we can get a lower and an upper bound on the savings brought by Staircase codes. The lower bound on the savings is given in (24).

$$1 \quad \frac{\mathbb{E}[T_{\text{SC}}]}{\mathbb{E}[T_{\text{SS}}]} \quad 1 \quad \min_{d \in \{k, \ldots, n\}} \quad \frac{(k \quad z)(\lambda c + H_n \quad H_{n \quad d})}{(d \quad z)(\lambda c + H_n \quad H_{n \quad k})} \quad . \qquad (24)$$

To get an idea of the actual savings and the tightness of the bound in (24), we ran numerical simulations of the mean waiting time induced by the use of Staircase codes. By looking at (24), we notice that the bound depends on λ and c only through[5] λc (our simulations show that the actual savings also have a strong dependency on λc). Therefore, we consider three cases for λc : large values of λc ($\lambda c = 100$), medium values of λc ($\lambda c = 1$) and small values of λc ($\lambda c = 0.001$). We ran the simulations for two regimes:

Fixed rate k/n: the plots can be seen in Figure 3. We deduce from the plots that the lower bound is tighter for large values of λc. Moreover, the savings increase with the decrease of the rate k/n and the increase of λc. Note that for large values of λc, the lower bound in (24) converges to $1 \quad k/n$.

[5]Note that for $c = 0$ we go to the exponential model and the savings would depend only on .



(a) $(n, k, z) = (4, 2, 1)$. (b) $(n, k, z) = (10, 5, 1)$. (c) $(n, k, z) = (20, 10, 1)$.

Fig. 4: Empirical complementary CDF of the Master's waiting time (and its average) observed on Amazon EC2 clusters for systems with rate $k/n = 1/2$. The data matrix A is a 378000 250 matrix with entries generated uniformly at random from $f1, \ldots, 255g$. Staircase codes bring 59% reduction in the mean waiting time for $n = 4$. Those numbers were obtained by repeating the multiplication process 1000 times.

Fixed number of parities $n \quad k$: We deduce from the plots that similarly to the fixed rate regime the lower bound is tight for large values of λc and that the savings increase with the increase of the number of parities $n \quad k$ and with the increase of λc. However, we observe that the savings vanish asymptotically with n in this regime. Due to space constraints, the plots are omitted and can be found in [2].

IX. Implementation and Validation of the Theoretical Model

We describe a representative sample of our implementation on Amazon EC2 clusters and discuss our observations. We present traces for systems with fixed rate $k/n = 1/2$ (Figure 4). We noticed that the straggler behavior, and therefore the savings, can depend on the date and time of the implementation. We refer interested readers to [2] where we highlight this dependence by presenting traces of one system implemented at different date and times.

Discussion on the theoretical model: Before giving the details, we summarize our findings. We observe that the savings of the system on EC2 can surpass the numerical values resulting from our theoretical model in Section II for large sizes of the matrix A. However, for small sizes of A, the savings in practice can be less.

The difference between the theoretical results and the implementations can be attributed to several reasons. First, in our model we assume in (2) that the total service time of a task does not change when divided into b sub-tasks, each requiring the same service time. Whereas, our implementation on Amazon shows that the download time decreases faster than linearly with the size of the sub-task for large sub-tasks. Second, for small sub-tasks, we noticed an additional overhead of sending the results of multiple sub-tasks. This overhead becomes non-negligible when the task is small. Third, we have assumed a homogeneous setting where all workers have the same behavior which is not always the case in practice.

Despite these differences, our adopted theoretical model is more amenable to theoretical analysis and provides insightful engineering guiding principles.

We present the implementation of $(4, 2, 1)$, $(10, 5, 1)$ and $(20, 10, 1)$ systems on Amazon EC2 clusters. We use M4.large EC2 instances [45] from Amazon web services (AWS) for our implementation. We assign the Master's job to an instance located in Virginia and the workers job to instances located in Ohio. We plot in Figures 4a, 4b and 4c the empirical complementary CDF of the Master's waiting time for Staircase codes and classical secret sharing codes for $(4, 2, 1)$, $(10, 5, 1)$ and $(20, 10, 1)$ systems, respectively. The average savings brought by Staircase codes are 59%, 42% and 32% for systems with $n = 4$, $n = 10$ and $n = 20$ workers, respectively. Note that for this set of implementation, the Master's data A is a matrix of size 378000 250 with entries generated uniformly at random from $f1, \ldots, 255g$. We run 1000 multiplications of A by a randomly generated vector $\mathbf{x}$. In the technical report [2] we present the trace of a $(4, 2, 1)$ system implemented at different dates and times on Amazon EC2 clusters.

X. Conclusion and Open Problems

We consider the problem of secure coded computing. We propose the use of a new family of secret sharing codes called Staircase codes that reduces the delays caused by stragglers. We show that Staircase codes always lead to smaller waiting time compared to classical secret sharing codes, e.g., Shamir secret sharing codes. The reason behind reducing the delays is that Staircase codes allow flexibility in the number of stragglers up to a given maximum, and universally achieve the information theoretic limit on the download cost by the Master,

leading to latency reduction. We consider the shifted exponential model for the workers's response time. In our analysis, we find upper and lower bounds on the Master's mean waiting time. We characterize the distribution of the Master's waiting time, and its mean, for systems with $n = k \quad 1$ and $n = k \quad 2$. Moreover, we derive an expression that can give the exact distribution, and the mean, of the waiting time of the Master. We supplement our theoretical study with extensive implementation on Amazon EC2 clusters.

While Staircase codes reduce the Master's waiting time by minimizing the download cost, they are not designed to minimize latency. The problem of designing codes that minimize the latency remains open in general. Another open problem, which we leave for future work, is when malicious workers corrupt the results sent to the Master.

REFERENCES

[1] R. Bitar, P. Parag, and S. El Rouayheb, "Minimizing latency for secure distributed computing," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 2900–2904, June 2017.

[2] R. Bitar, P. Parag, and S. El Rouayheb, "Minimizing latency for secure coded computing using secret sharing via staircase codes," *arXiv preprint*, vol. abs/1802.02640, 2018.

[3] https://setiathome.berkeley.edu.

[4] https://foldingathome.stanford.edu.

[5] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," *SIAM Journal on Computing*, vol. 43, no. 2, pp. 831–871, 2014.

[6] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Transactions on Information Theory*, vol. 64, no. 3, pp. 1514–1529, 2017.

[7] S. Dutta, V. Cadambe, and P. Grover, "Short-dot: Computing large linear transforms distributedly using coded short dot products," in 29*th Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 2092–2100, 2016.

[8] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[9] J. Dean and L. A. Barroso, "The tail at scale," *Communications of the ACM*, vol. 56, no. 2, pp. 74–80, 2013.

[10] M. J. Atallah and K. B. Frikken, "Securely outsourcing linear algebra computations," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '10, (New York, NY, USA), pp. 48–59, ACM, 2010.

[11] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[12] R. J. McEliece and D. V. Sarwate, "On sharing secrets and Reed-Solomon codes," *Communications of the ACM*, vol. 24, no. 9, pp. 583–584, 1981.

[13] R. Bitar and S. El Rouayheb, "Staircase codes for secret sharing with optimal communication and read overheads," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 1396–1400, July 2016.

[14] R. Bitar and S. El Rouayheb, "Staircase codes for secret sharing with optimal communication and read overheads," *IEEE Transactions on Information Theory*, vol. PP, no. 99, pp. 1–1, 2017.

[15] G. Liang and U. C. Kozat, "TOFEC: Achieving optimal throughput-delay trade-off of cloud storage using erasure codes," in *IEEE International Conference on Computer Communications*, 2014.

[16] W. Huang, M. Langberg, J. Kliewer, and J. Bruck, "Communication efficient secret sharing," *IEEE Transactions on Information Theory*, vol. 62, pp. 7195–7206, Dec 2016.

[17] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Straggler mitigation in distributed matrix multiplication: Fundamental limits and optimal coding," in *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 2022–2026, IEEE, 2018.

[18] Z. Jia and S. A. Jafar, "Cross subspace alignment codes for coded distributed batch matrix multiplication," *arXiv preprint arXiv:1909.13873*, 2019.

[19] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, Q. V. Le, *et al.*, "Large scale distributed deep networks," in *Advances in neural information processing systems*, pp. 1223–1231, 2012.

[20] J. Chen, R. Monga, S. Bengio, and R. Jozefowicz, "Revisiting distributed synchronous SGD," *arXiv preprint arXiv:1604.00981*, 2016.

[21] L. Huang, S. Pawar, H. Zhang, and K. Ramchandran, "Codes can reduce queueing delay in data centers," in *IEEE International Symposium on Information Theory (ISIT)*, 2012.

[22] G. Joshi, Y. Liu, and E. Soljanin, "Coding for fast content download," in *50th Annual Allerton Conference on Communication, Control, and Computing*, 2012.

[23] S. Kadhe, E. Soljanin, and A. Sprintson, "Analyzing the download time of availability codes," in *IEEE International Symposium on Information Theory (ISIT)*, 2015.

[24] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding," in 29*th Conference on Neural Information Processing Systems (NIPS)*, 2016.

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

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

[27] W. Halbawi, N. Azizan, F. Salehi, and B. Hassibi, "Improving distributed gradient descent using Reed-Solomon codes," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 2027–2031, 2018.

[28] S. Dutta, V. Cadambe, and P. Grover, "Coded convolution for parallel and distributed computing within a deadline," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 2403–2407, 2017.

[29] Y. Yang, P. Grover, and S. Kar, "Computing linear transformations with unreliable components," *IEEE Transactions on Information Theory*, 2017.

[30] R. Cramer, I. B. Damgård, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*. New York, NY, USA: Cambridge University Press, 1st ed., 2015.

[31] R. G. D'Oliveira, S. El Rouayheb, and D. Karpuk, "Gasp codes for secure distributed matrix multiplication," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 1107–1111, IEEE, 2019.

[32] W.-T. Chang and R. Tandon, "On the capacity of secure distributed matrix multiplication," in *IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, IEEE, 2018.

[33] J. Kakar, S. Ebadifar, and A. Sezgin, "Rate-efficiency and straggler-robustness through partition in distributed two-sided secure matrix computation," *arXiv preprint arXiv:1810.13006*, 2018.