Low latency replication coded storage over memory-constrained servers

Rooji Jinan* Ajay Badita* Pradeep Sarvepalli† Parimal Parag*

Abstract—**We consider a distributed storage system storing a single file, where the file is divided into equal sized fragments. The fragments are replicated with a common replication factor, and stored across servers with identical storage capacity. An incoming download request for this file is sent to all the servers, and it is considered serviced when all the unique fragments are downloaded. The download time for all fragments across all servers, is modeled as an independent and identically distributed (*i.i.d.*) random variable. The mean download time can be bounded in terms of the expected number of useful servers available after gathering each fragment. We find the mean number of useful servers after collecting each fragment, for a random storage scheme for replication codes. We show that the performance of the random storage for replication code achieves the upper bound for expected number of useful servers at every download asymptotically in number of servers for any storage capacity. Further, we show that the performance of this storage scheme is comparable to that of Maximum Distance Separable (MDS) coded storage.**

I. Introduction

One of the major challenges in the design of modern distributed systems is to ensure resilience against node failures and unpredictable download times [1]. Adding redundancy through error-correcting codes addresses these challenges along with offering fast access to the data due to parallelization gains. We study a redundant distributed storage system where a single file of unit size is stored over B servers that can each store α fraction of this file. We refer to this distributed storage system as an α-B system. Further, we suppose that single file of unit size is divided into V fragments, encoded into VR fragments, and stored over an α-B system. Such storage codes can be viewed as (n, k) error-correcting codes which encode $k = V$ information symbols into $n = VR$ encoded symbols. For example, in (VR, V) replication coding, we replicate each of the V file fragments R times, and the file can be decoded by downloading a single replica of each fragment.

Such storage systems where each of the B parallel servers stores a single coded fragment can be found in literature [1]–[3]. For this case it was shown that MDS codes provide optimal performance [3] in terms of the mean access delay. However, if we allow for an additional degree of freedom, namely division of a file into larger number of fragments with smaller size, then even non-MDS codes can become competitive. Consider the staircase codes proposed in [4] for secure computation which can be viewed as a (BK, V) q-ary code, which is non MDS. When the number of fragments stored per server $\alpha V > 1$, staircase codes show improvement over MDS codes with single coded fragment stored at each server. The mean download time of staircase codes was shown to be smaller than that of a $(B, B/R)$ MDS code in [5], [6].

Dividing the file into V fragments, with the code rate $1/R$ and the number of servers at B, it can be shown that among all (VR, V) codes stored over an α-B system, an MDS code has the smallest file download time for a class of download time distributions. However, one of the major drawbacks of MDS codes is its high decoding complexity. Even the best known MDS decoding algorithms are polynomial in the number of coded symbols VR [7]. Further, in storage systems where file sizes often change with frequent writes [8], the entire file has to be encoded again [9]. In addition, to be able to code V fragments of a file into VR MDS coded symbols, the symbols must belong to a sufficiently large alphabet [10, Theorem 4.1] which requires the fragment to be large.

Block replication codes score well on all of these fronts. We show the existence of a good storage scheme for (VR, V) replication code on an α-B system, by studying a random storage scheme that offers mean download time comparable to that of MDS codes. To this end, we modify the system model slightly where each server can store all VR fragments. Then using concentration arguments, we show that the fraction of file stored per server converges to the storage constraint α asymptotically in number of fragments, for the proposed randomized storage scheme.

A. Related work

Coding techniques have been used to ensure reliability in distributed storage systems with fault-prone network [1], [11], [12]. Another objective of interest to us is achieving low latency using storage codes [13], [14]. In this work, we study replication codes where files are stored redundantly over the system. Trade-off between latency and server utilization cost of availing redundancy was studied in [14]–[19]. Two well studied file encoding strategies used in distributed systems with redundant storage are MDS coding [3], [11], [14], [18], [20] and replication [19], [21]. It has been shown that MDS coding outperforms replication in mean file access latency [3], [22]. Further, [23] studies certain conditions under which MDS coding is preferred over replication and vice versa. Our work differs from these works as we allow larger fragmentation of the file such that each server can store more than one fragment,

The authors* are with Indian Institute of Science, Bangalore, Karnataka 560012, India. Email:*{roojijinan, ajaybadita, parimal}@iisc.ac.in.

Author† is with Indian Institute of Technology Madras, Chennai, Tamil Nadu 600036, India. Email:†{pradeep}@ee.iitm.ac.in.

This work was supported in part by the Department of Telecommunications, Government of India, under Grant DOTC-0001, the RBCCPS, and the Centre for Networked Intelligence (a Cisco CSR initiative) of the IISc.

2021 IEEE International Symposium on Information Theory (ISIT) | 978-1-5386-8209-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/ISIT45174.2021.9517901

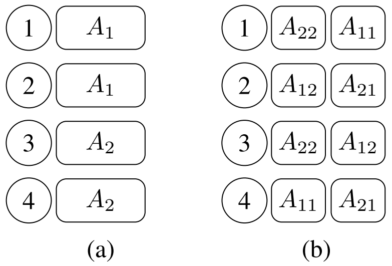

Fig. 1. Two storage policies indicating different fragmentation of a single file with a fixed server storage of half the file size for all four servers. In the scheme (a), the file is divided into two equal sized fragments A_1, A_2. In scheme (b), the file is divided into four equal sized fragments $A_{11}, A_{12}, A_{21}, A_{22}$.

while keeping the number of servers and storage per server unchanged as illustrated in Figure 1.

B. Main contributions

We study distributed storage of replication coded file fragments over α-B systems with each server storing multiple fragments. We list our main contributions below.

- We find a lower bound for the mean download time of a file in terms of the expected sum of number of useful servers for each fragment download when fragment download times are random and *i.i.d.* exponential.
- We provide an upper bound on the number of useful servers for any fragment storage scheme in an α-B system employing (VR, V) replication code.
- We show that among all the (VR, V) codes stored on an α-B system, an MDS code meets the upper bound on the number of useful servers.
- We propose a randomized placement for replication coded fragments on a storage system where each server can store VR fragments and show that it meets the upper bound on the number of useful servers asymptotically.
- We also show that for large number of fragments, the modified system under the randomized placement converges to α-B system.
- Finally, we demonstrate through numerical studies that the bound on the number of useful servers is a good indicator of the performance of the storage schemes with respect to their average download times.

II. System model

Let us suppose that a single file is fragmented into V pieces and stored on an α-B system with storage capacity per server $\alpha \triangleq \frac{K}{V} < 1$. That is, each of the B servers is assumed to have an identical storage capacity of K fragments of size $1/V$ of the file and hence can store α fraction of the file. We study (VR, V) replication codes with *replication factor*, R. That is, each file fragment is replicated R times. Note that, the system is assumed to have sufficient storage capacity to store all VR fragments, and this requires that $VR \leqslant KB$. We assume a single request in the system which is forked to all B servers. At each server $b \in [B]$, the request starts downloading stored fragments in succession.

A. Storage model

For a specific storage code, we refer to the collection of fragments stored on each server as the *storage scheme*. The storage scheme must be designed with a view to facilitate fast download of the file. The set of servers on which a fragment $v \in [V]$ is replicated, is denoted by $\Phi_v \subseteq [B]$ and called *occupancy set*. The storage scheme is completely determined by the collection of occupancy sets $\Phi \triangleq (\Phi_v : v \in [V])$. We also denote the set of fragments stored on a server $b \in [B]$ by the *fragment set* $S_b \subseteq [V]$. We further note that $S_b \triangleq \{v \in [V] : b \in \Phi_v\}$.

Definition 1. *For any (VR, V) replication code stored on B servers with storage capacity of K fragments, the α-(V, R)* replication storage ensemble *is defined as the collection*

$$\mathcal{S} \triangleq \left\{ \Phi \in (2^{[B]})^{[V]} \middle| |S_b| = K \text{ for all } b, VR = BK \right\}. \quad (1)$$

We classify α-(V, R) replication storage scheme as completely utilizing and underutilizing. An underutilizing α-(V, R) replication storage scheme allows multiple copies of the same fragment to be stored on the same server, i.e., $|\Phi_v| < R$. Such a scheme do not take full advantage of the possible parallelization. If each fragment is stored on R distinct servers, then the α-(V, R) replication storage scheme satisfies $|\Phi_v| = R$ for all fragments v and is completely utilizing.

B. Fragment download time model

The fragment download time at each server is modeled by a random variable that captures uncertainty due to network delays and server background processes [24]. We denote the fragment download time for fragment v at the server b by a nonnegative random variable T_{bv}. We assume that the marginal distribution of T_{bv} is identical for all fragments $v \in S_b$ at all servers $b \in [B]$. Motivated by analytical tractability, we further assume that the fragment download times T_{bv} are independent and exponentially distributed with rate μ. Thus, we have assumed fragment download times to be *i.i.d.*, which is a popular assumption in the literature [16], [21], [25] and the common distribution function F is given by $F(x) \triangleq P(\{T_{bv} \leqslant x\}) = 1 - e^{-\mu x}$, for all $x \geqslant 0$.

C. Download sequence

A request is forked in to all B servers and is considered to be serviced upon the download of all V distinct fragments. We assume that the request will not attempt to download a replica of an already downloaded fragment, and such download sequences are referred to as *minimal downloading sequences*. A minimal downloading sequence for replication codes consists of unique fragments. The ℓth downloaded fragment is denoted by v_ℓ, and the sequence of downloaded fragments until ℓth download is called download subsequence and denoted by $I_\ell \triangleq (v_1, \dots, v_\ell)$. We further assume that after a fragment download, the request immediately stops downloading the same fragment from other servers. Such an idealized model of cancellation at parallel servers serving same request has been adopted for analytical tractability and can be found to be used widely in literature [14], [15], [19], [21], [26], [27].

D. Problem formulation

Our main goal is to minimize the mean download time for a file stored using a storage scheme from the α-(V,R) replication storage ensemble, when the service times at each server are independent and exponentially distributed with rate μ. We denote the download time of ℓth distinct fragment v_ℓ by D_ℓ, where $D_0 \triangleq 0$ and $\ell \in \{0,\dots,V\}$. This indicates that a download subsequence I_ℓ of ℓ fragments has been downloaded at time D_ℓ. As the file is completely retrieved once V unique fragments are downloaded, the file download time is given by D_V. Any server that has fragments not yet downloaded by the request, is called a *useful server* [3]. The set of useful servers after ℓth download is given by $U(I_\ell) = \bigcup_{v\notin I_\ell} \Phi_v$ and let its cardinality be $N(I_\ell)$. The request is being served by $N(I_\ell)$ parallel servers in the duration $[D_\ell, D_{\ell+1})$. From the independent and memoryless service assumption at all servers of rate μ, we have $\mathbb{E}\left[D_{\ell+1} - D_\ell \middle| I_\ell\right] = \frac{1}{N(I_\ell)\mu}$. The download time for V fragments can be written as the sum of download time of individual fragments, i.e. $D_V = \sum_{\ell=0}^{V-1}(D_{\ell+1}-D_\ell)$. From the linearity and the tower property of expectation, it follows that the mean download time averaged over all fragments is

$$\frac{1}{V}\mathbb{E}\left[D_V\right] = \frac{1}{V}\mathbb{E}\left[\sum_{\ell=0}^{V-1}\frac{1}{N(I_\ell)\mu}\right]. \tag{2}$$

We see that the mean download time depends on $U(I_\ell)$, the set of useful servers remaining after ℓth download, which in turn depends on the storage scheme Φ. Next, we provide a lower bound on the mean file download time in terms of the sum of mean number of useful servers.

Lemma 2. *The mean download time can be lower bounded as* $\frac{1}{V}\mathbb{E}\left[D_V\right] \geqslant \frac{V}{\mu\sum_{\ell=0}^{V-1}\mathbb{E}[N(I_\ell)]}$.

(For proof, see [28, Lemma 5].) In some important settings, the sum $\sum_\ell \mathbb{E}\left[N(I_\ell)\right]$ is analytically more tractable when compared to $\sum_{\ell=0}^{V-1}\mathbb{E}\left[1/N(I_\ell)\right]$. Motivated by this fact, instead of minimizing the mean download time in (2), we study the following problem.

Problem 1. *Find a storage scheme Φ in an α-(V,R) replication storage ensemble that maximizes the mean number of useful servers averaged over all fragments, i.e.*

$$\Phi^* = \arg\max_{\Phi}\frac{1}{V}\sum_{\ell=0}^{V-1}\mathbb{E}\left[N(I_\ell)\right].$$

That is, we must find an optimal storage scheme that maximizes the normalized mean number of useful servers which is equivalent to minimizing the lower bound on mean download time. While Problem 1 does not optimize the mean download time, we empirically show that the schemes that maximize the mean number of useful servers has lower mean download times. Thus, the mean number of useful servers can be used as a good yardstick to measure the performance of an α-(V,R) replication storage scheme.

III. Analysis

In this section, we show the existence of a good α-(V,R) storage scheme. We first provide a simple upper bound on the number of useful servers for any α-(V,R) storage scheme.

Theorem 3. *For an α-(V,R) replication storage scheme $\Phi \in \mathcal{S}$ defined in* (1)*, the number of useful servers $N(I_\ell)$ after ℓ downloads is upper bounded in terms of $m \triangleq \lceil B/R\rceil$, as*

$$N(I_\ell) \leqslant B\mathbb{1}_{\{\ell\leqslant V-m\}} + (V-\ell)R\mathbb{1}_{\{\ell>V-m\}}. \tag{3}$$

(For proof, see [28, Theorem 6].)

Remark 4. *Recall that for an α-(V,R) replication storage scheme $B/R = V/K = 1/\alpha$. When $m = B/R$ is an integer, summing up both sides of* (3)*, and dividing both sums by the product BV, we obtain*

$$\frac{1}{BV}\sum_{\ell=0}^{V-1}N(I_\ell) \leqslant 1 - \frac{(m+1)}{2V}. \tag{4}$$

This gives us an upper bound on the normalized sum of number of useful servers.

Computation of the mean download time of a (VR,V) coded storage on an α-B system is challenging due to the combinatorial nature of the problem. To show existence of a good storage, we use randomized techniques. Specifically, we study a random storage scheme that offers mean download time comparable to that of MDS codes, for a large number of fragments. For analytical tractability, we modify the system model slightly where each server can store all VR fragments. Then using concentration arguments, we show in Theorem 8 that the fraction of file stored per server converges almost surely to the storage constraint α asymptotically in the number of fragments, for the proposed randomized storage scheme.

A. Randomized replication coded storage

The storage scheme for a (VR,V) replication code on the modified B server system is referred to as (B,V,R) *replication storage scheme* where each server has the capacity to store all the VR fragments.

Definition 5. *A (B,V,R) replication storage scheme, where rth replica of fragment v is stored on server $\Theta_{v,r} \in [B]$, chosen independently and uniformly at random from B servers, is called a* randomized (B,V,R) replication storage scheme*. The collection of all random (B,V,R) replication storage schemes is referred to as the* random (B,V,R) replication storage ensemble.

Thus, we observe that a randomized (B,V,R) replication storage scheme can be determined by *i.i.d.* random vectors $(\Theta_v : v \in [V])$, where $\Theta_v = (\Theta_{v,1},\dots,\Theta_{v,R})$ for each fragment v, and $P\left\{\Theta_{v,r} = b\right\} = \frac{1}{B}$, for all servers $b \in [B]$. Note that the random vector Θ_v is not a set but a vector, since more than one replica of a file fragment can be stored on a single server. For each fragment v, we can compute the number of replicas of this fragment stored at server b as $\beta_{vb} \triangleq \sum_{r\in[R]}\mathbb{1}_{\{\Theta_{v,r}=b\}}$.

Lemma 6. *For the random (B, V, R) replication storage scheme defined in Definition 5, we have $P(\cup_{b\in[B]}\{\beta_{vb} \geqslant 2\}) \geqslant 1 - e^{\frac{-\alpha(R-1)}{2}}, \quad v \in [V]$.*

(For proof, see [28, Lemma 18].) That is, there is a finite probability with which each file fragment repeats on a server. However, we can construct an occupancy set Φ_v for each fragment v from this vector Θ_v, by throwing away the repeated entries. Then, $\Phi_v = \{b \in [B] : \Theta_{v,r} = b \text{ for some } r \in [R]\}$. This implies that $|\Phi_v| \leqslant R$, and this inequality is strict if any entry in the vector Θ_v is repeated twice.

We will consider a family of (B, V, R) random replication storage schemes for increasing values of number of fragments V, while keeping the ratio $\alpha = \frac{R}{B}$ constant for the system. In this case, we will show that the fraction of file fragments stored by each server converges to α for the proposed random (B, V, R) replication storage scheme. The normalized number of fragments stored at any server $b \in [B]$ is defined as $\alpha_b^{\text{rep}} \triangleq \frac{1}{V}\sum_{v\in[V]}\sum_{r\in[R]}\mathbb{1}_{\{\Theta_{v,r}=b\}}$.

If the normalized number of fragments $\alpha_b^{\text{rep}} = \alpha$, the storage capacity per server in an α-(V, R) storage scheme, then the randomized (B, V, R) replication storage scheme defined in Definition 5 in terms of *i.i.d.* vectors $(\Theta_v : v \in [V])^1$, is an underutilizing α-(V, R) storage scheme with high probability, since there exists servers storing redundant replicas of the same fragment with high probability.

Definition 7. *A randomized (B, V, R) replication storage scheme is said to be an α-(V, R) storage scheme asymptotically in V, if for each server b, $\lim_{V\to\infty}\alpha_b^{\text{rep}} = \alpha$ almost surely.*

Theorem 8. *The randomized (B, V, R) storage scheme defined in Definition 5 is an α-(V, R) storage scheme asymptotically in V.*

(For proof, see [28, Theorem 20].) We can compute the sum of mean number of useful servers aggregated over all downloads, when the mean is taken over ensemble of random (B, V, R) replication storage schemes.

Theorem 9. *For the random (B, V, R) replication storage ensemble defined in Definition 5, we can write*

$$\frac{1}{BV}\sum_{\ell=0}^{V-1}\mathbb{E}\left[N(I_\ell)\right] = 1 - \frac{\left(1-\frac{1}{B}\right)\left(1-(1-\frac{1}{B})^{RV}\right)}{V\left(1-(1-\frac{1}{B})^{R}\right)}. \quad (5)$$

(For proof, refer [28, Theorem 21].)

Corollary 10. *A random (B, V, R) replication storage scheme is asymptotically optimal solution to the Problem 1 almost surely.*

Proof: For large number of fragments, we observe the following. By Theorem 8, proposed random (B, V, R) replication storage scheme almost surely converges to an α-(V, R) replication storage scheme and from Theorem 9, we obtain $\lim_{V\to\infty}\frac{1}{BV}\sum_{\ell=0}^{V-1}\mathbb{E}\left[N(I_\ell)\right] = 1$ for this scheme. From (4), we observe that the ensemble mean of normalized number of useful servers for the proposed random (B, V, R) replication storage scheme meets the upper bound for any α-(V, R) replication storage scheme, asymptotically in V. ■

^{1}Even though $\Theta : [V] \to [B]^R$ is a collection of *i.i.d.* random variables, we observe that normalized number of pieces on each server α_b^{rep} are dependent random variables. To see this, we observe that $P\{\Theta_{v,r} = b, \Theta_{v,r} = a\} = 0 \neq \frac{1}{B^2}$ for any $b \neq a$. It follows that $\mathbb{E}\left[\alpha_b^{\text{rep}}\alpha_a^{\text{rep}}\right] = \alpha^2 - \frac{1}{B}\alpha \neq \alpha^2 = \mathbb{E}\left[\alpha_b^{\text{rep}}\right]\mathbb{E}\left[\alpha_a^{\text{rep}}\right]$.

B. Comparison with MDS codes

In this section we consider storage schemes based on (VR, V) MDS codes assuming that the field is large enough so that a (VR, V) code exists. As mentioned earlier, MDS codes are known to outperform replication codes in terms of code rates and latency [3]. Here, we show that among all α-(V, R) coded storage schemes, the ones based on MDS codes minimize the mean download time. Further, we show that replication coded storage is asymptotically optimal.

Definition 11. *Consider a file with V fragments encoded to VR coded fragments, and completely utilizing storage of this (VR, V) code on a α-B system. Such storage schemes are referred to as* α-(V, R) coded storage schemes, *where the normalized storage capacity per server is $\alpha = R/B = K/V$ and the code rate is $1/R$.*

Remark 12. *For any α-(V, R) coded storage scheme, the number of useful servers $N(I_\ell)$ after ℓ downloads is always upper bounded by the total number of servers B, and hence $\frac{1}{BV}\sum_{\ell=0}^{V-1}N(I_\ell) \leqslant 1$.*

Definition 13. *Any V subset of VR coded fragments that suffices to decode a (VR, V) code, i.e., reconstruct the V uncoded fragments, is called an* information set *[3], [29]. For an α-(V, R) coded storage scheme, we can define the collection of all information sets [3, Section II], as $\mathcal{I}$.*

For a completely utilizing α-(V, R) replication storage scheme, information sets consist of distinct V fragments. For α-(VR, V) MDS coded storage, information sets are any V coded fragments, and hence $\mathcal{I} = \{S \subset [VR] : |S| = V\}$. This implies that the collection of information sets for MDS code includes collection of information sets for any other (VR, V) code. Then, the largest possible set of useful servers among all α-(V, R) coded storage schemes is the one achieved by MDS coded storage for the same download sequence I_ℓ. Using coupling arguments, we can establish the following result.

Theorem 14. *Among all α-(V, R) coded storage schemes, MDS codes minimize the mean download time.*

(For proof, see [28, Theorem 25].)

C. Asymptotic optimality of replication codes

We next find bounds on the number of useful servers for MDS coded storage, which can be used as a benchmark to compare replication coded storage.

Lemma 15. *For a completely utilizing α-(V, R) MDS storage scheme, the number of useful servers $N^{\text{mds}}(I_\ell)$ is bounded as $B - \left\lfloor\frac{\ell}{K}\right\rfloor \leqslant N^{\text{mds}}(I_\ell) \leqslant \min(B, VR - \ell)$.*

(For proof, see [28, Lemma 26].) The previous lemma will immediately give us the following result by taking average over all V fragments.

Corollary 16. *For a completely utilizing α-(V,R) MDS coded storage scheme with code rate $\frac{1}{R} \leqslant \frac{V}{B+V}$, the normalized aggregate number of useful servers is bounded as*

$$1 - \frac{1}{2R}(1-\frac{1}{V}) \leqslant \frac{1}{BV}\sum_{\ell=0}^{V-1} N^{\mathrm{mds}}(I_\ell) \leqslant 1. \qquad (6)$$

Remark 17. *Theorem 8 says that a random (B,V,R) replication storage scheme achieves a storage fraction $\alpha = R/B$ on each server, as the number of fragments V becomes large. Further, from proof of Corollary 10 we observe that the limit of average number of useful servers for a typical random (B,V,R) replication storage scheme meets the upper bound for MDS codes in (6), as the number of fragment grows. This implies that replication coded storage is asymptotically optimal.*

Optimality of replication codes for large storage: So far, we have considered the case $\alpha \leqslant 1$. For completeness we show that, when $K \geqslant V$, then there exists a (VR,V) replication code such that the number of useful servers remains B after every download.

Lemma 18. *There exists a (VR,V) replication code, that meets the universal upper bound on average number of useful servers for any completely utilizing α-(V,R) coded storage scheme when $\alpha \geqslant 1$.*

(For proof, refer [28, Lemma 28].) Thus, the average number of useful servers for replication code meets the universal upper bound for all fragment size V, when the servers can store the entire file.

D. Numerical results

We report the numerical results for a random (B,V,R) replication storage system with a fixed fraction $\alpha = \frac{R}{B} = 0.25$. In Fig. 2, we plot the normalized mean number of useful servers (5) for the randomized replication storage along with the corresponding upper bound (3), for increasing number of fragments V and a fixed fraction $\alpha = 0.25$. The mean number of useful servers have been normalized with respect to the number of servers B for the ease of comparison. From this figure, we observe that the mean number of useful servers for the randomized storage of replicated file fragments approaches the upper bound as the number of fragments V grows large. We conducted empirical studies by simulating a random (B,V,R) replication storage system for $B = V$, and fixed fraction $\alpha = 0.25$. In Fig. 3, we plot the empirical normalized average of number of useful servers as a function of the fraction of downloaded fragments $\frac{\ell}{V}$ for different number of fragments V. For each V, we also tabulated the average file download time in Table I. From this table and the Fig. 3, we observe that the (B,V,R) storage codes with larger V has uniformly higher average number of useful servers, and smaller average download time.

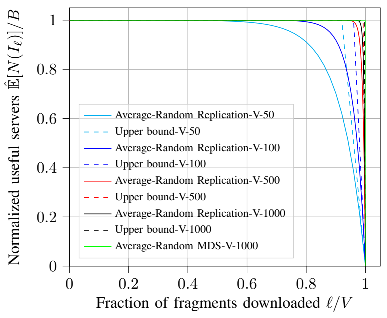

Fig. 2. Plot of the normalized mean number of useful servers given in (5) for random replication scheme, and the corresponding upper bound given in (3), as the number of fragments V increases in the set $\{50, 100, 500, 1000\}$ for $\alpha = R/B = 0.25$.

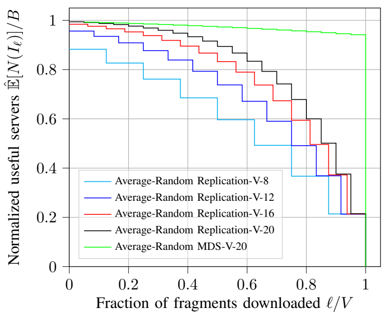

Fig. 3. Plot of the normalized empirical average number of useful servers for random (B,V,R) replication scheme for $B = V$, $\alpha = R/B = 0.25$, as the number of fragments V increases in the set $\{8, 12, 16, 20\}$.

TABLE I
AVERAGE DOWNLOAD TIMES OF RANDOM (B,V,R) REPLICATION STORAGE

Random storage code			**Average download time**
B	V	R	
8	8	2	26936.3
12	12	3	14922.4
16	16	4	9915.83
20	20	5	7324.77

IV. Conclusion

We investigated replication codes for latency minimization in a distributed storage system with storage constraints. We characterized the system performance in terms of the mean number of useful servers, when the service distribution is exponential. An important outcome of our investigations is that replication codes combined with probabilisitic placement and a larger fragmentation can lead to performances close to the proposed lower bounds. In fact, our numerical studies show that the average performance of random replication codes is very competitive with MDS codes while also affording simple encoding and decoding complexity. This motivates us to further study replication based schemes with deterministic placement strategies, their performance and other aspects of their implementation. Other directions of research are upper bounds on the mean download times and study of systems with nonexponential fragment download times.

References

[1] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. O. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, Aug. 2010.

[2] Y. Wu, A. G. Dimakis, and K. Ramchandran, "Deterministic regenerating codes for distributed storage," in *Allerton Conf. Commun. Ctrl Comput.*, Sep. 2007, pp. 1–5.

[3] A. Badita, P. Parag, and J.-F. Chamberland, "Latency analysis for distributed coded storage systems," *IEEE Trans. Inf. Theory*, vol. 65, no. 8, pp. 4683–4698, Aug. 2019.

[4] R. Bitar and S. E. Rouayheb, "Staircase codes for secret sharing with optimal communication and read overheads," *IEEE Trans. Inf. Theory*, vol. 64, no. 2, pp. 933–943, Feb. 2018.

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

[6] ——, "Minimizing latency for secure coded computing using secret sharing via staircase codes," *IEEE Trans. Commun.*, Apr. 2020.

[7] S. Lin and D. J. Costello, *Error control coding*. Prentice hall, 2001, vol. 2, no. 4.

[8] K. Ren, Y. Kwon, M. Balazinska, and B. Howe, "Hadoop's adolescence: An analysis of hadoop usage in scientific workloads," *Proc. VLDB Endow.*, vol. 6, no. 10, p. 853–864, Aug. 2013.

[9] F. Maturana, V. S. C. Mukka, and K. V. Rashmi, "Access-optimal linear MDS convertible codes for all parameters," in *IEEE Inter. Symp. Info. Theory (ISIT)*, 2020.

[10] R. M. Roth, *Introduction to coding theory*. Cambridge University Press, 2006.

[11] C. Suh and K. Ramchandran, "Exact-repair MDS codes for distributed storage using interference alignment," in *IEEE Inter. Symp. Info. Theory (ISIT)*, Jun. 2010, pp. 161–165.

[12] A. S. Rawat, D. S. Papailiopoulos, A. G. Dimakis, and S. Vishwanath, "Locality and availability in distributed storage," *IEEE Trans. Inf. Theory*, vol. 62, no. 8, pp. 4481–4493, Feb. 2016.

[13] J. Dean and L. A. Barroso, "The tail at scale," *Communications of the ACM*, vol. 56, pp. 74–80, Feb. 2013.

[14] G. Joshi, Y. Liu, and E. Soljanin, "On the delay-storage trade-off in content download from coded distributed storage systems," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 989–997, May 2014.

[15] G. Joshi, E. Soljanin, and G. W. Wornell, "Queues with redundancy: Latency-cost analysis," *SIGMETRICS Perform. Eval. Rev.*, vol. 43, no. 2, pp. 54–56, Sep. 2015.

[16] Y. Xiang, T. Lan, V. Aggarwal, and Y.-F. R. Chen, "Joint latency and cost optimization for erasure-coded data center storage," *IEEE/ACM Trans. Netw.*, vol. 24, no. 4, pp. 2443–2457, Sep. 2016.

[17] P. Parag, A. Bura, and J.-F. Chamberland, "Latency analysis for distributed storage," in *IEEE Inter. Conf. Comp. Commun. (INFOCOM)*, May 2017, pp. 1–9.

[18] A. Badita, P. Parag, and V. Aggarwal, "Sequential addition of coded tasks for straggler mitigation," in *IEEE Inter. Conf. Comp. Commun. (INFOCOM)*, Jul. 2020.

[19] ——, "Optimal server selection for straggler mitigation," *IEEE/ACM Trans. Netw.*, vol. 28, no. 2, pp. 709–721, 2020.

[20] ——, "Single-forking of coded subtasks for straggler mitigation," *IEEE/ACM Trans. Netw.*, 2021.

[21] D. Wang, G. Joshi, and G. Wornell, "Using straggler replication to reduce latency in large-scale parallel computing," *SIGMETRICS Perform. Eval. Rev.*, vol. 43, no. 3, pp. 7–11, Dec. 2015.

[22] B. Li, A. Ramamoorthy, and R. Srikant, "Mean-field-analysis of coding versus replication in cloud storage systems," in *IEEE Inter. Conf. Comp. Commun. (INFOCOM)*, Jul. 2016, pp. 1–9.

[23] P. Peng, E. Soljanin, and P. Whiting, "Diversity/parallelism trade-off in distributed systems with redundancy," *arXiv preprint arXiv:2010.02147*, 2020.

[24] D. Cheng, J. Rao, Y. Guo, and X. Zhou, "Improving mapreduce performance in heterogeneous environments with adaptive task tuning," in *Int. Middle. Conf. (Middleware)*. Bordeaux, France: ACM, 2014.

[25] N. B. Shah, K. Lee, and K. Ramchandran, "When do redundant requests reduce latency?" *IEEE Trans. Commun.*, vol. 64, no. 2, pp. 715–722, Dec. 2016.

[26] D. Wang, G. Joshi, and G. Wornell, "Efficient task replication for fast response times in parallel computation," *SIGMETRICS Perform. Eval. Rev.*, vol. 42, no. 1, pp. 599–600, Jun. 2014.

[27] H. Wang, J. Li, Z. Shen, and Y. Zhou, "Approximations and bounds for (n, k) fork-join queues: a linear transformation approach," in *2018 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, May 2018, pp. 422–431.

[28] R. Jinan, A. Badita, P. Sarvepalli, and P. Parag, "Latency optimal storage and scheduling of replicated fragments for memory-constrained servers," *arXiv 2010.01589*, 2020.

[29] P. Gopalan, G. Hu, S. Saraf, C. Wang, and S. Yekhanin, "Maximally recoverable codes for grid-like topologies," in *ACM-SIAM Symp. Disc. Algor. (SODA)*, May 2016, pp. 2092–2108.