MDS coding is better than replication for job completion times

Citation for published version:
Duffy, KR & Shneer, V 2021, 'MDS coding is better than replication for job completion times', *Operations Research Letters*, vol. 49, no. 1, pp. 91-95. https://doi.org/10.1016/j.orl.2020.11.010

Digital Object Identifier (DOI):
10.1016/j.orl.2020.11.010

Published In:
Operations Research Letters

MDS coding is better than replication for job completion times

Ken R. Duffy[a], Seva Shneer[b]

[a] *Hamilton Institute, Maynooth University, Maynooth, Ireland*
[b] *School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, UK*

Abstract

If queue-states are unknown to a dispatcher, a replication strategy has been proposed where job copies are sent randomly to servers. When the service of any copy completes, redundant copies are removed. For jobs that can be algebraically manipulated and linear servers, to get the response-time performance of sending d copies of n jobs via the replication strategy, with Maximum Distance Separable (MDS) codes we show that $n+d$ jobs suffice. That is, while replication is multiplicative, MDS is linear.

1. Introduction

It is well-known that if a job arrives to a system with many servers, its delay is minimized by joining the queue with the least waiting time. If there are large numbers of servers, the state of each of their queues may not be accessible at each job's arrival time. The celebrated power-of-d choices result (see [1], [2], [3]) establishes that by sampling a relatively small number of queues at random and joining the shortest of those, performance is asymptotically much better than in the case of random assignment. Many other load-balancing schemes have been proposed and investigated, see [4] for a current survey.

An interesting variant of this system has recently been introduced where the waiting times at any of the queues may not be available to an arriving job, see [5, 6, 7] and references therein. In that setting, as opposed to sampling d queues and joining the shortest waiting time, instead a **replication-d** strategy is proposed: for each job that arrives, a copy of it is placed in d distinct queues whose lengths and waiting times are unknown. The job's work is complete when the first of its replicas exits its service, and the remaining $d-1$ duplicate jobs are then removed from the system. For a heuristic illustration of the performance gain that is obtained from this strategy, assume that a system is in stationarity where any job sent to any server experiences a sojourn time, say W, comprised of the waiting time in the queue and the service time once it reaches the server, with distribution $\mathbb{P}(W \leq t) = F_W(t)$. Assuming queues are independent and identically distributed, a single job arrives and is subject to replication-d. With R denoting the job's completition time, its distribution is given by the minimum of d independent sojourn times

$$\begin{aligned}\mathbb{P}(R^{1,d} > t) &= \mathbb{P}(\min(W_1, \ldots, W_d) > t)\\ &= (1 - F_W(t))^d.\end{aligned}$$

With the system in stationarity and a batch of n jobs arriving where each job is subject to replication-d, the tail of the batch completion time distribution is governed by last job completion time in the batch and satisfies

$$\begin{aligned}\mathbb{P}(R^{n,d} > t) &= \mathbb{P}(\max(R_1^{1,d}, \ldots, R_n^{1,d}) > t)\\ &= 1 - \left(1 - (1 - F_W(t))^d\right)^n\\ &\sim n(1 - F_W(t))^d, \qquad (1)\end{aligned}$$

for large t. Thus, through the use of the replication-d strategy, the tail of the completion time distribution of the batch of n jobs is exponentially curtailed from approximately $n(1-F_W(t))$. That tail reduction is significant, and greatly reduces the straggler problem where the system is held up waiting for one job to be served before it can proceed to the next task.

In the present paper we consider the performance of an alternative approach that is available when the jobs to be served can be subject to algebraic manipulation and the output of the servers is a linear function of their input. This is the case, for example, if the jobs consist of digital packets that are traversing a network where the output of a server is its input and for large matrix multiplication tasks for Machine Learning. In the network setting, the replication-d strategy is similar in spirit to repetition coding [8], which is known to be suboptimal, and instead we consider a **MDS** (Maximum Distance Separable) approach. The benefits of MDS codes for making communications robust in networks subject to packet erasures are well-established. For information retrieval from a multi-server storage system, the improvements in response time that is attainable through the use of coding have been studied [9, 10, 11, 12, 13]. Both replication and MDS coding have also been proposed recently to resolve the stragglers problem in distributed gradient descent for Machine Learning [5, 14, 15, 16, 17, 18, 19, 20]. To the best of our knowledge, however, this is one of the first times its utility in reducing queueing delay in a feed-forward system has been shown.

Consider a batch arrival of n jobs, $J_1, \dots, J_n$, each of which consists of data of fixed size whose symbols take values in the reals or a large Galois field. The principle of the MDS coding approach is that rather than send duplicate jobs, one instead creates $n+m$ linear combinations of the form

$$K_j = A_1^{(j)} J_1 + \cdots + A_n^{(j)} J_n, \quad j \in \{1, \dots, n+m\},$$

where the $A_i^{(j)}$ are chosen in the reals or a finite fields. The idea underlying MDS codes is to consider each coded job, K_j, as a random linear equation such that the reciept of any linear function of any n of the $n+m$ linear combinations allows recovery of the processing of the original n jobs by Gaussian elimination. Reed-Solomon codes, for example, [21] are MDS codes. More generally, a Random Linear Code, where the coefficients are chosen uniformly at random, is an MDS code with high probability for a sufficiently large field size, e.g. [22].

When MDS is employed, the completion time of a batch is equal to the job completion time of any n out of the $n+m$ coded jobs. To heuristically understand the gain that can be obtained by MDS, again assume that the system is in stationarity with each queue independently having a sojourn time distribution F_W. A batch of n jobs arrives and are coded into $n+m$ MDS jobs. Their completition time has the same distribution as n-th order statistics of $n+m$ random variables with distribution F_W, whose complementary distribution is known to be given by

$$\begin{aligned}
&\mathbb{P}(C^{n,m} > t) \\
&= (n+m)\binom{n+m-1}{n-1} \times \\
&\sum_{k=0}^{n-1} \binom{n-1}{k} \frac{(-1)^k}{m+k+1} (1 - F_W(t))^{m+k+1}.
\end{aligned}$$

As $t \to \infty$, the tail is equivalent to its leading term

$$(n+m)\binom{n+m-1}{n-1} \frac{1}{m+1} (1 - F_W(t))^{m+1}. \quad (2)$$

Thus the tail of the response time with the MDS strategy is smaller than the tail of the response time achieved by replication-d in (1) so long as $m \geq d$. This non-rigorous sketch illustrates the main message of this paper: where nd copies of jobs are used for replication-d, for digital data subject to linear processing, MDS can provide better tail response times with only $n+d$ copies.

We present the precise model considered in the paper in Section 2. In Section 3 we consider the case where jobs have exponential service times and where k, the number of servers, tends to infinity. Under the mean-field assumption used in the replication-d literature, we demonstrate that as long as $m \geq d$ the tail distribution of batch completion times of jobs in stationarity is strictly smaller in the case of MDS when compared with replication-d, making the above heuristic arguments rigorous. We devote Section 4 to a heuristic analysis of the general service-time distribution case which demonstrates that our results should hold in full generality.

2. A more precise model

In the rest of the paper we shall assume that there are k servers, each with an infinite-buffer queue to store outstanding jobs. Each arrival is a batch of n jobs that appears according to a Poisson process of intensity $\lambda k/n$, so that, on average, there are λk jobs arriving per unit of time. For digital data as in communication networks, the batch arrival assumption is not restrictive as individual jobs can be sub-divided. Batch arrivals are also appropriate to represent the parallelisation of MapReduce computations and more general parallel-processing computer systems (see, e.g., [23]).

We assume each version of any job takes an exponential time with rate 1 to complete on any server, taken independently of everything else, including other copies of the same job, and each server's output is a linear function of its input.

Another key question is how, once one copy of a job has been processed to completion, its remaining copies are treated. In some circumstances, such as the queueing of data jobs in a communications network, it is not practical to remove copied jobs and they must be served. In other instances, such as for parallelisation of MapReduce computations, it would be possible to remove waiting tasks from queues and cease the service of copies being processed. This latter setting, considered in [7] and references therein, provides a model of greater mathematical interest, and we focus on it in this paper.

3. MDS vs replication-d with redundant removals

The tails of batch completition times are challenging to analyse, but one can examine the behaviour in the limit as the number of queues, k, becomes large.

In the system with replication, the job completition time distribution is derived in [7, Section 5]) under an assumption on asymptotic independence of queues (Assumption 1, given below). It is straightforward to adapt that derivation to the case of batch arrivals considered here. The tail of the completion time of any one of the n jobs is given by

$$\mathbb{P}(R^{1,d} > t) = \left(\frac{1}{\lambda + (1-\lambda)e^{t(d-1)}}\right)^{d/(d-1)}.$$

The completion time for n jobs in a batch is then the maximum of n independent random variables with this distribution, and a batch's response time then has the tail given by

$$\mathbb{P}(R^{n,d} > t) = 1 - \left(1 - \left(\frac{1}{\lambda + (1-\lambda)e^{t(d-1)}}\right)^{d/(d-1)}\right)^{n}. \quad (3)$$

Following [7], we adopt Assumption 1 on the asymptotic independence of the queues for our analysis of the MDS strategy, and refer to that article for a discussion of it.

Assumption 1. *Let T_i denote the completition time of a job, not subject to removal, at queue i out of a total of k. For k sufficiently large, the random variables $(T_{i_1}, \dots T_{i_{n+m}})$ are independent for any distinct $i_1, \dots, i_{n+m}$.*

We prove the following along the lines of the proof introduced in [7, Section 5]. We note that a differential equation on the completion times may also be obtained from the result of [24, Theorem 5.2] where much more general workload-based policies are considered (using derivations similar to those in Section 6.1 therein). We however present below a simple derivation of (4), using only straightforward queueing arguments.

Theorem 2. *For every batch arrival of n jobs, $n+m$ coded jobs are sent to $n+m$ queues chosen uniformly at random without replacement. As soon as any n coded jobs are completed, the remaining m jobs are removed from the system, including those currently in service. Let T be the random waiting time of a single (virtual) job that is subject to neither coding or removal, which is needed to determine the batch waiting time. Under Assumption 1, its waiting time satisfies the following differential equation*

$$\frac{d\mathbb{P}(T > t)}{dt} = -\mathbb{P}(T > t) \quad (4)$$

$$+ \alpha(n+m-1)\binom{n+m-2}{n-1}\times \sum_{i=0}^{n-1}\binom{n-1}{i}\frac{(-1)^i}{(m+i)(m+i+1)}(\mathbb{P}(T > t))^{m+i+1}, \quad (5)$$

where $\alpha = \lambda(m+n)/n$. For large waiting times, the tail of its distribution satisfies

$$\limsup_{t\to\infty} e^t \mathbb{P}(T > t) < \infty.$$

Let $C^{n,m}$ denote the random MDS batch completion time. Its distribution satisfies

$$\mathbb{P}(C^{n,m} > t) = (n+m)\binom{n+m-1}{n-1}\times \sum_{i=0}^{n-1}\binom{n-1}{i}\frac{(-1)^i}{m+i+1}\left(\mathbb{P}(T > t)\right)^{m+i+1}, \quad (6)$$

and its tail satisfies

$$\limsup_{t\to\infty} e^{(m+1)t}\mathbb{P}(C^{n,m} > t) < \infty. \quad (7)$$

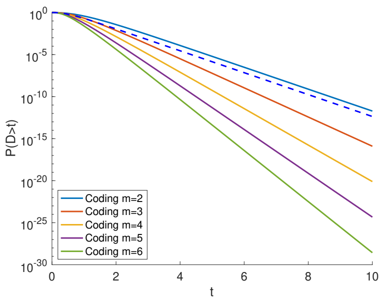

Figure 1: Complementary batch completition time distribution for a batch of $n = 3$ jobs. The dashed line corresponds to replication with $d = 3$ giving nd replicated jobs, while the solid lines correspond to MDS with $n + m$ coded jobs.

We note that this result encompasses the replication-d strategy by setting $n = 1$ and $m = d-1$, and that in that setting (4) is exactly the differential equation obtained in [7] (see the displayed equation just after (24) therein). These results confirm the heuristic analysis in the introduction that the tail of the batch completition time distribution for replication in equation (3) is slower than when MDS is used, equation (7), so long as $m \geq d$. Thus with only $n + d$ coded jobs, one can achieve better tail performance than sending nd jobs under the replication strategy.

In the case of replication, a closed form for the batch completition time distribution is available, but that is not the case for MDS as the differential equation (4) describing the virtual waiting time distribution of a non-coded job cannot be solved in closed form in general. It can, however, be readily solved numerically and inserted into equation (6) to evaluate the batch completition time distribution for MDS.

An example comparison is presented in Fig. 1 where batches consist of $n = 3$ jobs and the replication strategy places $d = 3$ copies of each into the system. For MDS, we consider a range of values for m from 2 to 6. The figure recapitulates the conclusion that MDS with $m \geq d$ leads to significant gains in completition time tail for larger values of t. Note that for $m = d = 3$, the batch completition time of MDS is not stochastically dominated by that of replication and that short delays are more likely with MDS, and it is only in the tail that MDS outperforms replication. For values of $m \geq 4$, however, the MDS batch completition time distribution is better for all times.

Proof of Theorem 2. Equation (6) is a direct application of known results on order-statistics distributions. We now prove (4). Following derivations in [7, Section 5], denote by T_i the non-redundant response time for a tagged job in queue i. Then

$$T_i = W_i + E_i,$$

where W_i is the workload (real workload, i.e. time to empty the queue if there were no more arrivals) and E_i is an Exponential(1) random variable (service requirement). Let us denote by F_T and F_W the distribution functions of T and W, respectively. Let us also denote their tails by $\overline{F}_T$ and $\overline{F}_W$. Then we can write

$$\begin{aligned}\overline{F}_T(t) &= e^{-t} + \int_0^t e^{-y}\overline{F}_W(t-y)dy \\ &= e^{-t} + \int_0^t e^{-(t-y)}\overline{F}_W(y)dy \\ &= e^{-t} + e^{-t}\int_0^t e^{y}\overline{F}_W(y)dy,\end{aligned}$$

differentiating which we get

$$\overline{F}_T{}'(t) = \overline{F}_W(t) - \overline{F}_T(t). \tag{8}$$

Let us now write an expression for $\overline{F}_W(t)$. We can look at the previous arrival (which in the case of MDS happens an Exponential($\lambda(m+n)/n$) time earlier than the tagged arrival. For simplicity denote $\alpha = \lambda m/n$. Condition first on the previous arrival having been y time before the tagged arrival. Then $W > t$ in one of two cases: either the previous arrival sees workload larger than $t + y$, or the previous arrival sees workload smaller than $t+y$, its own (non-redundant) time in queue i is larger than $t + y$ and by the time $t + y$ no more than $n - 1$ of the other $n + m - 1$ copies left other queues (or, in other words, nth order statistic of $n+m-1$ random variables exceeds $t + y$);

Integrating over all values of y, we obtain:

$$\overline{F}_W(t) = \int_0^\infty \alpha e^{-\alpha y} dy \Bigg(\overline{F}_W(t+y) + (\overline{F}_T(t+y) - \overline{F}_W(t+y)) \binom{n+m-2}{n-1} \times (n+m-1) \sum_{i=0}^{n-1} \binom{n-1}{i} \frac{(-1)^i}{m+i} (\overline{F}_T(t+y))^{m+i}$$

$$= e^{\alpha t} \int_t^\infty \alpha e^{-\alpha z} dz \Bigg(\overline{F}_W(z) + (\overline{F}_T(z) - \overline{F}_W(z))(n+m-1) \binom{n+m-2}{n-1} \times \sum_{i=0}^{n-1} \binom{n-1}{i} \frac{(-1)^i}{m+i} (\overline{F}_T(z))^{m+i} \Bigg), \quad (9)$$

where we used known results for the tail distribution of nth order statistic of $n+m-1$ random variables. Differentiating the above relation, we get

$$\overline{F}_W{}'(t) = \alpha \overline{F}_W(t) - \alpha \Bigg(\overline{F}_W(t) + (n+m-1) \binom{n+m-2}{n-1} \times (\overline{F}_T(t) - \overline{F}_W(t)) \sum_{i=0}^{n-1} \binom{n-1}{i} \frac{(-1)^i}{m+i} (\overline{F}_T(t))^{m+i} \Bigg)$$

$$= \alpha(n+m-1) \binom{n+m-2}{n-1} \times \overline{F}_T{}'(t) \sum_{i=0}^{n-1} \binom{n-1}{i} \frac{(-1)^i}{m+i} (\overline{F}_T(t))^{m+i},$$

where in the last equality we used (8). The above can be integrated and plugged into (8) to obtain

$$\overline{F}_T{}'(t) = -\overline{F}_T(t) + \alpha(n+m-1) \binom{n+m-2}{n-1} \times \sum_{i=0}^{n-1} \binom{n-1}{i} \frac{(-1)^i}{(m+i)(m+i+1)} (\overline{F}_T(t))^{m+i+1}.$$

This proves (4). Recall that $m - n \geq 1$. Since $\overline{F}_T(x) \to 0$ as $x \to \infty$, there exists X such that

$$\overline{F}_T{}'(x) \leq -\overline{F}_T(x) + A(\overline{F}_T(x))^2,$$

for all $x \geq X$, with some constant $A > 0$. Denote by $g(x) = 1/\overline{F}_T(x)$. Then

$$g'(x) = -\frac{\overline{F}_T(x)}{(\overline{F}_T(x))^2} \geq -\frac{-\overline{F}_T(x) + A(\overline{F}_T(x))^2}{(\overline{F}_T(x))^2} = g(x) - A$$

for all $x \geq X$. Since $g(x) \to \infty$ as $x \to \infty$, we can also assume that $g(x) \geq A$ for all $x \geq X$. Then the above implies that

$$\frac{g'(x)}{g(x) - A} \geq 1$$

for all $x \geq X$. Integrating the above inequality from X to x implies

$$\log \left(\frac{g(x) - A}{g(X) - A} \right) \geq x - X,$$

and hence $g(x) \geq A + (g(X) - A)e^{x-X} \geq Be^x$, with a constant B. This of course implies (7), and the proof is complete. □

4. General service-time distributions

The results found above demonstrate that MDS coding with $n+m$ copies of a batch of n performs better, in terms of the tail of response time in a large system, than replication with d copies for every job as long as $m \geq d$ and assuming that service times of jobs are exponential. A natural question is whether the results hold for general service times. Rigorous analysis of that more general setting is, however, extremely challenging as even stability conditions are not known in the replication scenario. Indeed, as follows from [25], the average increase in the total workload of the entire system (defined as the sum of workloads of individual queues seen as time to empty a queue if no new jobs arrive) caused by an arriving job is not smaller (resp., not larger) than the average size of a single job in the case when service times have non-decreasing (resp., non-increasing) hazard rates. One can see from the derivation of these results that the distribution of the amount of work brought into the system by an arriving job depends non-trivially on the state of the system. Therefore stability conditions even for distribution functions with monotone hazard rates are difficult to obtain. To the best our knowledge, there are no results at all for distributions with non-monotone hazard rates.

Despite rigorous analysis appearing to be out of reach, we conjecture that the exponential service time results remain valid in the case of more general service-time distributions and present a heuristic justification. Again we make use of assumption 1 and assume also that all systems under consideration are stable, by which we mean the existence of stationary distributions for the workloads and servers defined as the time to empty them if no further customers arrive. In addition, we assume that these stationary workloads have finite expectations.

Note again that the replication strategy is a particular case of the coding strategy with $n = 1$ and $m = d-1$. Therefore, if one wishes to demonstrate that the coding strategy with parameters n and m outperforms the replication strategy with a parameter d as long as $m \geq d$ in the asymptotics of the tail of the response time, it is sufficient to show that the tail asymptotics of the response time in the coding strategy do not depend on n and improve as m increases.

In the notation similar to that of the proof of Theorem 2, we can write

$$T_i = W_i + S_i,$$

where T_i denotes the non-redundant response time for a tagged job in queue i, W_i denotes the workload in queue i and S_i denotes the service time of the tagged job in queue i, which we assume in addition to have a finite expectation. As we have assumed that W_i has a finite expectation, this implies that T_i has a finite expectation too. The tail of the batch completion time is then (as in (6)) given by

$$\begin{aligned}&(n+m)\binom{n+m-1}{n-1}\times\\&\sum_{i=0}^{n-1}\binom{n-1}{i}\frac{(-1)^i}{m+i+1}\left(\mathbb{P}(T>t)\right)^{m+i+1}\\&\sim C(\mathbb{P}(T>t))^{m+1}\end{aligned}$$

as $t \to \infty$, where T is a random variable with the distribution of any of T_i, and C is a constant that depends on n. As only the constant depends on n, the asymptotic equivalence above demonstrates that it is sufficient to show that the tail of T does not depend on n, which immediately implies that the asymptotic tail behaviour does not depend on n and improves as m increases.

Note that (9) is valid as only the exponential distribution of inter-arrival times is used to obtain it. Similarly to the derivation of the following equation in the proof of Theorem 2, one can obtain, by differentiating (9),

$$\overline{F}_W{}'(t) = (\overline{F}_W(t) - \overline{F}_T(t))g(t),$$

where

$$\begin{aligned}g(t) = \alpha(n+m-1)&\binom{n+m-2}{n-1}\times\\&\sum_{i=0}^{n-1}\binom{n-1}{i}\frac{(-1)^i}{m+i}(\overline{F}_T(t))^{m+i}\\&\sim C(\overline{F}_T(t))^m\end{aligned}$$

as $t \to \infty$, with a constant C. One can solve this to obtain

$$\overline{F}_W(t) = e^{\int_{s=0}^{t} g(s)ds}\int_{z=t}^{\infty} g(z)\overline{F}_T(z)e^{-\int_{s=0}^{z} g(s)ds}dz.$$

We note here that, as our assumptions imply that T has a finite expectation, its tail $\overline{F}_T$ is integrable, and thus all integrals in the above are finite. The above implies that

$$\begin{aligned}\overline{F}_W(t) &\sim C_1\int_{z=t}^{\infty} g(z)\overline{F}_T(z)dz\\&\sim C_2\int_{z=t}^{\infty}\left(\overline{F}_T(z)\right)^{m+1}dz\end{aligned}$$

as $t \to \infty$, with some constants C_1 and C_2. The above asymptotic equivalence, along with the equation $T_i = W_i + S_i$, indicates that the asymptotic behaviour of the tails of W and T should not depend on n as only the constant C_2 above may depend on n.

[1] Y. Azar, A. Z. Broder, A. R. Karlin, E. Upfal, Balanced allocations, SIAM journal on computing 29 (1) (1999) 180–200.

[2] M. Mitzenmacher, The power of two choices in randomized load balancing, IEEE Transactions on Parallel and Distributed Systems 12 (10) (2001) 1094–1104.

[3] N. D. Vvedenskaya, R. L. Dobrushin, F. I. Karpelevich, Queueing system with selection of the shortest of two queues: An asymptotic approach, Problemy Peredachi Informatsii 32 (1) (1996) 20–34.

[4] M. van der Boor, S. C. Borst, J. S. van Leeuwaarden, D. Mukherjee, Scalable load balancing in networked systems: A survey of recent advances, arXiv preprint arXiv:1806.05444.

[5] N. B. Shah, K. Lee, K. Ramchandran, When do redundant requests reduce latency?, IEEE Transactions on Communications 64 (2) (2015) 715–722.

[6] K. Gardner, S. Zbarsky, S. Doroudi, M. Harchol-Balter, E. Hyytia, Reducing latency via redundant requests: Exact analysis, ACM SIGMETRICS Performance Evaluation Review 43 (1) (2015) 347–360.

[7] K. Gardner, M. Harchol-Balter, A. Scheller-Wolf, M. Velednitsky, S. Zbarsky, Redundancy-d: The power of d choices for redundancy, Operations Research 65 (4) (2017) 1078–1094.

[8] R. Roth, Introduction to coding theory, Cambridge University Press, 2006.

[9] L. Huang, S. Pawar, H. Zhang, K. Ramchandran, Codes can reduce queueing delay in data centers, in: IEEE International Symposium on Information Theory, 2012, pp. 2766–2770.

[10] N. B. Shah, K. Lee, K. Ramchandran, The mds queue: Analysing the latency performance of erasure codes, in: 2014 IEEE International Symposium on Information Theory, IEEE, 2014, pp. 861–865.

[11] G. Joshi, Y. Liu, E. Soljanin, On the delay-storage trade-off in content download from coded distributed storage systems, IEEE Journal on Selected Areas in Communications 32 (5) (2014) 989–997.

[12] B. Li, A. Ramamoorthy, R. Srikant, Mean-field-analysis of coding versus replication in cloud storage systems, in: IEEE International Conference on Computer Communications, IEEE, 2016, pp. 1–9.

[13] K. Lee, N. B. Shah, L. Huang, K. Ramchandran, The MDS queue: Analysing the latency performance of erasure codes, IEEE Transactions on Information Theory 63 (5) (2017) 2822–2842.

[14] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, K. Ramchandran, Speeding up distributed machine learning using codes, IEEE Transactions on Information Theory 64 (3) (2017) 1514–1529.

[15] R. Tandon, Q. Lei, A. G. Dimakis, N. Karampatziakis, Gradient coding: Avoiding stragglers in distributed learning, in: International Conference on Machine Learning, 2017, pp. 3368–3376.

[16] L. Chen, H. Wang, Z. Charles, D. Papailiopoulos, Draco: Byzantine-resilient distributed training via redundant gradients, Proceedings of Machine Learning Research 80 (2018) 903–912.

[17] W. Halbawi, N. Azizan, F. Salehi, B. Hassibi, Improving distributed gradient descent using Reed-Solomon codes, in: IEEE International Symposium on Information Theory, 2018, pp. 2027–2031.

[18] R. K. Maity, A. S. Rawat, A. Mazumdar, Robust gradient descent via moment encoding with LDPC codes, in: SysML Conference, 2019.

[19] E. Ozfaturay, D. Gunduz, S. Ulukus, Speeding up distributed gradient descent by utilizing non-persistent stragglers, in: IEEE International Symposium on Information Theory, 2019.

[20] E. Ozfatura, D. Gunduz, S. Ulukus, Gradient coding with clustering and multi-message communication, in: IEEE Data Science Workshop, 2019.

[21] I. S. Reed, G. Solomon, Polynomial codes over certain finite fields, Journal of the society for industrial and applied mathematics 8 (2) (1960) 300–304.

[22] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, B. Leong, A random linear network coding approach to multicast, IEEE Transactions on Information Theory 52 (10) (2006) 4413–4430.

[23] L. Ying, R. Srikant, X. Kang, The power of slightly more than one sample in randomized load balancing, Mathematics of Operations Research 42 (3) (2017) 692–722.

[24] T. Hellemans, T. Bodas, B. Van Houdt, Performance analysis of workload dependent load balancing policies, Proceedings of the ACM on Measurement and Analysis of Computing Systems 3 (2) (2019) 35.

[25] G. Joshi, E. Soljanin, G. Wornell, Efficient replication of queued tasks for latency reduction in cloud systems, in: 2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton), IEEE, 2015, pp. 107–114.