On Gradient Coding with Partial Recovery

Sahasrajit Sarmasarkar*, V. Lalitha† and Nikhil Karamchandani*
*Indian Institute of Technology, Bombay
Email: sahasrajit1998@gmail.com, nikhilk@ee.iitb.ac.in
†International Institute of Information Technology, Hyderabad
Email: lalitha.v@iiit.ac.in

Abstract

We consider a generalization of the recently proposed gradient coding framework where a large dataset is divided across n workers and each worker transmits to a master node one or more linear combinations of the gradients over the data subsets assigned to it. Unlike the conventional framework which requires the master node to recover the sum of the gradients over all the data subsets in the presence of s straggler workers, we relax the goal of the master node to computing the sum of at least some α fraction of the gradients. The broad goal of our work is to study the optimal computation and communication load per worker for this approximate gradient coding framework. We begin by deriving a lower bound on the computation load of any feasible scheme and also propose a strategy which achieves this lower bound, albeit at the cost of high communication load and a number of data partitions which can be polynomial in the number of workers n. We then restrict attention to schemes which utilize a number of data partitions equal to n and propose schemes based on cyclic assignment which have a lower communication load. When each worker transmits a single linear combination, we also prove lower bounds on the computation load of any scheme using n data partitions.

I. Introduction

In a distributed computing framework, a job is divided into multiple parallel tasks, which are computed on different servers, and the job is finished when all the tasks are complete. In this framework, a subset of workers can be arbitrarily slow as compared to the rest of the workers. These subset of workers are referred to as stragglers. Since the slowest tasks determine the job execution time, they form a bottleneck to the efficient execution of the job. Recently, there has been an extensive amount of work to mitigate the effect of stragglers by introducing redundancy in the computed tasks using coding theoretic techniques. The distributed computing applications for which codes have been designed include matrix-vector multiplication [1], matrix-matrix multiplication [2], [3], gradient computation [4], polynomial computation [5] and coded convolution [6]. A fundamental trade-off between computation and communication cost was established in [7], for the case of general distributed data shuffling problem.

A. Gradient Coding

In various machine learning applications, a principal task is to compute the gradient sum on large datasets. Hence, gradient sum computation is a natural application for distributed computing. Consider a dataset of d points over which the gradient sum of a certain objective function needs to be computed. In the case of uncoded computing, the data set is divided into n data subsets. Each worker computes a partial gradient on the data subset assigned to it and returns the results to the master node. The master computes the full gradient sum by combining the results. However, this scheme is not efficient when there are stragglers amongst the n worker nodes. Towards addressing this issue, *Gradient Coding* was proposed in [4], which ensures efficient distributed gradient computation even in the presence of stragglers by utilizing coding-theoretic techniques. For any scheme which is tolerant to s stragglers, a lower bound of $s+1$ on the computation load per worker was derived. Optimal gradient coding schemes, which achieve the lower bound with equality, were provided based on fractional repetition and cyclic assignments of data subsets.

The scheme based on cyclic assignment of data subsets in [4], is based on a random coding argument and hence the result is existential in nature. Explicit gradient coding schemes based on cyclic MDS codes over complex numbers and on Reed Solomon codes were designed in [8] and [9] respectively. When gradient sum computation can be formulated as a multivariate polynomial evaluation problem, the Lagrange coded computing scheme has been proposed in [5]. Communication-efficient gradient coding was introduced in [10] where the master node has to recover a gradient sum vector and it proposes coding across the elements of the gradient vector to reduce the number of transmitted symbols. Multi-message communication based gradient codes allow for multiple messages to be transmitted from workers to the master in each round and have been studied in [11], [12] which use this capability to utilize the work done by non-persistent stragglers. Heterogeneity-aware gradient coding was introduced in [13], where in addition to stragglers, heterogeneous non-straggling workers have been considered. The problem of distributed linearly separable computation has been introduced in [14]. Distributed linear transforms and gradient computation are special cases of this problem.

B. Approximate Gradient Coding

The above works consider the objective of exactly recovering the gradients sum in the presence of stragglers and as mentioned before, a fundamental converse argument in [4] finds that this requires the per worker computation load to scale linearly with the straggler tolerance level. Several works have found that for many practical distributed learning applications, it suffices to approximately recover the gradient sum [15]–[20]. Gradient coding schemes which trade-off the computation load and the ℓ_2 error between the actual full gradient and the computed full gradient, have been studied in [8], [21]–[23]. In this work, we consider the setting of gradient coding with partial recovery in which the gradient computed at the master is required to be the sum of at least α fraction of the data subsets. This is a different form of approximation as compared to that considered in previous works on gradient coding mentioned above and such forms of approximate gradient recovery have found application in distributed learning algorithms [15], [16]. A similar objective function was also studied recently in [12], [24], [25], where the goal was to design strategies which benefit from both uncoded and coded computing schemes, and extensive numerical simulations were done to illustrate the advantages of allowing partial recovery. Finally, we would like to point out that while [21] studied approximate gradient coding in terms of ℓ_2 error, their gradient code construction based on Batched Raptor codes can in fact be applied to the partial recovery framework being studied here as well. However, the guarantees are probabilistic in nature, where the randomization is on the set of stragglers whereas our focus here is on the deterministic worst-case setup as proposed for the original gradient coding problem [4].

C. Our Contributions

For the gradient coding with partial recovery framework, we give a lower bound on the computation load at each worker, which is independent of the number of data subsets. We provide two schemes which achieves this bound with equality, the second having marginally better communication load than the first for a subset of parameters. Though these schemes have minimum computation load, they have high communication load and require partitioning into a large number of data subsets. We give another class of cyclic gradient codes with the number of data subsets being equal to the number of workers in which every worker transmits at most two linear combinations of the gradients of the data subsets assigned to it, but has a slightly higher computation load. We also give a lower bound on the computation load per worker of any scheme with the number of data subsets being equal to the number of workers when each worker transmits one linear combination.

II. Problem Formulation

Consider a dataset D consisting of features-label pairs $\{(x_i, y_i)\}_{i=1}^d$ with each tuple $(x_i, y_i) \in \mathbb{R}^p \times \mathbb{R}$. Several machine learning problems wish to solve problems of the following form:

$$\beta^* = \arg\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^{d} L(x_i, y_i; \beta) + \lambda R(\beta)$$

where $L(.)$ is a task-specific function and $R(.)$ is the regularisation function. Often this problem is solved using gradient-based iterative approaches by computing the gradient at each step using the current value of the model $\beta^{(t)}$. Let $g^{(t)} := \sum\limits_{i=1}^{d} \nabla L(x_i, y_i : \beta^{(t)})$ be the gradient of the loss function computed at t^{th} step and the model parameter is updated as $\beta^{(t+1)} = h_R(\beta^{(t)}, g^{(t)})$ for some suitable mapping h_R. As the size d of the dataset becomes large, the computation of the gradient $g^{(t)}$ can become a bottleneck and one possible solution is to parallelize the computation by distributing the task across multiple workers.

We consider a gradient coding framework with n workers denoted by $W_1, W_2, ..., W_n$ and a master node. The entire data set D is divided into k equal partitions $D_1, D_2, .., D_k$ and let $\{g_l\}$ denote the partial gradients[1] over the data subsets $\{D_l\}$. Each worker i computes $m \geq 1$ linear combinations of $\{g_l\}$ given by (coded partial gradient) $\tilde{g}^i = [\tilde{g}_1^i; \tilde{g}_2^i; ...; \tilde{g}_m^i]$ with $\tilde{g}_j^i = \sum\limits_{l=1}^{k} A_{j,l}^i \cdot g_l$ for each $j \in [m]$, and transmits them to the master node. Let $A^i \in \mathbb{R}^{m \times k}$ denote the computation matrix corresponding to worker i with its $(j,l)^{th}$ entry given by $A_{j,l}^i$. We define the communication load and computation load of the gradient coding scheme described above.

Definition II.1. (Communication Load): For a gradient coding scheme specified by $\{A^i\}$, we define the communication load as m where m denotes the number of coded partial gradients transmitted by each worker.

[1]We drop the superscript t in the gradient notation for convenience

Definition II.2. (Max. Computation Load per worker): For a gradient coding scheme with communication load m and specified by $\{A^i\}$, we define the load per worker by $l = \frac{1}{k}.\max_{i\in[n]} |\bigcup_{j\in[m]} \text{supp}(A^i_j)|$ where $\text{supp}(A^i_j)$ denotes the set of non-zero entries in the j^{th} row of A^i.

The data subsets assigned to a worker W_i is $\{D_v : v \in \bigcup_{j\in[m]} \text{supp}(A^i_j)\}$. Note that we define the computation load relative to the total number of partitions k of the entire data set. On the other hand, the communication load m is not normalized since the size of each worker transmission is independent of the number of data subsets k.

We will refer to a gradient coding scheme with n workers, k data subsets, communication load m, and maximum computation load per worker l as an (n, k, m, l) gradient coding (GC) scheme. In conventional gradient coding schemes, the goal of the master node is to recover the sum of the partial gradients $\{g_i\}$ over all the k data subsets $\{D_i\}$ in the presence of straggler worker nodes. We now define a new framework in which the requirement for the master is relaxed to being able to recover the sum of a certain fraction of the partial gradients.

Definition II.3. ((α, s)-feasible (n, k, m, l) gradient coding (GC) schemes): For $\alpha \in (0, 1], 1 \leq s \leq n$, we call an (n, k, m, l) gradient coding scheme as (α, s)-feasible if the master node is able to compute $\sum_{i\in I} g_i$ for some $I \subseteq [k]$, $|I| \geq \alpha k$ whenever any $n - s$ workers are able to successfully communicate their results to the master node .

Thus, if an (n, k, m, l) GC is (α, s)-feasible, then it can tolerate s stragglers out of the n workers. Also, note that for $\alpha = 1$ the above definition reduces to that of conventional gradient codes. Finally, we will restrict attention to linear schemes here and thus for such a scheme, there must exist a vector $v \in \{0, 1\}^k$ with $||v||_0 \geq \alpha.k$ in the span of the rows of $\{A^i\}_{i\in I}$ for every $|I| \geq n - s$.

Our goal in this work is to analyze the minimum communication load (m) and computation load per worker (l) for (α, s)-feasible (n, k, m, l) GC schemes. One naive strategy to create such a GC scheme is to select some $\alpha.k$ data partitions out of D and then use a conventional (full) gradient coding scheme to recover the sum of gradients over the $\alpha.k$ data partitions while allowing for any set of s workers to straggle. Such a schemes would have a communication load of 1 and a lower bound of $\alpha(s + 1)/n$ on the max. computation load per worker [4]. In this work, we will propose (α, s)-feasible (n, k, m, l) GC schemes that have far lower max. computation load per worker.

III. Lower bound on the computation load

We begin by proving a lower bound on the computation load per worker l for any (α, s)-feasible (n, k, m, l) GC scheme.

Theorem 1. *For any (α, s)-feasible (n, k, m, l) GC scheme and $y = \lceil n.l \rceil$, we have*

$$\frac{\binom{s}{y}}{\binom{n}{y}} \leq 1 - \alpha. \tag{1}$$

The inequality in (1) implies a lower bound on $y = \lceil nl \rceil$ and for a scheme which assigns the same load to each worker, y denotes the average number of copies for each data subset stored across the n workers. Note that while the above lower bound is dependent on the parameters n, s, and α, it is independent of the number of data subsets k and communication load m. Also, for $\alpha = 1$ which corresponds to the conventional gradient coding setup, the lower bound above reduces to $y \geq s + 1$ as obtained in [4, Theorem 1].

To prove Theorem 1, we will first derive an intermediate condition given in the following lemma.

Lemma 1. *Consider any (α, s)-feasible (n, k, m, l) GC scheme and let y_i denote the number of distinct workers which are assigned the data subset D_i. Then, the following condition holds:*

$$\sum_{i=1}^{k} \binom{n - y_i}{n - s} \leq \binom{n}{s} k(1 - \alpha). \tag{2}$$

Proof. Consider all possible subsets of size s of the set of n workers and denote these subsets by $\{S_j\}$ for $j \in [\binom{n}{s}]$. Now consider any data subset D_i for some $i \in [k]$ and let E_i denote the set of workers it is assigned to. From the statement of the lemma, we have $|E_i| = y_i$. From the definition of an (α, s)-feasible (n, k, m, l) GC scheme, we have that each subset of $(n - s)$ workers should have access to at least α fraction of the datasets, and thus for each subset S_i of size s there can be at most $k(1 - \alpha)$ subsets E_j such that $E_j \subseteq S_i$.

For each $j \in [\binom{n}{s}]$, let $k_j = |\{E_i | i \in [k]; E_i \subseteq S_j\}|$, whose sum we bound in the following argument. From the argument above, $\sum_{j\in[\binom{n}{s}]} k_j \leq \binom{n}{s} k(1-\alpha)$. On the other hand, each set E_i is a subset of exactly $\binom{n-y_i}{s-y_i}$ subsets S_j for $j \in [\binom{n}{s}]$. Thus, we get $\sum_{i=1}^{k} \binom{n-y_i}{n-s} = \sum_{j\in[\binom{n}{s}]} k_j \leq \binom{n}{s} k(1-\alpha)$, completing the proof.

□

Now, we will use Lemma 1 to prove Theorem 1.

Proof of Theorem 1. Consider any (α, s)-feasible (n, k, m, l) GC scheme and let y_i denote the number of distinct workers which are assigned the data subset D_i. From the definition of the max. load per worker l, we have $\sum_{i\in[k]} y_i \leq n.k.l$ since each worker can be assigned at most $k.l$ data subsets. Furthermore, we have $\sum_{i=1}^{k} \binom{n-y_i}{n-s} \leq \binom{n}{s} k(1-\alpha)$ from Lemma 1.

Now define $b = \lfloor \frac{\sum_{i=1}^{k} y_i}{k} \rfloor$ and $k_1 = (b+1)k - \sum_{i=1}^{k} y_i$, thus from the claim 1, $k_1 \binom{n-b}{n-s} + (k-k_1)\binom{n-b-1}{n-s} \leq \sum_{i=1}^{k} \binom{n-y_i}{n-s}$ since $\sum_{i=1}^{k}(n-y_i) = k_1 \times (n-b) + (k-k_1) \times (n-b-1)$ and $n-b-1 = \lfloor \frac{\sum_{i=1}^{n}(n-y_i)}{k} \rfloor$. The L.H.S is the smallest when $\sum_{i=1}^{k} y_i = n \times k \times l$ since a increases with $\sum y_i$ and k_1 decreases with $\sum y_i$ when a is constant. Thus, the inequality reduces to $k\binom{n-b-1}{n-s} \leq \sum_{i=1}^{k} \binom{n-y_i}{n-s} \leq \binom{n}{s} k(1-\alpha)$ where $a = \lfloor n.l \rfloor$ because $\binom{n-b-1}{n-s} \leq \binom{n-b}{n-s}$ which proves Theorem 1.
□

Claim 1. *Consider any collection of t positive integers $\{a_i\}_{1\leq i\leq t}$. Define $a = \lfloor \frac{\sum_{i=1}^{t} a_i}{t} \rfloor$ and let t_1 be the unique positive integer satisfying $\sum a_i = t_1.a + (t-t_1)(a+1)$. Then we have $\sum_{i=1}^{t} \binom{a_i}{r} \geq t_1.\binom{a}{r} + (t-t_1).\binom{a+1}{r}$.*

IV. (α, s) FEASIBLE (n, k, m, l) GC SCHEMES WITH LEAST COMPUTATION LOAD

The following theorem shows that lower bound on computation load in Theorem 1 is achievable, albeit at high communication cost.

Theorem 2. *For every n, s, α and $1 \leq y \leq n$ satisfying $\frac{\binom{s}{y}}{\binom{n}{y}} \leq 1-\alpha$, there exists an (α, s)-feasible $(n, \binom{n}{y}, \binom{n-1}{y-1}, \frac{y}{n})$ GC scheme.*

Proof. We divide the full dataset D in to $k = \binom{n}{y}$ data subsets and index them by subsets of $[n]$ of size y, and for each $S \subset [n], S = \{i_1, i_2, \ldots, i_y\}$, let data subset D_S be assigned to workers $W_{i_1}, W_{i_2}, \ldots, W_{i_y}$. Thus, each worker would be assigned $\binom{n-1}{y-1}$ data subsets and the computation load per worker $l = \frac{\binom{n-1}{y-1}}{\binom{n}{y}} = \frac{y}{n}$. Each worker would then directly compute and individually transmit the gradients for all the data subsets assigned to it, which results in a communication load $m = \binom{n-1}{y-1}$. Next, we argue the correctness of this scheme.

Under any set of s stragglers, the number of data-parts which are not assigned to any worker other than these set of s stragglers is given by $\binom{s}{y}$. Thus, the master node can obtain the sum of at least $\binom{n}{y} - \binom{s}{y}$ gradients which is at least $\alpha.\binom{n}{y} = \alpha k$ since $\frac{\binom{s}{y}}{\binom{n}{y}} \leq 1-\alpha$. Thus the above mentioned scheme is an (α, s)-feasible GC scheme. □

We now show that the communication load can be slightly improved in some scenarios without incurring a penalty on the computation load per worker.

Theorem 3. *For every n, s, α and $1 \leq y \leq n$ which is co-prime with n and satisfies $\frac{\binom{s}{y}}{\binom{n}{y}} \leq 1-\alpha$, there exists an (α, s)-feasible $(n, \binom{n}{y}, 1 + \frac{y-1}{y}.\binom{n-1}{y-1}, \frac{y}{n})$ GC scheme.*

Proof. We assign data subsets to different workers in the same way as described in the proof of Theorem 2 and thus the number of data partitions k and the computation load per worker l remain the same. Let $I_j \subset [n], |I_j| = y$ denote the indices of the y workers to whom data subset D_j is assigned. For each data subset D_j, we choose a worker W_{t_j} from amongst the workers that data subset D_j is assigned to, i.e., $t_j \in I_j \ \forall j \in [\binom{n}{y}]$. This is done while ensuring that the process is balanced, i.e., each worker is chosen exactly the same number of times and thus we have $\forall\ i \in [n]$, $|B_i| = |\{j : j \in [\binom{n}{y}]$ s.t $i = t_j\}| = \binom{n-1}{y-1}/y$. Such an allocation is possible whenever y is co-prime with n and the details are provided in Appendix B. Next, each worker W_i transmits to the master node the sum of all the gradients assigned to it and in addition, individually transmits the gradients corresponding to all the data subsets assigned to it except those in B_i. Thus the communication load of this scheme is given by $1 + \frac{y-1}{y}\binom{n-1}{y-1}$.

We now describe the decoding procedure at the master node and argue the correctness of the scheme in the presence of at most s stragglers. Denote the set of non-straggler worker nodes by $I \subseteq [n]$ with $|I| \geq n-s$. Since the data

	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}
W_1	$1\times$	$1\times$	$1\checkmark$	$1\checkmark$						
W_2	$1\checkmark$				$1\times$	$1\times$	$1\checkmark$			
W_3		$1\checkmark$			$1\checkmark$			$1\times$	$1\times$	
W_4			$1\times$			$1\checkmark$		$1\checkmark$		$1\times$
W_5				$1\times$			$1\times$		$1\checkmark$	$1\checkmark$

TABLE 1

Assignment of data subsets (marked by $1\checkmark$ and $1\times$) to different workers in $(\frac{7}{10}, 3)$ feasible $(5, 10, 3, \frac{2}{5})$ GC scheme with data subsets marked by $1\checkmark$ having the corresponding gradients being directly transmitted by each corresponding worker.

subset assignment to the workers is identical to the one used in the proof of Theorem 2, we know that the number of gradients which are computed by at least one worker in I is greater than $\alpha k = \alpha\binom{n}{y}$. Thus to prove that the scheme is an (α, s)-feasible GC, it suffices to show that using the transmissions from the non-straggling worker nodes, the master node can recover the sum of the gradients corresponding to all data subsets assigned to them.

Recall that each non-straggler worker node in I transmits the sum of all its computed gradients in addition to some individual gradients. The master node adds up the sum transmissions from all nodes in I and then uses the individual gradient transmissions to suitably adjust the coefficients so that the sum of all the involved gradients can be recovered. Let $D_{1,I}$ denote the collection of data subsets which are assigned to exactly 1 worker amongst the non-straggling workers I. Clearly, the gradient of each such data subset in $D_{1,I}$ would have its coefficient as 1 in the above sum at the master node. Now consider the gradients of those data subsets which appeared more than once in the sum. Each such data subset must have been assigned to more than one worker in I and thus at least one worker in I would be directly transmitting the gradient of that data subset as per the scheme designed above. Thus, the master node can subtract an appropriate multiple of any such gradient from the sum calculated above and we can thus recover the sum of the gradients corresponding to all data subsets assigned to the non-straggling workers I. $\square$

Example 1. *An example for $n = 5$, $\alpha = 7/10$ and $s = 3$ is shown below as described above in proof of Thm 3. The smallest y satisfying $\frac{\binom{s}{y}}{\binom{n}{y}} \leq 1 - \alpha$ can be shown to 2. Since n and y are co-prime we can achieve a communication load of $1 + \frac{y-1}{y}\binom{n-1}{y-1} = 3$. The assignment of different data subsets to various workers is given in Table 1. Recall that under this scheme each worker W_i transmits the sum of the gradients of data subsets it is assigned and individually transmits gradients corresponding to those data subsets except those in B_i. For each worker W_i, the data subsets assigned to it which belong to B_i has been denoted by $1\times$ and the data subsets assigned to it but don't belong in B_i has been denoted by $1\checkmark$.*

For example, worker W_1 transmits the sum of the gradients of the data subsets D_1, D_2, D_3 and D_4 and individually the gradients of the data subsets corresponding to data subsets D_3 and D_4. For example if workers W_3, W_4 and W_5 straggle, the master can still compute the sum of gradients of subsets D_1 to D_7, D_4 , D_6, D_7 and D_1 using the transmissions by the workers W_1 and W_2. The master can compute the sum of the sum of the gradients transmitted by the workers W_1 and W_2 and subtract the gradient of the data subset D_1 which is transmitted by worker W_2.

Note that the scheme which just assigns only α fraction of data sets to the workers can be shown to have a lower bound on the max computation load per worker to be $\frac{\alpha(s+1)}{n} = 0.56$. Our scheme has a max. computation load per worker to be $\frac{2}{5}$ which is lower.

V. Cyclic (α, s)-feasible GC schemes

In the previous section, we presented two schemes which achieves minimum computation load at the cost of high communication load and large number of data partitions. In this section, we will consider (α, s)-feasible GC schemes, when the number of data subsets is restricted to n (the number of workers) and the assignment of data subsets is cyclic. We are interested in the cyclic assignment based GC schemes because they have been shown to be optimal for the case of gradient coding with full recovery [4], [8]. Also, for the case of gradient coding with partial recovery, random cyclic shift based schemes have been proposed in [12], though their optimality has not been shown. We provide two schemes based on cyclic assignment of data subsets of workers. The first scheme requires that the parameters of the GC scheme satisfy a certain divisibility criterion, in which case we show that there exists an (α, s)-feasible GC scheme with a communication load of 1. We then show that whenever the divisibility criterion is not met, cyclic schemes cannot achieve the desired computation load, when the communication load is 1. Finally, we show that there exists an (α, s)-feasible cyclic GC scheme with a communication load of 2, for all parameters.

Workers	D_1	D_2	D_3	D_4	D_5	D_6	D_7
W_1	1	1	1				
W_2		1	1	1			
W_3			1	1	1		
W_4				1	1	1	
W_5					1	1	1
W_6	1					1	1
W_7	1	1					1

TABLE 2
ASSIGNMENT OF DATA SUBSETS TO DIFFERENT WORKERS IN $(\frac{6}{7}, 3)$ FEASIBLE $(7, 7, 1, \frac{3}{7})$ GC SCHEME

Definition V.1. (Cyclic GC scheme): We define a (n, n, m, l) GC scheme as a cyclic GC scheme if worker W_1 is assigned the data subsets from D_1 to $D_{l\times n}$, worker W_2 is assigned the data subsets from D_2 to $D_{l\times n+1}$ and in general worker W_i is assigned the data subsets from D_i to $D_{1+((l\times n+i-2) \mod n)}$.

Theorem 4. *There exists an (α, s)-feasible $(n, n, 1, \frac{s+1+\beta-n}{n})$ cyclic GC scheme with $\beta = \lceil \alpha.n \rceil$ for every n, s, α if $s + 1 + \beta - n$ divides β.*

Proof. We follow the assignment scheme as described in Definition V.1 and each worker is assigned exactly $s+1+\beta-n$ data subsets. Each worker transmits the sum of the gradients of all the data subsets assigned to it.

To show that the scheme is (α, s)-feasible, we show that we can recover the sum of $\beta = \lceil \alpha.n \rceil$ data subsets in the presence of any s stragglers. Based on the straggler pattern, we pick a subset of $n - s$ non-straggling workers of size $\frac{\beta}{r}$, such that r data subsets assigned to these workers are mutually disjoint. We give an algorithm to identify these workers in Appendix 4.

□

Example 2. *An example for $n = 7$, $\alpha = 6/7$ and $s = 3$ is described below. The assignment of different data subsets to various workers is done using a cyclic GC scheme as described in the proof of Theorem 4 given in Table 2. Each worker transmits the sum of the gradients of data subsets it is assigned. For example, worker W_4 transmits the sum of the gradients of the data subsets D_4, D_5 and D_6. For example if workers W_1, W_4 and W_5 straggle, the master can still compute the sum of gradients of subsets D_2, D_3, D_4 , D_6, D_7 and D_1 using the transmissions by the workers W_2 and W_6.*

Note that the scheme which just assigns only α fraction of data sets to the workers can be shown to have a lower bound on the computation load per worker to be $\frac{\alpha(s+1)}{n} = 24/49$ as in [4, Theorem 1]. Also the cyclic scheme described in [4] has computation load per worker to be $\frac{4}{7}$. Our cyclic GC scheme has the computation load per worker as $3/7$ with a communication load of 1 which has smaller computation load than the two GC schemes described above. However we can achieve a smaller computation load using the scheme as described in the proof of Theorem 3 which would have a max. computation load per worker $\frac{y}{n} = 2/7$ though with a higher communication cost of $1 + \frac{y-1}{y}\binom{n-1}{y-1} = 4$.

The following theorem shows that if $s + 1 + \beta - n$ does not divide β, no (α, s) feasible $(n, n, 1, \frac{s+1+\beta-n}{n})$ cyclic GC scheme exists.

Theorem 5. *There exists no (α, s) feasible $(n, n, 1, \frac{s+1+\beta-n}{n})$ cyclic GC scheme if $s+1+\beta-n$ does not divide β where $\beta = \lceil \alpha.n \rceil$ and $\beta \leq n - 1$.*

Proof. Suppose there exists an (α, s) feasible $(n, n, 1, \frac{s+1+\beta-n}{n})$ cyclic GC scheme, thus each worker has access to exactly $v = s + 1 + \beta - n$ data subsets. Consider any 2 set of consecutive data subsets D_i and $D_{1+(i \mod n)}$. Choose a set of $s - 1$ consecutive workers from $W_{1+((i-s) \mod n)}$ to W_{i-1} and another worker W_{i+1} and straggle them. Since the master should be able to compute a sum of atleast β gradients from the results received from each worker except the set of s workers defined above, the coefficient of the gradients of data subsets D_i and $D_{1+(i \mod n)}$ transmitted by worker W_i have to be the same. This is because the master has access to exactly $\beta + 1$ gradients and the gradient of data subsets D_i and $D_{1+((i) \mod n)}$ is computed only by W_i amongst the set of non-straggling workers.

Using a very similar line of argument, we can show that the coefficient of the gradients of data subsets corresponding to D_i and $D_{1+(i \mod n)}$ transmitted by any other worker which has access to both of them must also be the same. This can be argued for every $i \in [n]$. This would imply that each worker just transmits the sum of all the gradients assigned to it as per the cyclic GC scheme discussed above.

Now suppose the set of non-straggling workers is denoted by $W_1, W_2, ..W_{n-s}$. Clearly under these set of workers the master would have access to exactly gradients of β data subsets. Suppose we denote the first row of the matrix A_i for

$i = 1, 2..., n-s$ as v_i. Since the master node should be able to compute the sum of the gradients of first β data-sets from transmissions by workers $W_1, W_2, ..W_{n-s}$, $v = [\underbrace{1,1,..1}_{\beta},\underbrace{0,0,..0}_{n-\beta}]$ must lie in the span of $\{v_i\}$. Also note that vector v_i has consecutive ones from position i to $(1 + ((i + r - 2) \mod n))$ for $r = s + 1 + \beta - n$ rest all zeroes.

Suppose $v = \sum_i c_i v_i$ for some $c_i \in \mathbb{R}$. This would imply that $c_1 = 1$,$c_2 = 0$,...,$c_{s+1+\beta-n} = 0$,$c_{s+2+\beta-n} = 0$,..$c_{2s+2+\beta-n} = 0$ and so on. More generally, $c_i = 1$ if $i \mod r = 1$ else 0 where $r = s + 1 + \beta - n$. Now we can substitute the $\{c_i\}$ in the equation $v = \sum_i c_i$ and observe that it can't be satisfied if $s + 1 + \beta - n$ does not divide β. Thus the master cannot recover the sum of β data subsets and hence such a cyclic GC scheme is not (α, s) feasible ☐

However, we can show that a cyclic (α, s) feasible cyclic $(n, n, 1, \frac{s+1+\beta-n}{n})$ GC scheme is always possible under a communication load of 2.

Theorem 6. *There exists an (α, s) feasible $(n, n, 2, \frac{s+1+\beta-n}{n})$ cyclic GC scheme with $\beta = \lceil \alpha.n \rceil$ for every n, s, α.*

Proof. If $s + 1 + \beta - n$ divides β, then the scheme described in proof of Theorem 4 achieves a communication load of 1. Else consider the following scheme described below.

We follow the assignment scheme as described in Definition V.1 and each worker is assigned exactly $s + 1 + \beta - n$ data subsets. Each worker transmits the sum of the gradients of all the data subsets assigned to it and the sum of first x data subsets assigned to it where x denotes the remainder when β is divided by $s + 1 + \beta - n$. For example worker W_1 transmits the sum of the gradients of the data subsets from D_1 to $D_{s+1+\beta-n}$ and the sum of the gradients of the data subsets from D_1 to D_x.

☐

Workers	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9
W_1	1	1	1						
W_2		1	1	1					
W_3			1	1	1				
W_4				1	1	1			
W_5					1	1	1		
W_6						1	1	1	
W_7							1	1	1
W_8	1							1	1
W_9	1	1							1

TABLE 3
ASSIGNMENT OF DATA SUBSETS TO DIFFERENT WORKERS IN $(\frac{7}{9}, 4)$ FEASIBLE $(9, 9, 2, \frac{3}{9})$ GC SCHEME

Example 3. *An example for $n = 9$, $\alpha = 7/9$ and $s = 4$ is described below. Note that $s + 1 + \beta - n = 3$ and the division of $\beta = 7$ by 3 gives remainder $x = 1$. The assignment of different data subsets to various workers can be described as follows: Note that each worker transmits the sum of the gradients of the data subsets it is assigned to and the first gradient computed by each. The assignment of data subsets to various workers has been shown in Table 3 For example, worker W_4 transmits the sum of the gradients of the data subsets D_4, D_5 and D_6 and the gradient of the data subset D_4. For example if workers W_2, W_4 and W_5 straggle, the master can still compute the sum of gradients of data subsets D_2, D_3, D_4 , D_6, D_7 D_8 and D_1 using the transmissions by the workers W_1, W_3 and W_6. We use transmission of the gradient of data subset D_1 by worker W_1, the sum of the gradients of D_3, D_4 and D_5 by worker W_3 and the sum of the gradients of D_6, D_7 and D_8 by worker W_6. Note that the scheme which just assigns only α fraction of data sets to the workers can be shown to have a lower bound on the max computation load per worker to be $\frac{\alpha(s+1)}{n} = 35/81$. Also the cyclic scheme as in [4] also has a max. computation load per worker to be $\frac{5}{9}$. Our cyclic GC scheme has a max. computation load per worker as $3/9$ with a communication load of 2 which is better than the two cyclic GC schemes described above. However we can achieve a smaller computation load using the scheme as described in Sec IV which would have a max. computation load per worker $\frac{y}{n} = 2/9$ though with a higher communication cost of $1 + \frac{y-1}{y}\binom{n-1}{y-1} = 5$.*

VI. (α, s)-FEASIBLE $(n, n, 1, l)$ GC SCHEMES UNDER LOW COMPUTATION LOAD

In this section, we consider the problem of partial gradient recovery under the restriction of $k = n$ data subsets, and focus on the regime with communication load $m = 1$ and a small computation load l. We begin with a simple lemma about the case of $l = 1/n$ which is the minimum possible computation load.

Lemma 2. *For any (α, s)-feasible $(n, n, 1, \frac{1}{n})$ GC scheme, we have $s \leq n - \beta$ for $\beta = \lceil \alpha.n \rceil$. Furthermore, there exists a simple $(\alpha, s = n - \beta)$-feasible $(n, n, 1, \frac{1}{n})$ GC scheme.*

Proof. For $l = 1/n$, each worker is assigned at most one data subset and thus when there are s stragglers, the master node can hope to recover the sum of the gradients corresponding to at most $n-s$ data subsets. Then from the definition of an (α, s)-feasible GC scheme, we have $\alpha.n \leq n - s$ which in turn implies $s \leq n - \lceil \alpha.n \rceil$. Finally, the trivial scheme which assigns a unique data subset to each worker node and each non-straggler node simply computes and transmits the corresponding gradient to the master node is indeed $(\alpha, n - \lceil \alpha.n \rceil)$-feasible. ☐

The next two results consider the impact of allowing for more stragglers on the computation load l.

Theorem 7. *For $\beta = \lceil \alpha.n \rceil$, consider any $(\alpha, s = n - \beta + 1)$ feasible $(n, n, 1, l)$ GC. Then the following hold true.*

- *If β is even and $\beta \leq n-1$, then $l \geq \frac{s+1+\beta-n}{n} = \frac{2}{n}$. Furthermore, there exists a cyclic scheme which achieves $l = \frac{2}{n}$.*
- *If β is odd and $\beta \leq n-1$, then $l > \frac{s+1+\beta-n}{n} = \frac{2}{n}$.*

Proof. The first half of the first statement follows from Theorem 1 by showing that inequality (1) is unsatisfied when $y = 1$. The existence of a cyclic scheme for even β with $l = 2/n$ can be shown using Theorem 4. Finally, the proof of the second part of the theorem can be found in Appendix E. ☐

Theorem 8. *For $\beta = \lceil \alpha.n \rceil$ and $\beta \leq n-1$, consider any $(\alpha, s > n - \beta + 1)$ feasible $(n, n, 1, l)$ GC. Then $l > \frac{(n-\beta+1)+1+\beta-n}{n} = \frac{2}{n}$.*

The proof of this result can be found in Appendix F.

VII. Discussion

For the exact gradient coding setup, it is known that the minimum computation load of $(s+1)/n$ and the minimum communication load of 1 can be achieved simultaneously [4]. For the partial gradient recovery framework discussed here, while we have shown that unlike exact gradient recovery there exists a trade-off between the computation and communication loads when restricting to cyclic schemes which use a number of data subsets equal to n, in general the question remains open. Also, we assume each transmission message to be of a fixed size (gradient dimension p) and define communication load as the number of messages transmitted by each worker. Allowing splitting of gradient vectors and coding across their components as done in [10] to reduce the individual message sizes is another direction for future work.

VIII. Acknowledgement

Nikhil Karamchandani acknowledges initial discussions on this problem with Bikash Kumar Dey and Chinmay Gurjarpadhye at Indian Institute of Technology, Bombay.

Note that for the proofs in the upcoming appendix, we denote $[n]$ as the set of consecutive integers from 1 to n. Also we denote the remainder when b divides a by $a\%b$ or $a \mod b$

Appendix A
Proof details of Claim 1

We restate and prove the claim 1 used in the proof of Theorem 1.

Claim. *Consider any collection of t positive integers $\{a_i\}_{1 \leq i \leq t}$. Define $a = \lfloor \frac{\sum_{i=1}^{t} a_i}{t} \rfloor$ and let t_1 be the unique positive integer satisfying $\sum a_i = t_1.a + (t-t_1)(a+1)$. Then we have $\sum_{i=1}^{t} \binom{a_i}{r} \geq t_1.\binom{a}{r} + (t-t_1).\binom{a+1}{r}$.*

Let us state the claim that we use to prove theorem 1.

Proof.

$$\binom{x+m_1}{r} - \binom{x}{r} = \sum_{i=1}^{m_1} \binom{x+i-1}{r-1}$$

$$\binom{x+1}{r} - \binom{x-m_2+1}{r} = \sum_{i=1}^{m_2} \binom{x-i+1}{r-1}$$

These follow from $\binom{n}{r} + \binom{n}{r+1} = \binom{n+1}{r+1}$. Since $\binom{x+i-1}{r-1} \geq \binom{x-j+1}{r-1}$ for any $0 \leq i \leq m_1$, $0 \leq j \leq m_2$, we can say $\frac{\binom{x+m_1+1}{r} - \binom{x+1}{r}}{m_1} \geq \frac{\binom{x+m_1}{r} - \binom{x}{r}}{m_1} \geq \frac{\binom{x+1}{r} - \binom{x-m_2+1}{r}}{m_2} \geq \frac{\binom{x}{r} - \binom{x-m_2}{r}}{m_2}$

Choose the list I as follows: $\{i \in [t] : a_i > a+1\}$ and list J as $\{i \in [t] : a_i < a\}$. Now choose a partition of I s.t $I = I_1 \cup I_2$ and $I_1 \cap I_2 = \Phi$ and J s.t $J = J_1 \cup J_2$ and $J_1 \cap J_2 = \Phi$ s.t $|I_1 \cup J_1 \cup \{i \in [t] : a_i = a+1\}| = t - t_1$. This would imply $|I_2 \cup J_2 \cup \{i \in [t] : a_i = a\}| = t_1$

Now denote $k_{min} = \min_{m_1>1} \frac{\binom{x+m_1}{r}-\binom{x}{r}}{m_1}$ and $k_{max} = \max_{m_2>1} \frac{\binom{x+1}{r}-\binom{x-m_2+1}{r}}{m_2}$, thus $k_{min} \geq k_{max}$
Thus,

$$\begin{aligned}
&\sum_{i\in I_1}\Big[\binom{a_i}{r} - \binom{a+1}{r}\Big] + \sum_{i\in I_2}\Big[\binom{a_i}{r} - \binom{a}{r}\Big] \\
\overset{(a)}{\geq} &\sum_{i\in I_1}\Big[\binom{a_i-1}{r} - \binom{a}{r}\Big] + \sum_{i\in I_2}\Big[\binom{a_i}{r} - \binom{a}{r}\Big] \\
\geq &k_{min}(\sum_{i\in I_1}(a_i-a-1) + \sum_{i\in I_2}(a_i-a))
\end{aligned}$$

$$\begin{aligned}
&\sum_{i\in J_1}[\binom{a+1}{r} - \binom{a_i}{r}] + \sum_{i\in J_2}[\binom{a}{r} - \binom{a_i}{r}] \\
\overset{(b)}{\leq} &\sum_{i\in J_1}[\binom{a}{r} - \binom{a_i-1}{r}] + \sum_{i\in J_2}[\binom{a}{r} - \binom{a_i}{r}] \\
\leq &k_{max}(\sum_{i\in J_1}(a-a_i+1) + \sum_{i\in J_2}(a-a_i))
\end{aligned}$$

Note (a) follows from the fact that $\binom{x+m_1+1}{r} - \binom{x+1}{r} \geq \binom{x+m_1}{r} - \binom{x}{r}$ and (b) follows using similar reasoning.
Now

$$\begin{aligned}
&\sum_{i} a_i = t_1.a + (t-t_1).(a+1) \\
\overset{(a)}{\implies} &\sum_{i\in I_1} a_i + \sum_{i\in I_2} a_i + \sum_{i\in J_1} a_i + \sum_{i\in J_2} a_i = a.(|I_2|+|J_2|) + (a+1).(|I_1|+|J_1|) \\
\implies &\sum_{i\in I_2}(a_i-a) + \sum_{i\in I_1}(a_i-a-1) = \sum_{i\in J_2}(a-a_i) + \sum_{i\in J_1}(a+1-a_1)
\end{aligned}$$

Note that (a) follows from the fact that $|I_1 \cup J_1 \cup \{i\in[t] : a_i = a+1\}| = t - t_1$ and $|I_2 \cup J_2 \cup \{i \in [t] : a_i = a\}| = t_1$
Since we prove previously that $k_{min} \geq k_{max}$, we argue that

$$\begin{aligned}
&\sum_{i\in I_1}\Big[\binom{a_i}{r} - \binom{a+1}{r}\Big] + \sum_{i\in I_2}\Big[\binom{a_i}{r} - \binom{a}{r}\Big] \geq \sum_{i\in J_1}[\binom{a+1}{r} - \binom{a_i}{r}] + \sum_{i\in J_2}[\binom{a}{r} - \binom{a_i}{r}] \\
\implies &\sum_{i\in I_1\cup I_2\cup J_1\cup J_2} \binom{a_i}{r} \geq |I_1+J_1|.\binom{a+1}{r} + |I_2+J_2|.\binom{a}{r} \\
\overset{(a)}{\implies} &\sum_{i}\binom{a_i}{r} \geq t_1.\binom{a}{r} + (t-t_1).\binom{a+1}{r}
\end{aligned}$$

Note (a) follows from the fact that $|I_1 \cup J_1 \cup \{i\in[t] : a_i = a+1\}| = t - t_1$ and $|I_2 \cup J_2 \cup \{i \in [t] : a_i = a\}| = t_1$ and definitions of I and J.

□

Appendix B
Proof details of Theorem 3

A. Construction

Let us re-index each data subset by the set of worker indices it is assigned to. For example, data subset D_{T_J} is assigned to workers indexed by set J. Consider all possible subsets containing 1 and order them in lexicographic order for example sets $\{\{1,2,3\},\{1,2,4\},\{1,2,5\},\{1,2,6\},\{1,3,4\},\{1,3,5\}\}$ are some subsets of cardinality 3 sorted in lexicographic order. Two sets I and J s.t $I, J \in [n]$ differ by a cyclic shift if $J = \{1 + (y+a-1)\%n | y \in I\}$ for some $a \in [n]$.

Consider the first distinct $\frac{1}{y}\binom{n-1}{y-1}$ subsets of cardinality y of $[n]$ with each subset containing 1 such that no two subsets differ by any cyclic shift. Let us denote this collection of sets by P_1. Now choose worker W_1 corresponding to all data subsets D_{T_J} $\forall J \in P_1$. Thus, in other words $t_{T_J} = 1$ $\forall J \in P_1$. Recall from the proof of Theorem 3 that worker W_{t_j} is chosen for data subset D_j such that t_j belongs to the indices of the j workers to whom data subset D_j is assigned.

Now let us define the set P_2. Increment each element in every subset of P_1 by 1 with rollover to 1 if crosses n to obtain P_2. Formally we denote $P_2 = \{\{1 + (u \mod n)|u \in J\}|J \in P_1\}$. Similarly we choose worker W_2 corresponding to all data subsets D_{T_J} $\forall J \in P_2$. In general, we define the subset P_x $\forall x \in [n]$ by increasing each element of P_1 by $x-1$ with rollover to 1 if the sum crosses n. Formally we denote $P_x = \{\{x - 1 + (u \mod n)|u \in J\}|J \in P_1\}$. Similarly we choose worker W_x corresponding to all data subsets D_{T_J} $\forall J \in P_x$.

Let us work out an example for the case of $n = 7$ and $y = 4$. Note that n and y are co-prime.

Example 4. *The distinct* $\binom{n-1}{y-1} = 20$ *subsets in lexicographic order containing 1 can be written as* $\{\{1,2,3,4\},\{1,2,3,5\},\{1,2,3,6\},\{1,2,3,7\},\{1,3,4,5\},...\{1,5,6,7\}\}$*. However observe that the subsets* $\{1,2,3,4\}$ *and* $\{1,2,3,7\}$ *differ by a cyclic shift of 6 and the subsets* $\{1,2,3,6\}$ *and* $\{1,3,4,5\}$ *also differ by a cyclic shift of* 5*. Thus we choose the set* P_1 *as* $\{\{1,2,3,4\},\{1,2,3,5\},\{1,2,3,6\},\{1,3,4,6\},\{1,3,4,7\}\}$*, similarly we define the set* P_2 *as* $\{\{2,3,4,5\},\{2,3,4,6\},\{2,3,4,7\},\{2,4,5,7\},\{2,4,5,1\}\}$ *by incrementing each element in each set of* P_1 *by 1 and thus set* P_7 *is defined as* $\{\{7,1,2,3\},\{7,1,2,4\},\{7,1,2,5\},\{7,2,3,5\},\{7,2,3,6\}\}$ *and hence we choose worker* W_1 *corresponding to data subsets which are assigned to the set of workers indexed by sets in* P_1*. Similarly we choose worker* W_2 *corresponding to data subsets which are assigned to the set of workers indexed by sets in* P_2 *and so till sets in* P_7*.*

B. Proof that the above construction works

Clearly no two subsets in any set P_i can be identical since they have been obtained by incrementing every element in distinct subsets in P_1 by $i-1$.

We first show that there can be no element in both P_i and P_j for $i \neq j$. Let us prove it by contradiction by assuming that there exists a subset J in both P_i and P_j. Suppose J was obtained in set P_i by shifting elements of subset A in P_1 by $i-1$ and J was obtained in set P_j by shifting elements of another subset B by $j-1$. Thus, subset B can be obtained from A by shifting each element of A by $j-i$ which is a contradiction since both A and B are consecutive distinct lexicographic subsets containing 1.

Let us consider the other case if subset J is obtained in set P_i and set P_j by shifting elements of the same subset A by $i-1$ and $j-1$ respectively. Thus shifting elements of subset J by $j-i$ gives the same subset. Consider the smallest element a such that shifting elements of J by $a-1$ gives the same subset J. Suppose the $(1+(t+r-1)\%y)^{th}$ element of J be equal to t^{th} element of the list obtained after shifting elements of J by a $\forall t \in [n]$. Let us denote the elements of J as $[j_1, j_2, ..j_y]$ where $j_1 < j_2 < .. < j_y$. This would imply that $\sum_{z=1}^{r} j_{1+(t+z-1)\%r} = a-1$ for every integer t i.e. any consecutive set of r elements element has the sum to be $a-1$.

Suppose y is not a multiple of r. Suppose not and say the remainder when r divides y is given by q, then we can argue that $\sum_{z=1}^{q} j_{1+(z+t-1)\%r}$ remains the same for all t clearly the sum of which is smaller than $a-1$, thus there would exist an integer smaller than a say b such that shifting elements of J by b gives the same subset J.

Thus y is a multiple of r say $y = r.m$, hence $n = \sum_{z=1}^{y} j_z = m.\sum_{z=1}^{r} j_z = m.(a-1)$ since any the sum of any set of r consecutive elements remain the same. Thus n and y have the same factor implying they are not co-prime.

Thus, we proved that no two elements of two distinct sets P_i and P_j can be the same. Since the sum of cardinalities of all subsets $\{P_i\}$ is $\frac{n}{y}\binom{n-1}{y-1} = \binom{n}{y}$ implying that each worker is chosen exactly the same no of times i.e $\frac{1}{y}\binom{n-1}{y-1}$ and there is a worker chosen for every data subset.

Appendix C
Correctness argument of the construction proposed in Theorem 4

Recall the construction from the proof of Theorem 4 where W_i contains the data subsets $D_i, D_{i+1}, ..D_{1+(i+r-2)\%r}$ where $r = s+1+\beta-n$. Let the workers denoted by $W_1, ..., W_\beta$ be grouped into r groups each group containing $\frac{\beta}{r}$ workers. Let us denote the groups by $\{\mathcal{A}_j\}_{j\in[r]}$. Suppose group $\mathcal{A}_j$ contains the $\frac{\beta}{r}$ workers $W_j, W_{j+r}, ...W_{j+\beta-r}$ which ensures that no one worker is present in two different groups. Note that worker denoted by W_i belongs in group $\mathcal{A}_{f(i)}$ where $f(i) = 1 + (i-1)\%r$.

Consider the set of straggling workers be denoted by S s.t $|S| = s$. Suppose there exists a group $\mathcal{A}_i$ with no straggling worker present, this would imply the existence of $\frac{\beta}{r}$ workers with disjoint set of data subsets implying the master would be able to calculate the sum of β gradients from the results computed by the non-straggling workers.

Suppose there does not exist a group $\mathcal{A}_i$ without any straggling worker present in it i.e each group has at least one straggling worker present.

Note that worker W_i returned from Alg 1 would have no worker W_j in $\mathcal{A}_{f(i)}$ s.t $j < i$ and $W_j \in S$.

Now consider the entire set of groups visited in the algorithm described as I. We can argue if $I = x+1$, there must exist at least x workers W_j satisfying $j < i$ and $W_j \in S$ (at least one worker from each of the x rows). Also note that there must be at least one straggling worker corresponding to each of $(r-x-1)$ groups which were not visited in the

Algorithm 1: Stopping Straggler 1

Choose largest $i \in [\beta]$ s.t $i \in S$.
while $\exists\ v < i\ s.t\ W_v \in \mathcal{A}_{f(i)}$ *and* $W_v \in [S]$ **do**
    Choose largest $j < i$ s.t $W_j \in S$.
    i =j
end
Output i.

algorithm since each group has at least one straggling worker. Note that each of these workers must have its index smaller than i, thus there would exist at least $(r-x-1)+x = r-1$ workers behind W_i.

Now consider worker W_i in group $\mathcal{A}_{f(i)}$ and suppose we have m workers W_j s.t. $j \geq i$ and $W_j \in \mathcal{A}_{f(i)}$. Suppose we denote all the workers in group $\mathcal{A}_{f(i)}$ as $W_{i_1}, ..., W_{i_{\frac{\beta}{r}}}$, thus $i = i_{\frac{\beta}{r}-m+1} = f(i) + (\frac{\beta}{r}-m)r$

We now claim there must exist at least a vector of integers $[k_1, k_2, ..., k_m]$ with these indices lying in the set of indices of non-straggling workers satisfying

$$\begin{aligned} & k_t - k_{t-1} \geq r \forall i \in [m-1], \ \ , k_1 \geq i \\ & k_m \leq n + f(i) - r \end{aligned} \tag{3}$$

Note that the conditions mentioned above would ensure that no overlap between the data subsets assigned to workers $W_{i_1}, W_{i_2}, ..., W_{i_{\frac{\beta}{r}-m}}, W_{k_1}, ..., W_{k_m}$.

The minimum size of set Z s.t there is no solution of $\{k_v\}_{v\in[m]}$ satisfying $k_v \notin Z \forall v \in [m]$ and (3) is given by $(n+f(i)-r) - i - (m-1).r + 1 = n + f(i) - m \times r - i + 1 = n + f(i) - m \times r - f(i) + (\frac{\beta}{r} - m)r + 1 = n - \beta + 1$

However, the number of straggled workers W_j s.t $j \geq i$ is at most $s - r + 1 = n - \beta$ which would imply that there exists at least a vector $[k_1, ..., k_m]$ satisfying (3), thus proving the existence of a set of $\frac{\beta}{r}$ non-straggling workers such that the data subsets assigned to them don't overlap.

We give two examples of sets of stragglers to demonstrate the proof strategy mentioned above and describe two sets of stragglers for $n = 18$ and $\beta = 15$ and $s = 7$

Example 5. *Recall that the set of workers is given by $W_1, W_2, ..., W_{18}$. Suppose the set of straggling workers is denoted W_{15}, W_{13}, W_{10}, W_8, W_4, W_{12} and W_{11}. Note that the group $\mathcal{A}_1$ would contain the workers W_1, W_6 and W_{11}, the group $\mathcal{A}_2$ would contain the workers W_2, W_6 and W_{12} and so on and finally the group $\mathcal{A}_5$ contains the workers W_5, W_{10} and W_{15}. First it is important to note that each group has at least one straggling worker.*

Note that the largest indexed straggling worker amongst the first fifteen is 15. Thus i is initialised to 15. However there is a straggling worker with a smaller index 10 in the same group $\mathcal{A}_5$, hence we set i get to 13. Again, since there is another straggling worker with index 8 in the group $\mathcal{A}_3$, we set i as 12. However, since there is no straggling worker with smaller index in group $\mathcal{A}_2$, we return 12 Note that W_{12} has exactly 4 stragglers with indices smaller than 12 none of them being in the group. Note that m is 1 in this case and we can choose k_1 as 14 (belonging to set of non-straggling workers) satisfying (3). Thus we obtain the can use the transmissions by workers W_2,W_7 and W_{14} to obtain the sum of 15 gradients.

Example 6. *Recall that the set of workers is given by $W_1, W_2, ..., W_{18}$. Suppose the set of straggling workers is denoted W_{14}, W_{13}, W_9, W_8, W_6, W_7 and W_5. Note that the group $\mathcal{A}_1$ would contain the workers W_1, W_6 and W_{11}, the group $\mathcal{A}_2$ would contain the workers W_2, W_6 and W_{12} and so on and finally the group $\mathcal{A}_5$ contains the workers W_5, W_{10} and W_{15}. Also observe that each group has at least one straggling worker.*

Note that the largest indexed straggling worker amongst the first fifteen is 14. Thus i is initialised to 14. However there is a straggling worker with a smaller index 9 in the same group $\mathcal{A}_4$, hence we set i get to 13. Again, since there is another straggling worker with index 8 in the group $\mathcal{A}_3$, we set i as 8 as there is no straggling worker with index larger than 8. Note that W_8 has exactly 4 stragglers with indices smaller than 8 with none of them being in the group $\mathcal{A}_3$. Note that m equals 2 in this case and we can choose $k_1 = 10$ and $k_2 = 15$ (belonging to set of non-straggling workers) satisfying (3). Thus we obtain the can use the transmissions by workers W_3,W_{10} and W_{15} to obtain the sum of 15 gradients.

Appendix D
Correctness argument of the construction proposed in Theorem 6

Recall that worker W_i contains the data subsets $D_i, D_{i+1}, ..D_{1+(i+r-2)\%r}$ where $r = s + 1 + \beta - n$. Note that we denote the remainder by x when r divides n. Let the workers denoted by $W_1, ..., W_\gamma$ be grouped into r groups each group containing $\frac{\gamma}{r}$ workers where $\gamma = \beta - x$ which clearly divides r. Let us denote the groups by $\{\mathcal{A}_j\}_{j\in[r]}$. Suppose group $\mathcal{A}_j$ contains the $\frac{\gamma}{r}$ workers $W_j, W_{j+r}, ...W_{j+\gamma-r}$ which ensures that no one worker is present in two different groups. Note that worker denoted by W_i belongs in group $\mathcal{A}_{f(i)}$ where $f(i) = 1 + (i-1)\%r$.

Consider the set of straggling workers be denoted by S s.t $|S| = s$.

Algorithm 2: Stopping Straggler 2

Choose largest $i \in [\gamma]$ s.t $i \in S$.
while $\exists\ k < i\ s.t\ W_k \in \mathcal{A}_{f(i)}\ and\ W_j \in [S]$ **do**
  Choose largest $j < i$ s.t. $W_j \in S$.
  i=j
end
Output i.

Note that worker W_i returned from the Algorithm 2 would have no worker W_j in $\mathcal{A}_{f(i)}$ s.t $j < i$ and $W_j \in S$.

Now consider the entire set of groups visited in the algorithm described as I. We can argue if $I = z + 1$, there must exist at least z workers W_j satisfying $j < i$ and $W_j \in S$ (at least one worker from each of the x rows). Suppose there exists m workers W_j s.t $j \geq i$ and $W_j \in \mathcal{A}_{f(i)}$ and we denote all the workers in group $\mathcal{A}_{f(i)}$ as $W_{i_1}, ..., W_{i_{\frac{\gamma}{r}}}$, thus $i = i_{\frac{\gamma}{r}-m+1} = f(i) + (\frac{\gamma}{r} - m)r$. We consider two cases i.e. when each group $\{\mathcal{A}_i\}$ has at least one-straggling worker and when there exist groups without any straggling worker.

Case I: There does not exist any group without any straggling worker.

Since the algorithm visited exactly $(z+1)$ distinct groups implying the existence of exactly $(r-z-1)$ groups which are not visited each of which must have at least one straggling worker with index smaller than i. Thus there exist at least $(r-z-1)+z = r-1$ workers with index smaller than i. Now consider the smallest index u larger than i such that W_u is a non-straggling worker.

We now claim there must exist at least a vector $[k_1, k_2, ..., k_m]$ satisfying

$$\begin{aligned} &k_t - k_{t-1} \geq r,\ \ , k_1 \geq u + x \\ &k_m \leq n + f(i) - r \end{aligned} \tag{4}$$

Note that the conditions mentioned above would ensure that no overlap between the sum of all the r data subsets transmitted by the workers $W_{i_1}, W_{i_2}, ..., W_{i_{\frac{\gamma}{r}-m}}, W_{k_1}, ..., W_{k_m}$ and the data subset of first x gradients transmitted by the worker W_u. Note that a key difference in this approach is that the data subsets assigned to workers $W_{i_1}, W_{i_2}, ..., W_{i_{\frac{\gamma}{r}-m}}, W_{k_1}, ..., W_{k_m}$ and W_u may overlapping unlike the proof of Theorem 4.

Recall from the definition of γ that $\gamma = \beta - x$.

The minimum size of set Z s.t there is no solution of $\{k_v\}_{v\in[m]}$ satisfying $k_v \notin Z \forall v \in [m]$ and (4) can be shown to be $n + f(i) - r - (u+x) - (m-1)r + 1 = n + f(i) - (u+x) - m \times r + 1 = (n-u) + (i-\beta) + (f(i) - i + \gamma) - m \times r + 1 = (n-u) + (i-\beta) + 1$.

However note that since W_u is the smallest index non-straggling worker larger than i, there are at least $u-i$ straggling workers from W_i to W_u. However, since the total number of stragglers starting from W_{u+x} is $(s-(r-1)-(u-i) = -\beta + n - u + i$ which is clearly smaller than the minimum size needed to ensure no solution of (4).

Thus we can recover the desired sum of β gradients from the transmission of sum of r gradients by the workers $W_{i_1}, W_{i_2}, ..., W_{i_{\frac{\gamma}{r}-m}}, W_{k_1}, ..., W_{k_m}$ and the sum of x gradients by worker W_u

Case -II: There exist groups $\{\mathcal{A}_j\}_{j\in[r]}$ with no straggling workers in any of these groups.

Case II-(a): The largest and the smallest indices of groups without any straggling worker differ by at least x.

Under this assumption, we can argue that there would exist a set of $\frac{\gamma}{r} + 1$ workers such that the first x data-subsets of the one worker would be non-overlapping with all the data subsets of the other $\frac{\gamma}{r}$ workers.

Case II-(b): The largest and the smallest indices of such groups without any straggling worker differ by a quantity smaller than x.

Suppose the set of all such indices is denoted by J with $|J| = t$. Since it is a cyclic scheme and cyclic shifts in data subsets assigned to workers don't make any difference, we assume the smallest and largest index of J to be 1 and a where $a < x + 1$.

Now consider $B_w = \mathcal{A}_w \cup \{W_{w+(\gamma)}\} \forall w \in J$ and $w \geq r - x$ Note that since the largest and the smallest index of J differ by a quantity smaller than x, there can be any common worker between two distinct subsets B_i and B_j $i, j \in J$. Also note that $w + \gamma \leq n$ since $w \leq x$ and $n \geq \beta$.

Suppose there exists $w \in J$ s.t no worker in B_w straggles, then choose the master can recover the sum of β gradients by the sum of r gradients of workers $\mathcal{A}_w$ and the sum of first x gradients transmitted by $W_{w+\gamma}$.

Suppose there does not exist any worker $w \in J$ s.t no worker in B_w straggles. In this case, each worker in $W_{w+(\gamma)} \forall w \in J$ must straggle as the other workers in $\mathcal{A}_w \forall w \in J$ don't straggle. Choose the largest element in J which has been assumed to be a in this case and thus consider the worker $W_{a+\gamma}$. The worker with index i has clearly smaller index than all workers in $W_{w+(\gamma)} \forall w \in J$. Also we know there are at least $(r-z-1)+(z-t)=(r-t-1)$ straggled workers with indices smaller than i. Thus we have $(r-t-1)+1+(t-1)=r-1$ straggled workers with indices smaller than $a+\gamma$. We denote the workers in group $\mathcal{A}_a$ as $\{W_{a_1}, W_{a_2}, ..., W_{a_{\frac{\gamma}{r}}}\}$ Using a very similar argument as in Case D, we can show that there would exist z satisfying $z - a_{\frac{\gamma}{r}} \geq r$ and $z \leq n+a-x$ with W_z being a non-straggling worker. This can be argued from the fact the number of straggling workers with indices larger than or equal to $a_{\frac{\gamma}{r}}$ is at most $(s-r+1)=(n-\beta)$ since the number of stragglers with index less than $a_{\frac{\gamma}{r}}$ is at most $r-1$. Thus we can recover the desired sum of β gradients from the transmission of sum of r gradients by the workers $W_{a_1}, W_{a_2}, ..., W_{a_{\frac{\gamma}{r}}}$ and the sum of x gradients by worker W_z.

Appendix E
Proof of Theorem 7

Proof. Let us prove by contradiction. Suppose we have $l = \frac{2}{n}$ i.e. every worker transmits and computes a linear combination of gradients of at most two data subsets.

We represent every data subset as a node in the graph. Since every worker can be assigned at most 2 data subsets, we denote it as an edge between the two corresponding nodes if it indeed transmits a linear combination of two data subsets; else we represent it as a self-loop around the node corresponding to the data subset assigned to it. We divide the problem into two cases (described below) and prove it for each case. Also note that the second case is the most general case.

- We assume a uniform distribution of two data subsets to every worker with each data subset being assigned to at exactly two workers. Thus, the graph consists of disconnected components with each component being a cycle.
- We assume the graph consists of one or more disconnected components.

Case-I: The graph consists of disconnected component with each component being a cycle.

Since every gradient is being computed by two workers, each node must be present in two edges. Also note that we have n edges and n nodes, thus the graph must be comprised of disjoint cyclic components with each component being a cycle. There could be a pair of isolated nodes with a pair of edges connecting them which we treat as a connected cyclic component only like the component C in the Fig. 1.

Let us denote the sizes(vertices/edges) of the components by $c_1, c_2, ...c_t$ if there are t such components with $\sum c_t = n$ W.L.OG, we assume $c_1 \leq c_2 \leq c_3 ... \leq c_t$. Let p denote the smallest index such that $\sum_{i=1}^{p} c_i \geq s+1$.

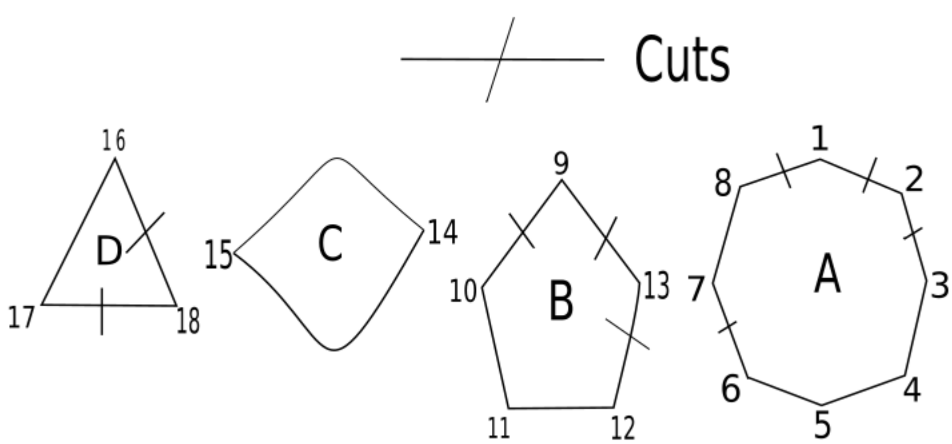

Fig. 1. Representation as data subsets as nodes with numbers denoting nodes and letters denoting components

Note that the lines on edges denoting the straggling workers in the diagram.

We state the following claim and use it for the proof of the theorem.

Claim 2. *If the above gradient code is $(\alpha, s = n - \beta + 1)$ feasible where $\beta = \lceil n\alpha \rceil$, then each worker must transmit the sum of the gradients of the data subsets assigned to it,*

Proof. We divide the proof into two parts - for edges in cycles with size larger than or equal to s and cycles with size smaller than s.

Case (a): Consider any edge in some component with size larger than or equal to $s + 1$.

For simplicity, we denote the size of this component by x and the node of this component is $A_1, A_2, ..., A_x$ and the edges are $A_1 - A_2, A_2 - A_3, ...A_x - A_1$. We assume the edge corresponding to this worker as $A_1 - A_2$. Now we consider the case when workers corresponding to edges $A_x - A_1$,$A_2 - A_3$, $A_3 - A_4$,...,$A_s - A_{s+1}$ straggle- for example the cut in component A. In this case clearly the gradients to data subsets $A_3, .., A_s$ won't be accessible to the master at all, thus these $s - 2$ gradients cannot be present in the sum computed by the master, thus the master has access to at most $(\beta + 1)$ gradients with data subsets corresponding to A_1 and A_2 being present in exactly one non straggling worker. Since the master has to compute the sum of at least β gradients, the coefficient of the gradients corresponding to data subsets A_1 and A_2 in the worker which transmits their linear combination has to be the same.

Case (b) : Consider an edge in some component r of size smaller than or equal to s.

For simplicity, we denote the size of this component by x and the node of this component is $A_1, A_2, ..., A_x$ and the edges are $A_1 - A_2, A_2 - A_3, ...A_x - A_1$.

Let p_{min} be the minimum index of i such that $\sum_{j=1;j\neq r}^{i} c_i \geq s - x + 1$ and $p_{last} = s - x + 1 - \sum_{j=1;j\neq r}^{p_{min}-1} c_i$.

Thus, we straggle the following s workers-

- All workers in the first $p_{min} - 1$ components excluding component numbered r.
- Exactly p_{last} continuous edges of component c_i like the cut shown in component B.
- All edges except $A_1 - A_2$ in component numbered r example - component D in the diagram.

Now we can observe that the master won't have access to $s - x + x - 2 = s - 2$ gradients if the above set of s workers straggle. Thus the master has access to at most $(\beta + 1)$ gradients with data subsets corresponding to A_1 and A_2 being present to only one non-straggling worker. Since the master has to compute the sum of at least β gradients, the coefficient of the gradients corresponding to data subsets A_1 and A_2 in the worker which transmits their linear combination has to be the same.

Thus, we prove that the coefficients of the gradient of each data subset in each worker must be the same without assuming β is odd. ☐

However, we show that if β is odd, this construction cannot yield the desired sum of the gradients at the master.

We know that the sizes of the components are $c_1, c_2, ..., c_t$ with $\sum_t c_t = n$ and choose $d_1, d_2, ..., d_n$ such that $\sum_t d_t = s$ with $d_i = c_i \forall 1 \leq i < m$ for some m, $d_m \leq c_m$ and $d_i = 0\ \forall i \geq m + 1$. The selection of stragglers show that the master won't have access to at least $s - 1$ gradients. None of the gradients corresponding to any of data subsets present in any of the first $m - 1$ workers can be computed by the master.

Suppose $c_m - d_m$ is odd, then there there must exist some $i \geq m + 1$ s.t $(c_i - d_i)$ is odd as $\sum_{i\geq m+1}(c_i - d_i) = \beta - 1 - c_m - d_m$ which is odd. Thus, we assume c_j to be odd for some $j \geq m+1$ as $d_j = 0 \forall j \geq m+1$. Now we decrease d_m by 1 and set d_j to 1. Note that this ensures that $\sum d_t$ remains s. Now let us define the s workers which straggle. All the workers corresponding to the edges in the first $m - 1$ components straggle and a set of continuous d_m edges in component numbered m straggle and continuous d_j edges in component numbered j straggle.

Consider component numbered m. The workers which don't straggle correspond to a set of continuous $c_m - d_m$ edges which is even and hence only a sum of $c_m - d_m$ gradients could be obtained corresponding to that cycle. An example demonstrating the fact is shown in Fig. 2 and Fig. 3. In Fig. 2, if each worker transmits the sum of data-subsets assigned to it, we recover all the sum of gradients of data subsets spanned by it through transmissions from workers W_1 and W_3. However in Fig. 3, if each worker transmits the sum of data-subsets assigned to it, we cannot recover all the sum of gradients of all data subsets spanned by it through transmissions. We can at most recover the sum of gradients of data subsets A_1 and A_2 or A_2 and A_3 since the number of workers is even.

Similarly in component numbered j, there are exactly $c_j - d_j$ workers corresponding to a set of continuous edges which don't straggle. Either the workers which don't straggle form the entire set of edges in the cycle in which case we obtain the sum of gradients corresponding to all the data subsets in the cycle, or the workers which don't straggle form a continuous set of even edges potentially excluding one edge in the cycle in which case only a sum of $c_t - d_t$ gradients corresponding to the data subsets in the cycle can be computed.

For any other cycle numbered $i > m$ if no worker is straggled we can obtain the sum of gradients of all the data subsets representing the nodes in the cycle i.e. a sum of gradients of $c_i - d_i = c_i$ data subsets.

Thus, we could obtain a sum of gradients of exactly $\sum_{i=m+1}^{t} c_i - d_i = n - s = \beta - 1$ data subsets which contradicts the requirement that the master should compute a sum of gradients of at least β data subsets.

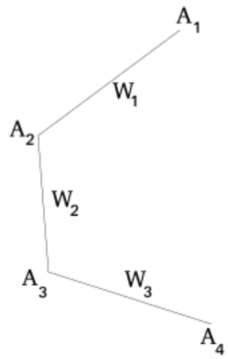

Fig. 2. A continuous set of odd number of edges

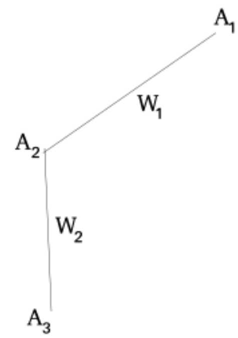

Fig. 3. A continuous set of even number of edges

Case II: Suppose the above graph is composed of t distinct disconnected components.

Suppose the edges of acyclic components is given by set E_1.We denote an order of removing edges in E_1 such that no new disconnected component is created at an step. Note that such an ordering can be ensured if we remove edges starting from a leaf node. Also observe that since no new disconnected component is created and only edges in acyclic components are removed, removal of t edges would ensure at most $n-t$ nodes in its span since no new disconnected component is created in the process.

Now after these edges are removed, consider the largest set of edges (and self-loops) (denoted by set E_2) that can be removed so that each component has at least one cycle. Note that these edges would be removed in order such that no new disconnected component with one or more isolated edges is created in the process of removal of edges thus, ensuring at most $n-t$ nodes in the span of $n-t$ remaining edges in every step. Thus, after removal of $|E_1|+|E_2|$ edges, we would have distinct components with each component being a cycle.

Note that cuts on edges Fig. 4 denotes the edges in sets E_1 and E_2 in order described above with $n=28$. Also E_1 corresponds to edges numbered 1-7 and E_2 corresponds to edges numbered 8-14. Also note at no stage while removing edges in the order as numbered is a new disconnected component created with one or more isolated edges.

Thus, the maximum number of edges (or self-loops) that can be removed without reducing the number of cycles in the graph is denoted by $c_{max}=|E_1|+|E_2|$. Note that after removal of $|E_1|+|E_2|$ edges, we would have distinct components with each component being a cycle. Let us now consider three different conditions on s(maximum number of stragglers) and prove it in each case.

Case II-(a): $s \leq c_{max}=|E_1|+|E_2|$

Since the process of removing edges in each step ensures that exactly $n-t$ nodes lie in the span of remaining $n-t$ edges, we can just straggle the workers corresponding to the first s edges in the process described above and argue that exactly $n-s=\beta-1$ data-subsets are accessible to the master, thus leading to a contradiction.

Note that if we start straggling the workers corresponding to the edges in order ensure that no new disconnected component is created after the straggling workers are removed from the graph. Thus, removal of v edges would ensure at most $n-v$ nodes in the span of remaining edges. This would follow from the fact that each component continues to have exactly one cycle. Such an ordered removal of edges can be done if we remove edges along a path starting from a leaf node to a cycle.

Hence, if the total number of straggling workers is less than or equal to the maximum number of edges that can be removed without affecting any cycle, we can argue that no more than $n-s=\beta-1$ nodes can lie in its span, thus implying the master cannot compute the sum of β gradients from the set of non-straggling workers.

Case II-(b): $s \geq c_{max}+2=|E_1|+|E_2|+2$

Under this constraint we first straggle all the edges corresponding to $E_1 \cup E_2$ to obtain a similar structure of distinct cycles as the situation in Case E. We now proceed in a very similar way as in Case E and show that there exist a set of stragglers such that the master cannot compute the sum of any set of β gradients. Note that, we require $s \geq c_{max}+2=|E_1|+|E_2|+2$ since the previous case also assumes at least 2 stragglers since $s=n-\beta+1 \geq 2$ with $\beta \leq n-1$.

Case II-(c): $s=c_{max}+1=|E_1|+|E_2|+1$

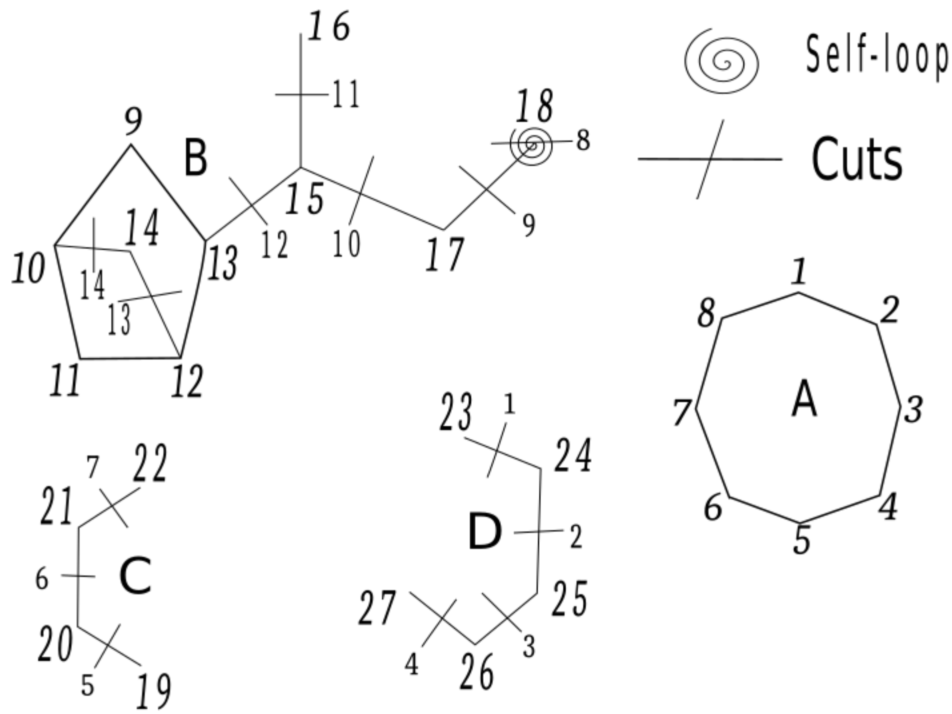

Fig. 4. Representation as data subsets as nodes with numbers denoting nodes and cuts denoting removal of edges

Suppose there are more than 2 vertices in a cycle into which paths from leave nodes branch into. Well in this case, actually we can show that all workers corresponding to every edges in the cycle would have the same coefficient for both the data subsets. Suppose A_1, A_2, A_3 be three consecutive nodes in the cycle and from each node there exists a cyclic path to a leaf node. Now we straggle the entire path to a leaf node from A_2, the edge $A_p - A_1$ (assuming p nodes in the cycle) and edge $A_2 - A_3$, the entire path from a leaf node to A_3 except the edge which connects it to the cycle. Using these set of stragglers we can show that the worker corresponding to the edge $A_1 - A_2$ has same coefficient for both the data subsets. Similarly, we can argue that the workers corresponding to all edges have the same coefficient for both the workers and a similar selection of stragglers will show that no sum of β gradients exists in the linear span of non-straggling workers.

Suppose there are at most two vertices in the cycle into which paths from leaf nodes merge into. In this case, there might be at most two edges in the cycle the workers corresponding to which may not have the same coefficient for the gradients of the data subsets assigned to them.

Suppose there exists only a path to node A_1 in the cycle to a leaf node of length c_{max}, thus in this example we can show that all workers except those corresponding to edges $A_1 - A_2$ and $A_n - A_1$ have the same coefficient for both the gradients of the data subsets assigned to it.

Now if p(the edges in cycle) is odd, straggle the workers corresponding to the edges in the path from A_1 to leaf node and the edge $A_1 - A_2$. We can show that since an even number of consecutive edges remain in the component after straggling the edges we cannot have a sum of all the gradients of the data subsets contained in the span of the edges corresponding to non-straggling workers. Since β data subsets remain in the span after the selection of straggling workers, we cannot have any sum of β gradients in its span.

Suppose p is even, straggle the workers corresponding to the edges in the path from A_1 to leaf node and the edge $A_1 - A_2$. Now the number of edges remaining in this component is odd which would imply that there must exist some other non-straggling cyclic component with odd number of edges as the total number of non-straggling edges is $n - s = \beta - 1$ which is even. Also there must exist a cyclic component after removal of c_{max} edges which has an odd number of edges. Thus instead of straggling node $A_1 - A_2$, straggle a node in another cyclic component with odd number of edges and the same argument as above follows. Similarly, we can argue for the other cases too if two nodes in a cycle have paths from leaf nodes branching into it.

□

Appendix F
Proof of theorem 8

Proof. Let us prove by contradiction. Suppose we have $l = \frac{2}{n}$ i.e. every worker transmits and computes a linear combination of gradients of at most two data subsets.

We represent every data subsets as a node in the graph. Since every worker can be assigned at most 2 data subsets, we denote it as an edge if it indeed transmits a linear combination of two data subsets, else we represent it a self-loop around a node corresponding to the data subset assigned to it.

Suppose the graph is represented in t distinct components and suppose u components have no cycle in it. There might be some components with multiple cycles in it too. Now we define an order of straggling workers. First start with those components which don't have any cycle and straggle the workers corresponding to those edges which start from a leaf edge and straggle workers continuously along a path. Note that this process would ensure that no new component is created when the edges corresponding to straggling workers are removed from the graph, thus at most $n-t$ nodes in the span when t edges are removed. Now consider the maximum number of edges that can be removed (corresponding workers straggled) such that each cyclic component continues to have at least one cycle. Note that we straggle these edges in order such that no new component is created in the process of removing edges from the graph. This would ensure that there is at most $n-t$ nodes in the span of remaining edges when t workers are straggled (removed) from the graph. At the end, we would have only cyclic components remaining. Straggle the edges in each cycle continuously till no edge in a cycle is left and then start with the next cycle. We stop when there is no worker is left.

Note that in this process of straggling, when $n-v$ edges corresponding to workers are remaining we can have atmost $n-v+1$ vertices or data subsets being spanned. This can be argued from the fact that unless we break a new cycle a data subset is always removed from the span whenever a worker is straggled and no distinct component of the graph is being created in the process of straggling (removing) workers. Thus straggling of s workers under the above mentioned process would ensure at most $n-s+1 \leq \beta - 1$ data subsets being accessible at the master implying a contradiction.

□

References

[1] S. Dutta, V. Cadambe, and P. Grover, "Short-dot: computing large linear transforms distributedly using coded short dot products," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016, pp. 2100–2108.

[2] 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.

[3] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Polynomial codes: an optimal design for high-dimensional coded matrix multiplication," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 4406–4416.

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

[5] Q. Yu, S. Li, N. Raviv, S. M. M. Kalan, M. Soltanolkotabi, and S. A. Avestimehr, "Lagrange coded computing: Optimal design for resiliency, security, and privacy," in *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019, pp. 1215–1225.

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

[7] 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, 2017.

[8] N. Raviv, I. Tamo, R. Tandon, and A. G. Dimakis, "Gradient coding from cyclic mds codes and expander graphs," *IEEE Transactions on Information Theory*, vol. 66, no. 12, pp. 7475–7489, 2020.

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

[10] M. Ye and E. Abbe, "Communication-computation efficient gradient coding," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5610–5619.

[11] E. Ozfatura, D. Gündüz, and S. Ulukus, "Speeding up distributed gradient descent by utilizing non-persistent stragglers," in *2019 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2019, pp. 2729–2733.

[12] E. Ozfatura, S. Ulukus, and D. Gündüz, "Straggler-aware distributed learning: Communication–computation latency trade-off," *Entropy*, vol. 22, no. 5, p. 544, 2020.

[13] H. Wang, S. Guo, B. Tang, R. Li, and C. Li, "Heterogeneity-aware gradient coding for straggler tolerance," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 555–564.

[14] K. Wan, H. Sun, M. Ji, and G. Caire, "Distributed linearly separable computation," *arXiv preprint arXiv:2007.00345*, 2020.

[15] S. Dutta, G. Joshi, S. Ghosh, P. Dube, and P. Nagpurkar, "Slow and stale gradients can win the race: Error-runtime trade-offs in distributed sgd," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2018, pp. 803–812.

[16] J. Chen, X. Pan, R. Monga, S. Bengio, and R. Jozefowicz, "Revisiting distributed synchronous sgd," 2017.

[17] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.

[18] H. Wang, Z. Charles, and D. Papailiopoulos, "Erasurehead: Distributed gradient descent without delays using approximate gradient coding," *arXiv preprint arXiv:1901.09671*, 2019.

[19] R. Bitar, M. Wootters, and S. El Rouayheb, "Stochastic gradient coding for straggler mitigation in distributed learning," *IEEE Journal on Selected Areas in Information Theory*, vol. 1, no. 1, pp. 277–291, 2020.
[20] R. K. Maity, A. S. Rawa, and A. Mazumdar, "Robust gradient descent via moment encoding and ldpc codes," in *2019 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2019, pp. 2734–2738.
[21] S. Wang, J. Liu, and N. Shroff, "Fundamental limits of approximate gradient coding," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 3, no. 3, pp. 1–22, 2019.
[22] S. Kadhe, O. O. Koyluoglu, and K. Ramchandran, "Gradient coding based on block designs for mitigating adversarial stragglers," in *2019 IEEE International Symposium on Information Theory (ISIT)*, 2019, pp. 2813–2817.
[23] Z. Charles, D. Papailiopoulos, and J. Ellenberg, "Approximate gradient coding via sparse random graphs," 2017.
[24] E. Ozfatura, S. Ulukus, and D. Gündüz, "Distributed gradient descent with coded partial gradient computations," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3492–3496.
[25] E. Ozfatura, S. Ulukus, and D. Gunduz, "Coded distributed computing with partial recovery," *arXiv preprint arXiv:2007.02191*, 2020.