Variable Coded Batch Matrix Multiplication

Lev Tauz, Debarnab Mitra, and Lara Dolecek

Department of Electrical and Computer Engineering, University of California, Los Angeles, USA

levtauz@ucla.edu, debarnabucla@g.ucla.edu , dolecek@ee.ucla.edu

***Abstract*—We introduce the novel Variable Coded Distributed Batch Matrix Multiplication (VCDBMM) problem which generalizes many previous coded distributed matrix multiplication problems by allowing for matrix products to re-use matrices, thus creating natural redundancy in the system. Inspired in part by Cross-Subspace Alignment codes, we develop Flexible Cross-Subspace Alignment (FCSA) codes that are flexible enough to utilize this natural redundancy and provide a full characterization of FCSA codes which allow for a wide variety of system complexities including good straggler resilience and fast decoding. We theoretically demonstrate that, under certain practical conditions, FCSA codes are within a factor of 2 of the optimal solution in terms of straggler resilience. Furthermore, simulations demonstrate that our codes can achieve even better optimality gaps in practice, even going as low as 1.7.**

I. Introduction and System Model

Large scale distributed matrix-matrix multiplication is a fundamental component of modern data analytics and is used to deal with the exponential rise of big data. The key idea is that a system of workers each store a fraction of the total data and perform local matrix multiplication that is then compiled at a fusion node. While having a large number of workers allows for theoretically better parallelization gains, the presence of *stragglers* (i.e., workers that fail or are slow to respond) significantly hampers distributed systems due to the increase in tail latency [1]. Stragglers can arise for a variety reasons such as variable communication delay due to a shared network or heterogeneity in processing time due to different life cycles of storage mediums (such as non-volatile memories) used at each worker. The major issue with stragglers is that many distributed tasks are designed such that all worker results are required to get the final result which is clearly slowed down by stragglers. To mitigate stragglers, coded computation is a methodology that injects computational redundancy through the use of error-correcting codes which allows for the final result to be extracted from a subset of the worker results [2]–[5].

Codes for matrix-matrix computations are broadly separated into two problem spaces: i) matrix partitioning for computing a single computation task [2], [3], [6], and ii) batch processing of multiple distinct computation tasks [4], [5]. We can summarize one example of problem space (i) as using coding to determine the product $\mathbf{AB}$ by coding across the row partition $\mathbf{A}^T = (\mathbf{A}_i^T)_{i=1}^n$ and the column partition $\mathbf{B} = (\mathbf{B}_i)_{i=1}^m$ to determine $(\mathbf{A}_i\mathbf{B}_j)_{i=1,j=1}^{n,m}$ [2] (note that the partitions are matrices). Similarly, we summarize problem space (ii) by a system that receives two lists of matrices $(\mathbf{A}_i)_{i=1}^p, (\mathbf{B}_i)_{i=1}^p$ and the goal is to use coding to determine $(\mathbf{A}_i\mathbf{B}_i)_{i=1}^p$ [4], [5]. While state-of-the-art codes for these problem spaces provide near optimal straggler resilience, ultimately they rely on the rigid structure of the computation tasks they aim to compute and are hard to extrapolate to more variable tasks. For example, a variable computation task could require the following matrix products $(\mathbf{A}_1\mathbf{B}_1, \mathbf{A}_1\mathbf{B}_2, \mathbf{A}_2\mathbf{B}_2, \mathbf{A}_2\mathbf{B}_3)$ which is clearly not well suited to the previous two problem spaces. An example of such a variable task in practice is when a group of users, such as mobile devices, who each have their own data $(\mathbf{A}_i)$, want to perform some computation (matrix multiplication) on data stored in the cloud $(\mathbf{B}_j)$. It is reasonable that user requests may have significant overlap and utilizing this redundant overlap can improve the throughput of the system.

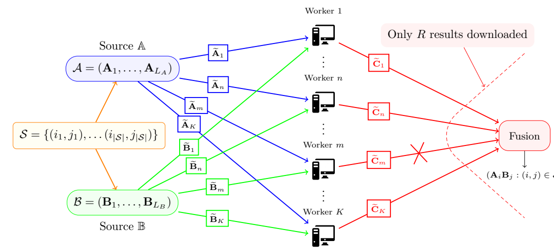

Fig. 1: System Model for Variable Coded Distributed Batch Matrix Multiplication with a recovery threshold of R.

In this work, we present the Variable Coded Distributed Batch Matrix Multiplication (VCDBMM) problem that generalizes the two stated problem spaces. Assume that a distributed system is provided with two sets of matrices $\mathcal{A} = \{\mathbf{A}_1, \ldots, \mathbf{A}_{L_A}\}$ and $\mathcal{B} = \{\mathbf{B}_1, \ldots, \mathbf{B}_{L_B}\}$ such that $\mathbf{A}_i \in \mathbb{F}^{\alpha\times\beta}$ and $\mathbf{B}_j \in \mathbb{F}^{\beta\times\gamma}$ and a set of computation goals $\mathcal{S} = \{(i_1, j_1), \ldots (i_{|\mathcal{S}|}, j_{|\mathcal{S}|})\}$. The objective is to calculate the product $\mathbf{A}_i\mathbf{B}_j$ for every $(i, j) \in \mathcal{S}$. Fig. 1 provides an overview of the system model of VCDBMM where there are K workers and $\mathcal{A}$ and $\mathcal{B}$ are located at two sources $\mathbb{A}$ and $\mathbb{B}$, respectively. Each source encodes the data stored within itself and sends it to a worker node. The worker nodes perform a simple matrix product of its inputs and provides its output to a fusion node. Due to the presence of stragglers, only some of the worker results are received by the fusion node. An important metric to measure how resilient a system is to stragglers is the *recovery threshold* which is the minimum number of worker outputs needed to recover the desired results.

Our goal is to demonstrate a coding scheme that has a low recovery threshold with minimal overhead. We achieve this through Flexible Cross-Subspace Alignment (FCSA) codes that take advantage of the redundancy in the VCDBMM problem to provide straggler resilience.

Due to limited space, the following sections are shortened from our long version [7] and we do not provide a full description of FCSA codes. We refer readers to the long version for detailed background and derivation of equations.

Notation: Let boldface capital letters represent matrices. Let $[n]$ denote the set $\{1, 2, \ldots, n\}$. Given two sets A and B, $A\times B$ is the Cartesian product of the two sets. The notation $\widetilde{\mathcal{O}}(a\log^2 b)$ suppresses poly-log terms, i.e., $\widetilde{\mathcal{O}}(a\log^2 b) = \mathcal{O}(a\log^2 b\log\log(b))$.

II. Flexible Cross-Subspace Alignment Codes

First, we define two constructs that we use to succinctly describe FCSA codes.

Definition 1. *For a given computation list $\mathcal{S}$, we define a task assignment $\mathcal{Q}$ as a set of tuples $\mathcal{Q} = \{(\mathcal{L}_A^q, \mathcal{L}_B^q)\}_{q=1}^{|\mathcal{Q}|}$ where $\mathcal{L}_A^q \subseteq [L_A], \mathcal{L}_B^q \subseteq [L_B]$ are chosen such that for every $s \in \mathcal{S}$ there exists exactly one $q \in [|\mathcal{Q}|]$ such that $s \in \mathcal{L}_A^q \times \mathcal{L}_B^q$. Define $L_A^{(q)} = |\mathcal{L}_A^q|$ and $L_B^{(q)} = |\mathcal{L}_B^q|$.*

Now, given a task assignment $\mathcal{Q}$, we define a power assignment $\mathcal{P}$ as a tuple of vectors of non-negative integers $\mathcal{P} = \{(P^{A,q}, P^{B,q})\}_{q=1}^{|\mathcal{Q}|}$. Let $P_i^{A,q}$ ($P_j^{B,q}$) be the i^{th} (j^{th}) element in $P^{A,q}$ ($P^{B,q}$). The power assignment $\mathcal{P}$ must satisfy the following constraints for all $q \in [|\mathcal{Q}|]$:

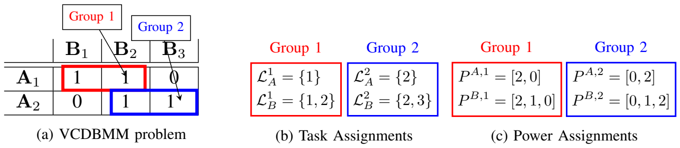

Fig. 2: An example FCSA code with a task and power assignment which results in a recovery threshold of 5 for $m = p = n = 1$ ($R_{1,1,1} = 1$) while the best recovery threshold using other codes is 6, under the same communication costs.

1) $\{P_i^{A,q} + P_j^{B,q} - L_A^{(q)} L_B^{(q)} : (i, j) \in \mathcal{L}_A^q \times \mathcal{L}_B^q\}$ *is a permutation of the set* $[L_A^{(q)} L_B^{(q)}]$.
2) $P_i^{A,q} + P_j^{B,q} \leq L_A^{(q)} L_B^{(q)}$ *for all* $(i, j) \notin \mathcal{L}_A^q \times \mathcal{L}_B^q$

We provide an explicit construction of power assignments in the full version [7] that satisfy the constraints. Using the task and power assignment constructs, we provide a full characterization of FCSA codes in the following theorem.

Theorem 2. *Assume that a task assignment* $\mathcal{Q}$ *and power assignment* $\mathcal{P}$ *are provided. Additionally, parameters* $m, p, n \in \mathbb{Z}_>$ *are provided such that* $m|\alpha$, $p|\beta$, *and* $n|\gamma$. *Let* $R = R_{m,p,n}$ *denote the bilinear complexity*[1] *of multiplying an* m*-by-*p *matrix and a* p*-by-*n *matrix and let* $\rho > 0$ *be a parameter such that* $\rho|R$. *If* $|\mathbb{F}| > R_{m,p,n}|\mathcal{Q}| + K$, *FCSA codes achieve the following:*

$$\textit{Recovery Threshold: } \widetilde{R} = (R + \rho) \sum_{q=1}^{|\mathcal{Q}|} L_A^{(q)} L_B^{(q)}$$
$$- \min_{i \in [L_A]} \left(\sum_{q=1}^{|\mathcal{Q}|} P_i^{A,q} \right) - \min_{j \in [L_B]} \left(\sum_{q=1}^{|\mathcal{Q}|} P_j^{B,q} \right) + 1$$
$$\textit{Source } \mathbb{A} \textit{ Upload Cost: } \frac{R\alpha\beta}{\rho m p}, \textit{ Source } \mathbb{B} \textit{ Upload Cost: } \frac{R\beta\gamma}{\rho p n}$$
$$\textit{Download Cost: } \frac{\alpha\gamma}{mn}, \textit{Worker Complexity: } \mathcal{O}\left(\frac{R}{\rho} \cdot \frac{\alpha\beta\gamma}{mpn} \right)$$
$$\textit{Source } \mathbb{A} \textit{ Encoding Complexity: } \mathcal{O}\left(\alpha\beta L_A R (\frac{K}{mp} + 1) \right),$$
$$\textit{Source } \mathbb{B} \textit{ Encoding Complexity: } \mathcal{O}\left(\beta\gamma L_B R (\frac{K}{np} + 1) \right),$$
$$\textit{Decoding Complexity: } \mathcal{O}(\alpha\gamma R|\mathcal{S}| + \frac{\alpha\gamma}{mn} R \sum_{q=1}^{|\mathcal{Q}|} (L_A^{(q)} L_B^{(q)})^2)$$
$$+ \widetilde{\mathcal{O}}(\frac{\alpha\gamma}{mn} \widetilde{R} \log^2(\widetilde{R})).$$

Fig. 2 provides a small example of a valid task and power assignment from which we can calculate all the necessary metrics. At a high level, FCSA codes group up matrices in $\mathcal{A}$ and $\mathcal{B}$ such that all matrix pairs in that group are retrievable due to alignment in a decodable subspace while other matrix terms are compounded in a garbage subspace. Power assignment reduces the sizes of these subspaces which lowers the recovery threshold.

III. Numerical Analysis of Recovery Threshold

From Theorem 2, we can see that FCSA codes provide a variety of operating points for a system to choose from. Now, we analyze the optimality of the recovery threshold of FCSA codes by providing a lower bound on the optimal recovery threshold.

Theorem 3. *For a fixed download cost of* $\frac{\alpha\gamma}{nm}$, *then the optimal recovery threshold* $R_{\mathcal{S}}^*$ *satisfies* $R_{\mathcal{S}}^* \geq mn|\mathcal{S}|$.

The key idea to prove this lower bound is utilizing mutual information to lower bound the number of symbols needed to sufficiently describe the desired result. Using Theorem 3, we can analyze how close FCSA codes are to the optimal recovery threshold. Next, we provide two subclasses of FCSA codes named Type-1 (T1) FCSA and Type-2 (T2) FCSA codes by explicitly stating their task assignment which allows for a simple expression for their recovery threshold (see [7] for explicit construction). The next theorem shows that both these subclasses achieve a recovery threshold within a multiplicative factor of 2.

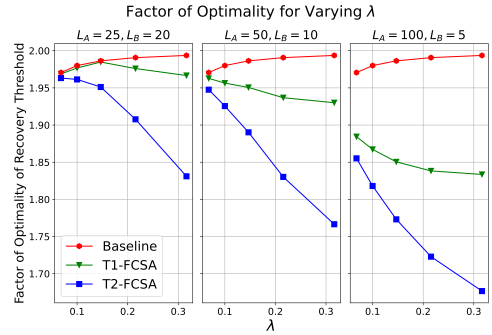

Fig. 3: $G_{L_A, L_B, \lambda}$ for the $V_\lambda(L_A, L_B)$ ensemble with different L_A and L_B and varying values of λ.

Theorem 4. *Let* R_{FCSA}^{T1} *and* $R_{FCSA}^{T_2}$ *be the recovery thresholds of T1-FCSA and T2-FCSA codes, respectively. For a fixed download cost of* $\frac{\alpha\gamma}{nm}$, $R_{FCSA}^{T2} \leq R_{FCSA}^{T1} \leq 2R_{\mathcal{S}}^*$.

We note that the codes provided in [4] can also achieve a multiplicative factor of optimality of 2 for VCDBMM. The major difference is that this bound is quite tight for the codes in [4] while we can empirically show that FCSA codes can achieve an even better recovery threshold then the bound implies. Additionally, we note that while T2-FCSA codes have a better recovery threshold than T1-FCSA codes, T1-FCSA codes have a simpler decoding complexity [7]. Thus, we will compare the recovery thresholds of both types of FCSA codes.

We now provide a numerical analysis of the average multiplicative factor of optimality for T1-FCSA and T2-FCSA codes, i.e. the average value of the recovery threshold divided by the lower bound. This average is taken over an ensemble of VCDBMM problems where the number of matrices L_A and L_B are fixed and each pair of matrices appears in $\mathcal{S}$ with probability $0 < \lambda < 1$. For simplicity, we fix the download cost to be $\alpha\gamma$. We compare to a baseline recovery threshold which is the lowest recovery threshold of the codes provided in [2], [5] with equivalent communication costs. Fig 3 provides the results of the experiments. As we can see, FCSA codes can achieve an average multiplicative factor of optimality as low as 1.7. Other experiments over different ensembles show similar results and can be viewed in the full version [7]. Thus, FCSA codes are well-suited to solve the VCDBMM problem.

[1]See [8] for a description of bilinear complexity. While the bilinear complexity is not known for all m, p, n, there are many well known constructions that achieve an upper bound on the bilinear complexity and $R_{m,p,n}$ is known to be sub-cubic in its parameters [4].

References

[1] J. Dean and L. A. Barroso, “The tail at scale,” *Communications of the ACM*, 2013.
[2] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, “Polynomial codes: an optimal design for high-dimensional coded matrix multiplication,” in *Proc. Conf. on Neural Info. Proc. Systems (NIPS)*, 2017.
[3] S. Dutta *et al.*, “On the optimal recovery threshold of coded matrix multiplication,” *IEEE Trans. on Info. Theory*, 2020.
[4] Q. Yu and A. S. Avestimehr, “Entangled polynomial codes for secure, private, and batch distributed matrix multiplication: Breaking the" cubic" barrier,” in *Proc. IEEE Int. Symp. on Info. Theory (ISIT)*, 2020.
[5] Z. Jia and S. A. Jafar, “Cross subspace alignment codes for coded distributed batch computation,” *IEEE Trans. on Info. Theory*, 2021.
[6] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, “Straggler mitigation in distributed matrix multiplication: Fundamental limits and optimal coding,” *IEEE Trans. on Info. Theory*, 2020.
[7] L. Tauz and L. Dolecek, “Variable coded batch matrix multiplication,” *IEEE Journal on Selected Areas in Information Theory*, 2022.
[8] M. Bläser, “Fast matrix multiplication,” *Theory Comput.*, 2013.