Byzantine Attack Identification in Coded Computing via Group testing Approach

Hong Sang Woo, Yang Hee cheol*, Shin Won Jae† , Lee Jung woo
Seoul National Univ., *Chungnam National Univ., † Ajou Univ.
tkddn0606@snu.ac.kr, *hcyang@cnu.ac.kr,
† wjshin@ajou.ac.kr, junglee@snu.ac.kr

Abstract

Coded computing has proved its efficiency in handling a straggler issue in distributed computing framework. It uses error correcting codes to mitigate the effect of the stragglers. However, in a coded distributed computing framework, there may exist Byzantine workers who send the wrong computation results to a master in order to contaminate the overall computation output. Therefore, it is essential to identify Byzantine workers from their computation results. In this paper, we consider the Byzantine attack identification problem in distributed computing for distributed matrix multiplication tasks. We propose a new coding scheme, namely locally testable codes, and suggest a hierarchical group testing method for Byzantine attack identification on locally testable codes. We show that the hierarchical group testing method requires a smaller number of tests than naively applying the conventional group testing method for the existing coded computing schemes.

I. INTRODUCTION

In this paper, we consider Byzantine attack scenario in distributed computing. For Byzantine attack scenario, some malicious workers, i.e., Byzantine workers, may intentionally return the wrong results on their assigned computation tasks in order to contaminate the final computation product at a master. Therefore, if there exist Byzantine workers, the reliability of the distributed computing systems will be threatened. Consequently, it is highly important to identify Byzantine workers in a distributed computing system.

The main contribution of this paper is threefold:

- Our work is the first to apply group testing methods for Byzantine worker identification in distributed matrix-matrix multiplication. Using group testing for Byzantine worker identification has linear complexity asymptotically and can provide twice more robustness compare to existing methods like Lagrange coded computing [1], which have quadratic complexity.

- We also propose a new coding scheme for distributed matrix multiplication, namely locally testable codes. In conventional coded computing schemes like [2]-[3], more than R (the number of results that are needed to decode the final product, which will be defined as recovery threshold) workers are needed to apply group testing. In locally testable codes, a master can perform group testing for Byzantine attack identification on a smaller number of workers, which facilitates more efficient Byzantine worker identification. We also claim that there exist a trade-off between the minimum testable size and the achievable recovery threshold in locally testable codes, thus one can choose adequate parameter for locally testable codes depending on their distributed computing environment.

- We suggest a hierarchical group testing algorithm suitable for locally testable codes. In hierarchical group testing, Byzantine attack identification is performed in two stages: local test and global test. By hierarchical group testing, it is shown that the total number of group testing trials and the computational complexity for group testing can be significantly reduced.

2. SYSTEM MODEL FOR CODED DISTRIBUTED COMPUTING

We consider a coded distributed computing for matrix multiplication task, where a master wants to perform a matrix multiplication \( \mathbf{C} = \mathbf{A}^T \mathbf{B} \) with W workers. The system model is depicted in Figure 1. Each worker \( \mathbf{W}_i \), is assigned to compute a task \( \mathbf{C}_i = \mathbf{A}_i \times \mathbf{B}_i \), where \( \mathbf{A}_i \) and \( \mathbf{B}_i \) are encoded from A and B using encoding functions \( \mathbf{p}_A \) and \( \mathbf{p}_B \) at a master, respectively. After the master receives enough number of results from the workers, i.e., recovery threshold, it starts decoding the final product \( \mathbf{C} = \mathbf{A}^T \mathbf{B} \).

Assumption on the Byzantine workers: We assume that among the workers, some of them could be Byzantine workers who deliberately send wrong values to the master. Byzantine workers are assumed to behave maliciously to contaminate the computation results or to slow down the process. We assume the worst case scenario where the Byzantine workers have knowledge about the protocol of distributed computing and the input matrices A and B. We also assume that the Byzantine workers cannot collude each other, which means Byzantine workers cannot access the information about the allocated task at the other workers.
Main problem of this paper: i) how to design the encoding functions $p_x$ and $p_y$, and the decoding function $d(p_y)$ for a distributed matrix multiplication task $C = A^T B$, and ii) how to identify Byzantine workers among the workers from the computation results, a.k.a. Byzantine attack identification.

3. BYZANTINE WORKER IDENTIFICATION USING GROUP TESTING

Group testing [4] is a mathematical strategy for identifying the defective members in a large group. Each member in the group can be tested individually or mutually to find out defective members. Example of group testing for 9 members including 1 Byzantine member is depicted in Figure 2. In this example, identification is done with 6 group tests.

Samples are mixed together in equal-sized groups and tested. If a result of group test is positive, every sample is retested individually.

![Example of group testing for 9 members including 1 Byzantine member](image)

Figure 2. Example of group testing for 9 members including 1 Byzantine member

Coded computing uses encoding matrix for encoding functions. By using the inverse of encoding matrix as group testing matrix, group test can be done for more than $R$ workers in conventional coded computing schemes. In locally testable codes, we use carefully designed Vandermonde matrix for encoding matrix, which enables group testing on smaller number of workers.

4. RESULTS

We firstly compare Byzantine worker identification using group testing with existing method in Table 1.

![The required number of the group testing trials with different number of Byzantine workers](image)

Figure 3. The required number of the group testing trials with different number of Byzantine workers

Furthermore, as we can see in Figure 4, locally testable codes provide trade-off between minimum testable size and recovery threshold. This implies required number of group testing can be further reduced by selecting adequate value of parameters, increasing the recovery threshold as a trade-off.

![Trade-off between minimum testable size and recovery threshold of locally testable codes when K=24](image)

Figure 4. Trade-off between minimum testable size and recovery threshold of locally testable codes when $K=24$

As we can see in the Table 1, group testing has twice more robustness than existing method and has much lower identification complexity. However, naively applying group testing to existing codes could not fully exploit the strength of group testing methods, since unlike conventional group testing scenario, existing methods of coded computing have fixed minimum testable size as $R + 1$. Therefore, we design a new code that relax the minimum testable size constraints, thereby further reducing the required number of group testing trials. Figure 3 shows required number of group testing trials for locally testable codes and existing schemes.

### ACKNOWLEDGMENT

This work is in part supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2021-0-00467, Intelligent 6G Wireless Access System, 100%), INMAC, and BK21-plus.

### REFERENCES


