PolyDot Coded Privacy Preserving Multi-Party Computation at the Edge

Elahe Vedadi
University of Illinois at Chicago
evedad2@uic.edu

Yasaman Keshtkarjahromi
Seagate Technology
yasaman.keshtkarjahromi@seagate.com

Hulya Seferoglu
University of Illinois at Chicago
hulya@uic.edu

***Abstract*—We investigate the problem of privacy preserving distributed matrix multiplication in edge networks using multi-party computation (MPC). Coded multi-party computation (CMPC) is an emerging approach to reduce the required number of workers in MPC by employing coded computation. Existing CMPC approaches usually combine coded computation algorithms designed for efficient matrix multiplication with MPC. We show that this approach is not efficient. We design a novel CMPC algorithm; PolyDot coded MPC (PolyDot-CMPC) by using PolyDot codes. We exploit "garbage terms" that naturally arise when polynomials are constructed in the design of PolyDot-CMPC to reduce the number of workers needed for privacy-preserving computation. We show that entangled polynomial codes, which are consistently better than PolyDot codes in coded computation setup, are not necessarily better than PolyDot-CMPC in MPC setting.**

I. Introduction

Privacy-preserving distributed computing in edge networks is crucial for Internet of Things (IoT) applications including smart homes, self-driving cars, wearables, etc. Multi-party computation (MPC), which is a privacy-preserving distributed computing framework [1], is a promising approach. The main goal of MPC is to calculate a function of data stored in multiple parties such as end devices and edge servers in edge computing systems. In this paper, we focus on BGW (Ben-Or, Goldwasser and Widgerson) [2], an information theoretic MPC solution due to its lower computing load as well as quantum safe nature [3] rather than cryptographic solutions [4], [5]. Despite its potential, BGW should adapt to the limited resources of edge networks.

Coded-MPC (CMPC) [6], [7] aims to improve BGW and make it adaptive to limited edge resources by employing coded computation [8], [9]. Coded computation advocates splitting computationally intensive tasks into smaller ones, coding these sub tasks using error correcting codes, and distributively processing coded tasks in parallel at workers (end devices or edge servers in our setup). This idea turns out to address the straggling workers problem [8], [9]. CMPC uses the coded computation idea in MPC setup to reduce the required number of workers, which is limited in edge systems.

This work was supported in parts by the Army Research Lab (ARL) under Grant W911NF-2120272 and National Science Foundation (NSF) under Grants CCF-1942878 and CNS-1801708.

Existing CMPC approaches [6], [7] usually combine coded computation algorithms designed for efficient matrix multiplication with MPC. In this paper, we show that this approach is not efficient with regard to reducing the required number of workers as it does not consider an important relationship between coded computation and MPC. Actually, the required number of workers (or efficiency of a code) is directly related with the powers of the created polynomials in coded computation. For example, the efficiency of polynomial codes reduces if there are gaps in the powers of the polynomials in coded computation. On the other hand, our key observation shows that such gaps help to reduce the required number of workers in CMPC setup. In particular, when there are gaps among powers of the coded terms, multiplication of the coded terms may create additional terms that we name "garbage terms", which can be used to reduce the required number of workers. The next example illustrates our key observation.

Example 1: MatDot-Coded MPC.[1] Let us assume that there are two end devices; source 1 and source 2 that own matrices A and B, respectively. Our objective is to compute $Y = A^T B$, which is a computationally exhaustive task for large A and B matrices, while preserving privacy. To achieve this goal, end users need the help of edge servers (workers). Assume that matrices A and B are divided into two parts row-wise such that: $A^T = [A_1 \;\; A_2]$ and $B^T = [B_1 \;\; B_2]$, where $Y = A^T B$ is constructed as $A^T B = A_1 B_1 + A_2 B_2$.

When the number of colluding workers is $z = 2$, source 1 and source 2 construct polynomials $F_A(x) = A_1 + A_2 x + \bar{A}_3 x^2 + \bar{A}_4 x^3$ and $F_B(x) = B_1 x + B_2 + \bar{B}_3 x^2 + \bar{B}_4 x^3$. The first two terms, namely, coded terms in these polynomials are determined by MatDot codes [10], and the second two terms, *i.e.,* secret terms, are designed by our proposed PolyDot-CMPC method, which we explain later in the paper. We note that the degree of the secret terms starts from two. The reason is that the multiplication of the coded terms becomes $(A_1 B_1 + A_2 B_2)x + A_1 B_2 + A_2 B_1 x^2$, where the only term we need to recover $Y = A^T B$ is $(A_1 B_1 + A_2 B_2)x$. Other terms, namely, $A_1 B_2$ and $A_2 B_1 x^2$, are called *garbage terms*.

After $F_A(\alpha_n)$ and $F_B(\alpha_n)$ are sent from source 1 and source 2 to workers, worker n determines $H(\alpha_n) =$

[1] Although our PolyDot-CMPC mechanism uses PolyDot codes, we use MatDot codes in this example to explain the "garbage terms" in a simple way.

$F_A(\alpha_n)F_B(\alpha_n)$, where $H(x) = A_1B_2 + (A_1B_1 + A_2B_2)x + \sum_{i=2}^{6} H_i x^i$. Next, each worker n computes the multiplication of r_n with $H(\alpha_n)$ and creates the polynomial $G_n(x)$ as $G_n(x) = r_n H(\alpha_n) + R_0^{(n)}x + R_1^{(n)}x^2$, where the selection of r_n's, $R_0^{(n)}$'s, and $R_1^{(n)}$'s will be explained later in the paper. Then, worker n sends $G_n(\alpha_{n'})$ to worker n'. After all data exchanges, worker n', knowing $G_n(\alpha_{n'})$, calculates their sum and sends $I(\alpha_{n'}) = \sum_{n=1}^{7} G_n(\alpha_{n'})$ to the master (one of the edge devices that would like to get the calculated value of $Y = A^TB$), where $I(x) = A_1B_1 + A_2B_2 + \sum_{n=1}^{7} R_0^{(n)}x + \sum_{n=1}^{7} R_1^{(n)}x^2$. In the last phase, the master reconstructs $I(x)$ once it receives $I(\alpha_n)$ from $1 + z = 3$ workers. After reconstructing $I(x)$ and determining all coefficients, $Y = A^TB = A_1B_1 + A_2B_2$ is calculated in a privacy-preserving manner. The number of terms with non-zero coefficients in $H(x)$ is equal to 7. Thus, 7 workers are required for privacy-preserving computation. For the same number of colluding workers and matrix partitions, polynomial coded MPC [6], which divides matrices A and B into two column-wise partitions, requires 11 workers.[2] □

The above example demonstrates the importance of the garbage terms for the efficiency of CMPC algorithms. Based on this observation and exploiting the garbage terms, we design PolyDot-CMPC. We show that PolyDot-CMPC reduces the required number of workers for several colluding workers as compared to entangled polynomial coded MPC (Entangled-CMPC) [7]. This result is surprising as entangled polynomial codes are consistently better than PolyDot codes in coded computation setup [11]. We also compare PolyDot-CMPC with baselines; SSMM [12], and GCSA-NA [13]. We show that PolyDot-CMPC performs better than SSMM [12] and GCSA-NA [13] for a range of colluding workers.

The structure of the rest of this paper is as follows. Section II presents our system model. Section III outlines the attack model we consider in this work. Section IV presents our PolyDot-CMPC algorithm as well as its performance analysis as compared to baselines. Section V provides simulation results of PolyDot-CMPC. Section VI concludes the paper.

II. System Model

We consider an MPC system containing E sources, N workers, and a master node, where all of them are edge devices with limited resources. There exists no connection among source nodes, but there are connections between sources and workers. All workers are connected to each other, and there exists a connection between the master node and each worker. Private data χ_e is stored at source node e. The goal is to compute $Y = f(\chi_1, \ldots, \chi_E)$ in a privacy-preserving manner. The function $f(.)$ stands for any polynomial function, but we focus on the multiplication of two square matrices (which can be easily extended to general matrices). In particular, we consider $\chi_1 = A$ and $\chi_2 = B$, and calculate $Y = f(A, B) = A^TB$.

[2]This example is a special case of both PolyDot-CMPC and Entangled-CMPC [7], when matrices A and B are partitioned row-wise, but the idea of garbage terms is not discussed in [7].

Given the above system model, we use the following notation in the rest of this paper. Considering two arbitrary sets $\mathbf{I}$ and $\mathbf{J}$, with integer elements $i, j \in \mathbb{Z}$, we have; (i) $\mathbf{I} + \mathbf{J} = \{i + j : i \in \mathbf{I},\ j \in \mathbf{J}\}$; (ii) $\mathbf{I} + j = \{i + j : i \in \mathbf{I}\}$; and (iii) $|\mathbf{I}|$ stands for the cardinality of $\mathbf{I}$. We define Ω_i^j as $\Omega_i^j = \{i, \ldots, j\}$. We show the divisibility with $k|m$, *i.e.,* m is divisible by k. Considering a polynomial $f(x) = \sum_{i=0}^{n} a_i x^i$, $\mathbf{P}(f(x))$ is defined as the set of powers of the terms in $f(x)$ with non-zero coefficients, *i.e.,* $\mathbf{P}(f(x)) = \{i \in \mathbb{Z} : 0 \leq i \leq n,\ a_i \neq 0\}$. Finally, if a matrix A is divided into s row-wise and t column-wise partitions, it is represented as

$$A = \begin{bmatrix} A_{0,0} & \ldots & A_{0,t-1} \\ \vdots & \ddots & \vdots \\ A_{s-1,0} & \ldots & A_{s-1,t-1} \end{bmatrix}. \quad (1)$$

III. Attack Model

A semi-honest system model is considered in this paper where all parties (master, workers, and sources) are honest and follow the exact protocol defined by PolyDot-CMPC, but they are eavesdropping and potentially spying about private data. We design PolyDot-CMPC such that it is information theoretically secure against z colluding workers, where z is less than half of the total number of workers, *i.e.,* $z < N/2$. More specifically, we provide privacy requirements from source, worker and master nodes' perspective next.

Sources: The private data of each source node, should be kept private from all other sources. Our system model satisfies this condition since, source nodes do not communicate. Also, the worker nodes and the master node do not send data to any of the source nodes.

Workers: There should not be any privacy violation when workers receive data from sources, communicate with other workers and the master. Such privacy requirement should be satisfied if no more than z workers collude. More formally, the following condition should be satisfied; $\tilde{H}(\chi_1, \ldots, \chi_E | \bigcup_{n \in \mathcal{N}_c} (\{G_{n'}(\alpha_n), n' \in \Omega_1^N\}, \bigcup_{e \in \Omega_1^E} F_e(\alpha_n))) = \tilde{H}(\chi_1, \ldots, \chi_E)$, where $\tilde{H}$ is the Shannon entropy, α_n is from finite field and known by all workers, $G_{n'}(\alpha_n)$ is the data that worker n gets from worker n', $F_e(\alpha_n)$ is the data that worker n gets from source e, and $\mathcal{N}_c$ is a subset of $\{0, \ldots, N\}$ with cardinality less than or equal to z.

Master: Everything, except the final result Y, should be kept private from the master node. In particular, the following condition should be satisfied; $\tilde{H}(\chi_1, \ldots, \chi_E | Y, \bigcup_{n \in \Omega_1^N} I(\alpha_n)) = \tilde{H}(\chi_1, \ldots, \chi_E | Y)$, where $I(\alpha_n)$ is the data received from worker n by the master node.

IV. PolyDot Coded MPC (PolyDot-CMPC)

In this section, we present our PolyDot coded MPC framework (PolyDot-CMPC) that employs PolyDot coding [10] to create coded terms. Our design is based on leveraging the garbage terms that are not required for computing $Y = A^TB$ and reusing them in the secret terms.

A. PolyDot-CMPC

Sources. Source 1 and source 2 divide matrices $A \in \mathbb{F}^{m\times m}$ and $B \in \mathbb{F}^{m\times m}$ into s row-wise and t column-wise partitions as in (1), where $s, t \in \mathbb{N}$, and $s|m$ and $t|m$ hold. Using the splitted matrices $A_{i,j} \in A^T$ and $B_{k,l} \in B$, where $i, l \in \Omega_0^{t-1}$, $j, k \in \Omega_0^{s-1}$, they generate polynomials $F_A(x)$ and $F_B(x)$, which consist of coded and secret terms, *i.e.,* $F_{i'}(x) = C_{i'}(x) + S_{i'}(x)$, $i' \in \{A, B\}$, where $C_{i'}(x)$'s are the coded terms defined by PolyDot codes [10], and $S_{i'}(x)$'s are the secret terms that we construct. Next, we discuss the construction of $S_{i'}(x)$, hence $F_A(x)$ and $F_B(x)$ in detail.

Let $\mathbf{P}(C_A(x))$ and $\mathbf{P}(C_B(x))$ be sets of the powers of the polynomials $C_A(x)$ and $C_B(x)$ with coefficients larger than zero. $\mathbf{P}(C_A(x))$ and $\mathbf{P}(C_B(x))$ are expressed as

$$\begin{aligned}\mathbf{P}(C_A(x)) =&\{i + tj \in \mathbb{N} : i \in \Omega_0^{t-1},\ j \in \Omega_0^{s-1}\}\\ =&\{0, \dots, ts-1\},\end{aligned} \tag{2}$$

$$\begin{aligned}\mathbf{P}(C_B(x)) =&\{t(s-1-k) + l\theta' \in \mathbb{N} : k \in \Omega_0^{s-1},\ l \in \Omega_0^{t-1}\}\\ =&\{tq' + l\theta' \in \mathbb{N} : q' \in \Omega_0^{s-1},\ l \in \Omega_0^{t-1}\},\end{aligned} \tag{3}$$

where $s, t \in \mathbb{N}$, and $\theta' = t(2s-1)$.

As seen from (2) and (3), $\mathbf{P}(C_A(x)C_B(x))$ is the set of the powers of the polynomial $C_A(x)C_B(x)$ with coefficients larger than zero, and is expressed as $\mathbf{P}(C_A(x)C_B(x)) = \{i + t(s-1+j-k) + tl(2s-1) \in \mathbb{N} : i, l \in \Omega_0^{t-1},\ j, k \in \Omega_0^{s-1}\}$. Furthermore, we know from [10] that $Y_{i,l} = \sum_{j=0}^{s-1} A_{i,j}B_{j,l}$, which are the coefficients of $x^{i+t(s-1)+tl(2s-1)}$ in $C_A(x)C_B(x)$, are the elements of the final result $Y = A^TB$. Therefore, we define $\{i + t(s-1) + tl(2s-1) \in \mathbb{N} : i, l \in \Omega_0^{t-1}\}$ as the set of important powers of $C_A(x)C_B(x)$. We define the secret terms $S_A(x)$ and $S_B(x)$ so that the important powers of $C_A(x)C_B(x)$ do not have common terms with $\mathbf{P}(C_A(x)S_B(x))$, $\mathbf{P}(S_A(x)C_B(x))$, and $\mathbf{P}(S_A(x)S_B(x))$. The reason is that $Y_{i,l}$'s should not have any overlap with the other components for successful recovery of Y. The following conditions should hold to guarantee this requirement.

$$\begin{aligned}&\text{C1: } i + t(s-1) + tl(2s-1) \notin \mathbf{P}(S_A(x)) + \mathbf{P}(C_B(x)),\\ &\text{C2: } i + t(s-1) + tl(2s-1) \notin \mathbf{P}(S_A(x)) + \mathbf{P}(S_B(x)),\\ &\text{C3: } i + t(s-1) + tl(2s-1) \notin \mathbf{P}(S_B(x)) + \mathbf{P}(C_A(x)),\end{aligned} \tag{4}$$

where $i, l \in \Omega_0^{t-1}$ and $s, t \in \mathbb{N}$. We determine $\mathbf{P}(S_A(x))$ and $\mathbf{P}(S_B(x))$ according to the following set of rules; (i) determine all elements of $\mathbf{P}(S_A(x))$, starting from the minimum possible element, satisfying C1 in (4), (ii) fix $\mathbf{P}(S_A(x))$ in C2 of (4), and find all elements of the subset of $\mathbf{P}(S_B(x))$, starting from the minimum possible element, that satisfies C2; we call this subset as $\mathbf{P}'(S_B(x))$, (iii) determine all elements of the subset of $\mathbf{P}(S_B(x))$, starting from the minimum possible element, that satisfies C3 in (4); we call this subset as $\mathbf{P}''(S_B(x))$, and (iv) find the intersection of $\mathbf{P}'(S_B(x))$ and $\mathbf{P}''(S_B(x))$ to form $\mathbf{P}(S_B(x))$. In our PolyDot-CMPC mechanism, we define the polynomials $F_A(x)$ and $F_B(x)$, based on the above strategy as formalized in Theorem 1.

Theorem 1: With the following design of $F_A(x)$ and $F_B(x)$ in PolyDot-CMPC, the conditions in (4) are satisfied.

$$F_A(x) = \begin{cases} F_{A_1}(x) & z > ts-t \text{ and } s, t \neq 1\\ F_{A_2}(x) & z \leq ts-t \text{ or } t = 1 \text{ or } s = 1\end{cases} \tag{5}$$

$$\begin{aligned}F_{A_1}(x) =& \underbrace{\sum_{i=0}^{t-1}\sum_{j=0}^{s-1} A_{i,j}x^{i+tj}}_{\triangleq C_A(x)} + \underbrace{\sum_{w=0}^{t(s-1)-1}\sum_{l=0}^{p-1} \bar{A}_{(w+\theta' l)}x^{ts+\theta' l+w}}_{\triangleq S_{A_1}(x)}\\ &+ \underbrace{\sum_{u=0}^{z-1-pt(s-1)} \bar{A}_{(u+t(s-1)+\theta'(p-1))}x^{ts+\theta' p+u}}_{\triangleq S_{A_1}(x)},\end{aligned} \tag{6}$$

$$F_{A_2}(x) = \underbrace{\sum_{i=0}^{t-1}\sum_{j=0}^{s-1} A_{i,j}x^{i+tj}}_{\triangleq C_A(x)} + \underbrace{\sum_{u=0}^{z-1} \bar{A}_u x^{ts+\theta' p+u}}_{\triangleq S_{A_2}(x)}, \tag{7}$$

$$F_B(x) = \begin{cases} F_{B_1}(x) & z > \tau \text{ or } t = 1 \text{ or } s = 1\\ F_{B_2}(x) & \frac{\tau+1}{2} < z \leq \tau \text{ and } s, t \neq 1\\ F_{B_3}(x) & z \leq \frac{\tau+1}{2} \text{ and } s, t \neq 1\end{cases} \tag{8}$$

$$F_{B_1}(x) = \underbrace{\sum_{k=0}^{s-1}\sum_{l=0}^{t-1} B_{k,l}x^{t(s-1-k)+\theta' l}}_{\triangleq C_B(x)} + \underbrace{\sum_{r=0}^{z-1} \bar{B}_r x^{ts+\theta'(t-1)+r}}_{\triangleq S_{B_1}(x)}, \tag{9}$$

$$\begin{aligned}F_{B_2}(x) =& \underbrace{\sum_{k=0}^{s-1}\sum_{l=0}^{t-1} B_{k,l}x^{t(s-1-k)+\theta' l}}_{\triangleq C_B(x)}\\ &+ \underbrace{\sum_{d=0}^{\tau-z}\sum_{l'=0}^{p'-1} \bar{B}_{(d+\theta' l')}x^{ts+\theta' l'+d}}_{\triangleq S_{B_2}(x)}\\ &+ \underbrace{\sum_{v=0}^{z-1-p'(\tau-z+1)} \bar{B}_{(v+\tau-z+1+\theta'(p'-1))}x^{ts+\theta' p'+v}}_{\triangleq S_{B_2}(x)},\end{aligned} \tag{10}$$

$$F_{B_3}(x) = \underbrace{\sum_{k=0}^{s-1}\sum_{l=0}^{t-1} B_{k,l}x^{t(s-1-k)+\theta' l}}_{\triangleq C_B(x)} + \underbrace{\sum_{v=0}^{z-1} \bar{B}_v x^{ts+v}}_{\triangleq S_{B_3}(x)} \tag{11}$$

where $p = \min\{\lfloor \frac{z-1}{ts-t} \rfloor, t-1\}$, $\tau = \theta' - ts -$

t, $p' = \min\{\lfloor \frac{z-1}{\tau-z+1} \rfloor, t-1\}$. Moreover, $\bar{A}_{(w+\theta' l)}$, $\bar{A}_{(u+t(s-1)+\theta'(p-1))}$, and $\bar{A}_u$, are selected independently and uniformly at random in $\mathbb{F}^{\frac{m}{t} \times \frac{m}{s}}$, and $\bar{B}_r$, $\bar{B}_{d+\theta' l'}$, $\bar{B}_{(v+\tau-z+1+\theta'(p'-1))}$, and $\bar{B}_v$ are chosen independently and uniformly at random in $\mathbb{F}^{\frac{m}{s} \times \frac{m}{t}}$.

Proof: The proof is provided in Appendix A in [14]. □

The degrees of secret terms in Theorem 1 are selected by exploiting the "garbage terms", which are all the terms coming from the multiplication of $C_A(x)$ and $C_B(x)$, except for the terms with indices $i + t(s-1) + \theta' l,\ i, l \in \Omega_0^{t-1}$, as these terms will be used to recover $Y = A^T B$.

Workers. Worker n receives $F_A(\alpha_n)$ and $F_B(\alpha_n)$ from source 1 and source 2, and computes $H(\alpha_n) = F_A(\alpha_n) F_B(\alpha_n)$. Then, worker n calculates $G_n(x)$ as

$$G_n(x) = \sum_{i=0}^{t-1} \sum_{l=0}^{t-1} r_n^{(i,l)} H(\alpha_n) x^{i+tl} + \sum_{w=0}^{z-1} R_w^{(n)} x^{t^2+w}, \quad (12)$$

where $R_w^{(n)}$'s are selected independently and uniformly at random from $\mathbb{F}^{\frac{m}{t} \times \frac{m}{t}}$, and $r_n^{(i,l)}$'s are obtained satisfying $\sum_{j=0}^{s-1} A_{ij} B_{jl} = \sum_{n=1}^{N} r_n^{(i,l)} H(\alpha_n)$ using the Lagrange interpolation rule, and known by all workers.

Next, worker n shares $G_n(\alpha_{n'})$ with other workers n'. After all the communications among workers, each worker n' has access to all $G_n(\alpha_{n'})$'s. Worker n' computes the summation of all $G_n(\alpha_{n'})$'s, and sends this result, *i.e.,* $I(\alpha_{n'})$, to the master node, where $I(x) = \sum_{n=1}^{N} G_n(x)$.

Master. The master node can reconstruct the polynomial $I(x)$ by receiving $\deg(I(x))+1 = t^2+z$ results from workers, and it directly gives the desired output $Y = A^T B$. The reason is that the coefficients of the first t^2 terms of $I(x)$ are exactly equal to the elements of the final result $Y = A^T B$.

Theorem 2: The required number of workers for multiplication of two massive matrices A and B employing PolyDot-CMPC, in a privacy preserving manner while there exist z colluding workers in the system and due to the resource limitations each worker is capable of working on at most $\frac{1}{st}$ fraction of each input matrix, is expressed as follows

$$N_{\text{PolyDot-CMPC}} = \begin{cases} \psi_1, & ts < z \text{ or } t = 1 \\ \psi_2, & ts - t < z \leq ts \text{ and } t, s \neq 1 \\ \psi_3, & ts - 2t < z \leq ts - t \text{ and } t, s \neq 1 \\ \psi_4, & \upsilon' < z \leq ts - 2t \text{ and } t, s \neq 1 \\ \psi_5, & z \leq \upsilon' \text{ and } t, s \neq 1 \\ \psi_6, & s = 1 \text{ and } t \geq z \text{ and } t \neq 1 \end{cases} \quad (13)$$

where $\psi_1 = (p+2)ts + \theta'(t-1) + 2z - 1$, $\psi_2 = 2ts + \theta'(t-1) + 3z - 1$, $\psi_3 = 2ts + \theta'(t-1) + 2z - 1$, $\psi_4 = (t+1)ts + (t-1)(z+t-1) + 2z - 1$, $\psi_5 = \theta' t + z$, and $\psi_6 = t^2 + 2t + tz - 1$, $s|m$, and $t|m$ are satisfied, $p = \min\{\lfloor \frac{z-1}{\theta'-ts} \rfloor, t-1\}$, $\theta' = 2ts - t$ and $\upsilon' = \max\{ts - 2t - s + 2, \frac{ts-2t+1}{2}\}$.

Proof: The proof is provided in Appendix B in [14]. □

PolyDot-CMPC satisfies privacy requirements stated in Section II. The proof directly follows from [6] (Theorem 3).

B. PolyDot-CMPC in Perspective

This section provides a theoretical analysis for the number of workers required by PolyDot-CMPC as compared to the baselines; Entangled-CMPC [7], SSMM [12] and GCSA-NA [13][3].

Lemma 3: PolyDot-CMPC is more efficient than Entangled-CMPC with regards to requiring smaller number of workers in the following regions:

1) $z > ts,\ p < \frac{t-1}{s},\ t \neq 1$
2) $ts - s < z \leq ts,\ t - 1 > s,\ s, t \neq 1$
3) $(t-1)^2 < z < t(t-1),\ s = t - 1,\ s, t \neq 1$
4) $ts - t - \min\{0, 1 - \frac{2s-5}{t-3}\} < z \leq ts - s,\ t > 3,\ s \neq 1$
5) $s = 2,\ t = 3,\ z = 4$
6) $t = 2,\ s = 2,\ z = 1, 2$
7) $\max\{st - t - s - \frac{2}{t-2}, ts - 2t\} < z \leq ts - t,\ t > 2,\ t \geq s,\ s \neq 1$
8) $t < s \leq 2t,\ ts - s < z \leq ts - t,\ s, t \neq 1$
9) $t = 2,\ 3 \leq s \leq 4,\ 2(s-2) < z \leq 2(s-1)$
10) $st - 2t < z \leq ts - s,\ t > 2,\ t < s \leq 2t$
11) $s > 2t,\ ts - 2t < z \leq ts - t,\ s, t \neq 1$
12) $2t \geq s,\ \max\{ts - 2t - s + 2, \frac{ts-2t+1}{2}\} < z \leq \min\{st - 2t, 2ts - t^2 + t - 2s + 1\},\ s, t \neq 1$
13) $s > 2t,\ ts - s < z \leq ts - 2t,\ t \neq 1, 2$
14) $4 < s < z < 2s - 4,\ t = 2$
15) $ts - 2t - s + 2 < z < ts - s,\ 2t < s,\ s, t \neq 1$
16) $st - 2s - t - \frac{1}{t-1} < z \leq \max\{ts - 2t - s + 2, \frac{ts-2t+1}{2}\},\ s, t \neq 1.$

In all other regions for the values of the system parameters s, t, and z, PolyDot-CMPC requires the same or larger number of workers.

Proof: The proof is provided in Appendix C.A in [14]. □

Lemma 4: PolyDot-CMPC performs better than SSMM in terms of requiring smaller number of workers in the following two regions:

1) $z > \max\{ts, ts - t + \frac{pts}{t-1}\}, t \neq 1$
2) $\frac{t-1}{t-2}(st - t) < z \leq ts.$

In all other regions for the values of the system parameters s, t, and z, PolyDot-CMPC requires the same or larger number of workers.

Proof: The proof is provided in Appendix C.B in [14]. □

Lemma 5: PolyDot-CMPC performs better than GCSA-NA in terms of requiring smaller number of workers in the following regions:

1) $z > ts, p < \frac{t-1}{s}, t \neq 1$
2) $s < t, ts - t < z \leq \min\{ts, t(t-1) - 1\}$
3) $z \leq ts - t$
4) $s = 1, t > z, t \neq 2.$

[3]GCSA-NA is constructed for batch matrix multiplication. However, by considering the number of batches as one, it becomes an appropriate baseline to compare PolyDot-CMPC.

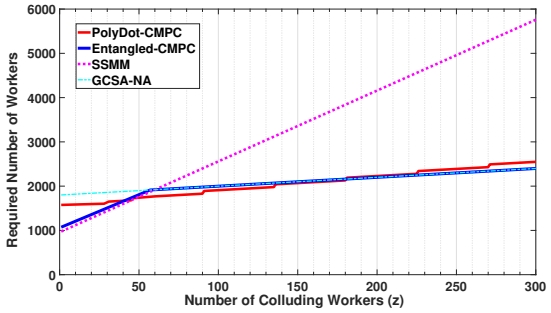

Fig. 1. Required number of workers versus number of colluding workers. The parameters are set to $s = 4,\ t = 15$ and $1 \leq z \leq 300$.

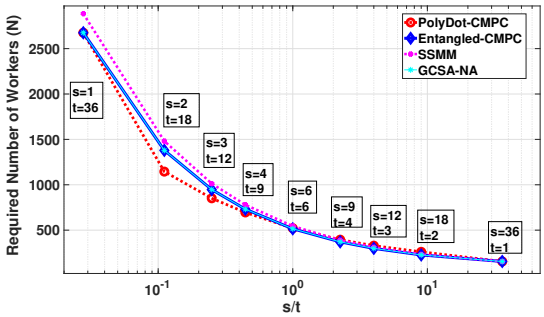

Fig. 2. Required number of workers versus s/t for fixed $z = 42$ and $st = 36$.

In all other regions for the values of the system parameters s, t, and z, PolyDot-CMPC requires the same or larger number of workers.

Proof: The proof is provided in Appendix C.C in [14]. □

V. Performance Evaluation

In this section, the performance of PolyDot-CMPC is evaluated via simulations and compared with the baseline methods, (i) Entangled-CMPC [7], (ii) SSMM [12], and (iii) GCSA-NA [13]. In this setup we have $m = 36000$, *i.e.,* both A and B are square matrices with the size of 36000×36000.

Fig. 1 shows the number of workers required for computing $Y = A^T B$ versus the number of colluding workers, where $s = 4,\ t = 15$, and $1 \leq z \leq 300$. For small number of colluding workers, *i.e.,* $1 \leq z \leq 48$, SSMM [12] performs the best as it requires minimum number of workers. PolyDot-CMPC performs better than all the baselines when $49 \leq z \leq 180$. On the other hand, GCSA-NA [13] and Entangled-CMPC [7] have similar performance and perform better than the other mechanisms when $181 \leq z \leq 300$. These results confirm Lemmas 3, 4, and 5 as PolyDot-CMPC performs better than the baselines for a range of colluding workers.

Fig. 2 illustrates the required number of workers versus s/t, the number of row partitions divided by the number of column partitions, for fixed $z = 42$ and $st = 36$. As seen, PolyDot-CMPC performs better than the other baseline methods concerning the required number of workers for $(s,t) \in \{(2,18),(3,12),(4,9)\}$, since in this scenario we have $42 = z > ts = 36$, and for these values of s, t, we have p equal to 2, 1 and 1, respectively. Thus, conditions 1 in Lemmas 3, 4, and 5 are satisfied. However, for $(s,t) \in \{(1,36),(6,6),(9,4),(12,3),(18,2),(36,1)\}$, these conditions are no longer satisfied.

VI. Conclusion

We have studied the problem of privacy preserving matrix multiplication in edge networks using MPC. We have proposed a new coded privacy-preserving computation mechanism; PolyDot-CMPC, which is designed by employing PolyDot codes. We have used "garbage terms" that naturally arise when polynomials are constructed in the design of PolyDot-CMPC to reduce the number of workers needed for privacy-preserving computation. We have analyzed and simulated PolyDot-CMPC, and demonstrated that the garbage terms are important in the design and efficiency of CMPC algorithms.

References

[1] J. Saia and M. Zamani, "Recent results in scalable multi-party computation," in *SOFSEM 2015: Theory and Practice of Computer Science*, G. F. Italiano, T. Margaria-Steffen, J. Pokorný, J.-J. Quisquater, and R. Wattenhofer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 24–44.

[2] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*, 2019, pp. 351–371.

[3] U. Maurer, "Information-theoretic cryptography," in *Advances in Cryptology — CRYPTO' 99*, M. Wiener, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 47–65.

[4] A. C.-C. Yao, "How to generate and exchange secrets," in *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, 1986, pp. 162–167.

[5] S. M. O. Goldreich and A. Wigderson, "How to play any mental game," in *Proc. of the 19th STOC*, 1987, pp. 218–229.

[6] H. Akbari-Nodehi and M. A. Maddah-Ali, "Secure coded multi-party computation for massive matrix operations," *IEEE Transactions on Information Theory*, vol. 67, no. 4, pp. 2379–2398, 2021.

[7] H. A. Nodehi, S. R. H. Najarkolaei, and M. A. Maddah-Ali, "Entangled polynomial coding in limited-sharing multi-party computation," in *2018 IEEE Information Theory Workshop (ITW)*, 2018, pp. 1–5.

[8] 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, March 2018.

[9] 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, Jan 2018.

[10] M. Fahim, H. Jeong, F. Haddadpour, S. Dutta, V. Cadambe, and P. Grover, "On the optimal recovery threshold of coded matrix multiplication," in *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2017, pp. 1264–1270.

[11] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Straggler mitigation in distributed matrix multiplication: Fundamental limits and optimal coding," *IEEE Transactions on Information Theory*, vol. 66, no. 3, pp. 1920–1933, 2020.

[12] J. Zhu, Q. Yan, and X. Tang, "Improved constructions for secure multi-party batch matrix multiplication," *IEEE Transactions on Communications*, vol. 69, pp. 7673–7690, 2021.

[13] Z. Chen, Z. Jia, Z. Wang, and S. A. Jafar, "Gcsa codes with noise alignment for secure coded multi-party batch matrix multiplication," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 1, pp. 306–316, 2021.

[14] E. Vedadi, Y. Keshtkarjahromi, and H. Seferoglu, "Polydot coded privacy preserving multi-party computation at the edge," 2021. [Online]. Available: https://arxiv.org/abs/2106.08290