# 데이터 프라이버시를 보장하는 부호화 분산 컴퓨팅 기법

양희철, 홍상우*, 이정우*
충남대학교, *서울대학교

hcyang@cnu.ac.kr, *tkddn0606@snu.ac.kr, *junglee@snu.ac.kr

# Privacy-Preserving Coded Computation in Distributed Computing

Heecheol Yang, Sangwoo Hong*, Jungwoo Lee*
Chungnam National University, *Seoul National University

요 약

본 논문은 분산 컴퓨팅을 수행하는 서버(server)로부터 정보 이론적 데이터 프라이버시를 보장하는 부호화 분산 컴퓨팅 기법을 제안한다. 분산 컴퓨팅에서 느리게 연산을 처리하는 낙오 서버(straggler)의 영향을 줄이기 위해 연산량이 적은 다수의 연산 업무를 각 서버에 할당하는 환경을 가정하였다. 마스터(master)가 행렬 곱 연산을 수행하는 분산 컴퓨팅 시스템에서 데이터 프라이버시를 서버로부터 보장하기 위해 행렬 간 대칭성이 보장된 쿼리(query)를 생성하여 서버에 전송한다. 이를 통해 낙오 서버의 영향을 감소하면서 동시에 데이터 프라이버시를 보장할 수 있음을 밝혔다.

## Ⅰ. 서 론

본 논문에서는 마스터(master)가 다수의 서버(server)를 활용하여 행렬 곱 연산을 병렬적으로 처리하는 분산 컴퓨팅 환경을 다룬다. 마스터가 다수 서버를 활용한 분산 컴퓨팅을 수행할 경우, 연산 결과를 느리게 처리하는 낙오 서버(straggler)에 의한 낙오 효과(straggling effect)로 분산 컴퓨팅의 전체 연산 처리 성능이 저하된다는 연구 결과가 발표되었다. [1] 이를 해결하기 위해, 오류 정정 부호 기반의 연산 업무 중복 할당 기법이 제안되었으며, 개별 서버가 처리하는 연산량이 증가하더라도 낙오 효과의 감소로 인해 연산 처리 시간이 감소됨을 보였다. [2] 또한, 낙오 서버의 일부 연산 결과를 최대한 활용하여 분산 컴퓨팅 성능을 향상시키기 위해 개별 서버에 작은 크기의 다수 연산 업무를 할당하여, 개별 서버가 작은 연산을 처리할 때마다 연산 결과를 마스터에 전달할 수 있는 분산 컴퓨팅 기법이 제안되었다. [3] 이에 더해, 분산 컴퓨팅 환경에서 업무 할당을 위한 통신량을 감소시키기 위해 동일한 데이터 인코딩 결과를 여러 번 활용할 수 있는 인코딩 기법이 제안되었다. [4] 본 논문에서는 이에 더해 개별 서버로부터 데이터 프라이버시를 보장하는 부호화 분산 컴퓨팅 기법을 제안한다. 행렬 곱 연산을 다수의 서버에서 나누어 수행하는 분산 컴퓨팅 시스템에서 서버가 연산의 대상이 되는 행렬에 대한 정보를 얻을 수 없음을 보였고, 마스터에서 전체 연산 결과를 얻기 위해 필요한 연산 결과의 수인 복구 한계치를 밝혔다.

## Ⅱ. 본론

### A. 시스템 모델

본 논문에서는 마스터가 행렬 $\mathrm{A} \in \mathrm{F}^{\mathrm{N}\times\mathrm{N}}$와 라이브러리 $\mathbf{B} = \{\mathrm{B}^{(\mathrm{k})}\}_{k=1}^{K}$에 포함된 행렬 $\mathrm{B}^{(\mathrm{D})} \in \mathrm{F}^{\mathrm{N}\times\mathrm{N}}, \mathrm{D} \in \{1,2,\dots,\mathrm{K}\}$에 대한 행렬 곱 $\mathrm{C} = \mathrm{AB}^{(\mathrm{D})}$를 수행하기 위해 P개의 서버 $\{\mathrm{W_i}\}_{\mathrm{i}=1}^{\mathrm{P}}$를 활용하는 분산 컴퓨팅 시스템을 고려하였다. 마스터는 데이터 프라이버시와 낙오 효과를 고려하여 두 행렬을 인코딩하기 위한 쿼리(query)를 개별 서버에게 전달한다. 마스터는 서버 $\mathrm{W_i}$에게 L개의 연산을 할당하기 위해 L개의 쿼리 $\left\{\mathrm{Q}_{\mathrm{A,i,j}}^{(D)}, \mathrm{Q}_{\mathrm{B,i,j}}^{(D)}\right\}_{j=1}^{L}$를 전달하고, 서버 $\mathrm{W_i}$는 쿼리를 통해 행렬 A와 라이브러리 $\mathbf{B}$를 인코딩하여 L개의 연산 $\tilde{C}_{i,j} = \tilde{A}_{i,j}\tilde{B}_{i,j}, j \in \{1,2,\dots,\mathrm{L}\}$를 계산한 후, 그 결과를 마스터에게 다시 전달한다. 마스터는 전체 P개의 서버가 전달하는 PL개의 연산 결과 중 임의의 Q개의 연산 결과를 받으면 전체 연산의 결과인 C를 복구할 수 있으며, 이 때 Q를 복구 한계치(recovery threshold)로 정의한다. 전체 P개의 서버가 서로 공모하지 않는 상황에서 서버로부터 데이터 프라이버시를 보장하기 위해, 마스터가 서버 $\mathrm{W_i}$, $\mathrm{i} \in [1\!:\!\mathrm{P}]$에게 전달하는 쿼리는 아래와 같은 조건을 만족해야 한다.

$$\mathrm{I}\left(D; \left\{\mathrm{Q}_{\mathrm{A,j}}^{(D)}, \mathrm{Q}_{\mathrm{B,j}}^{(D)}\right\}_{j=1}^{L}, \mathrm{A}, \mathbf{B}\right) = 0, \forall \mathrm{i} \in [1\!:\!\mathrm{P}].$$

### B. 부호화 분산 컴퓨팅 기법

제안하는 기법에서는 행렬 A와 라이브러리 $\mathbf{B}$에 포함된 행렬 $\mathrm{B}^{(\mathrm{k})}, k \in \{1,2,\dots,\mathrm{K}\}$를 다음과 같이 나누어 표현할 수 있다고 가정한다.

$$\mathrm{A}=\begin{bmatrix}A_1\\ \vdots\\ A_m\end{bmatrix},\mathrm{B}^{(\mathrm{k})}=\begin{bmatrix}B_1^{(k)} & \cdots & B_n^{(k)}\end{bmatrix}$$

또한, 행렬 곱 연산 인코딩을 위해 아래와 같은 조건을 만족하는 다항식 $\mathrm{f(x)}$을 사용한다.

i) $\mathrm{f(x)}$는 $(\mathrm{m}+\mathrm{L})$차 다항식이다.

ii) $\mathrm{f(x)}$는 $\mathrm{i}\in\{1,2,\dots,\mathrm{P}\}$에 대해 $\mathrm{g}(\alpha_{\mathrm{i},1})=\cdots=\mathrm{g}(\alpha_{\mathrm{i,L}})$을 만족하고 $\mathrm{d}\in\{1,2,\dots,\mathrm{K}\}-\{\mathrm{D}\}$에 대해 $\mathrm{g}(\alpha_{\mathrm{d+W},1})=\cdots=\mathrm{g}(\alpha_{\mathrm{d+W,L}})$을 만족하는 $\mathrm{L}$개의 점 집합 $\mathrm{P}+\mathrm{K}-1$개를 가진다.

이 때, 서버 $\mathrm{W_i}$가 쿼리를 통해 전달받은 정보로 인코딩한 행렬 곱 연산은 아래와 같이 표현된다.

$$\tilde{A}_{i,j}=\sum_{p=1}^{m}A_p\alpha_{\mathrm{i,j}}^{p-1}+\sum_{q=1}^{L}R_{A_q}\alpha_{\mathrm{i,j}}^{m+q-1},$$

$$\tilde{B}_{i,j}=\sum_{r=1}^{n}B_r^{(D)}g^r(\alpha_{\mathrm{i,j}})+\sum_{d=1,d\neq D}^{K}\sum_{r=1}^{n}B_r^{(d)}g^r(\alpha_{\mathrm{d+W,j}}).$$

이 때, $R_{A_q}$, $\mathrm{q}\in[1:\mathrm{L}]$는 행렬 $\mathrm{B^{(D)}}$의 프라이버시 정보를 서버로부터 지키기 위해 더하는 랜덤 행렬을 뜻한다. 행렬 $\tilde{B}_{i,j}$, $\mathrm{j}\in[1:\mathrm{L}]$는 다항식 $\mathrm{g(x)}$의 조건 ii)에 따라 모두 같은 값을 가져 서버 관점에서 라이브러리 $\mathbf{B}$에 포함된 행렬 간 대칭성이 지켜지므로 연산의 대상 행렬인 $\mathrm{B^{(D)}}$의 프라이버시 정보를 알아낼 수 없다.

서버가 계산하는 $\tilde{C}_{i,j}=\tilde{A}_{i,j}\tilde{B}_{i,j}$는 아래 다항식 $\mathrm{p}(x)$의 $\mathrm{x}=\alpha_{\mathrm{i,j}}$에서의 값과 같다.

$$\mathrm{p(x)}=\left(\sum_{p=1}^{m}A_px^{p-1}+\sum_{q=1}^{L}R_{A_q}x^{m+q-1}\right)\times\left(\sum_{r=1}^{n}B_r^{(D)}g^r(\mathrm{x})+\sum_{d=1,d\neq D}^{K}\sum_{r=1}^{n}B_r^{(d)}g^r(\alpha_{\mathrm{d+W,j}})\right).$$

다항식 $\mathrm{p}(x)$는 $((\mathrm{m}+\mathrm{L})(\mathrm{n}+1)-1)$차 다항식이므로 마스터는 이에 대한 $(\mathrm{m}+\mathrm{L})(\mathrm{n}+1)$개 점에서의 값으로부터 다항식의 모든 계수를 구할 수 있다. 다항식 $\mathrm{p}(x)$의 계수에 $\mathrm{C}=\mathrm{AB^{(D)}}$를 복구하기 위한 $\mathrm{mn}$개의 행렬 곱 $A_pB_r$, $\mathrm{p}\in[1:\mathrm{m}],\mathrm{r}\in[1:\mathrm{n}]$이 포함되어 있다. 따라서 마스터는 서버가 전달하는 $(\mathrm{m}+\mathrm{L})(\mathrm{n}+1)$개의 연산 결과를 통해 $\mathrm{C}=\mathrm{AB^{(D)}}$를 구할 수 있다. 따라서, 제안하는 데이터 인코딩 기법의 복구 한계치 $\mathrm{Q}$는 아래와 같다.

$$\mathrm{Q}=(\mathrm{m}+\mathrm{L})(\mathrm{n}+1)$$

## ACKNOWLEDGMENT

이 연구는 한국연구재단 이공분야기초연구사업(NRF-2020R1G1A100375)의 연구결과로 수행되었음.

## 참 고 문 헌

[1] J. Dean and L. A. Barroso, “The tail at scale,” *Communications of the ACM*, vol. 56, no. 2, pp. 74-80, Feb. 2013.

[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, Mar. 2018.

[3] S. Kiani, N. Ferdinand, and S. C. Draper, “Exploitation of stragglers in coded computation,” *in Proc. IEEE International Symposium on Information Theory (ISIT)*, Jun. 2018.

[4] S. Hong, H. Yang, and J. Lee, “Squeezed polynomial codes: Communication-efficient coded computation in straggler-exploiting distributed matrix multiplication,” IEEE Access, vol. 8, pp. 190516-190528, Oct. 2020.