논문 20-45-10-20 The Journal of Korean Institute of Communications and Information Sciences '20-10 Vol.45 No.10
https://doi.org/10.7840/kics.2020.45.10.1804

데이터 보안을 보장하는 분산 컴퓨팅에서 통신량과 복구 한계치 사이의 이론적 Trade-Off

양 희 철◆°

A Fundamental Trade-Off between Communication Load and Recovery Threshold in Secure Distributed Computing

Heecheol Yang◆°

요 약

본 논문에서는 데이터 보안을 보장하며 행렬 곱 연산을 수행하는 분산 컴퓨팅 환경에서 컴퓨팅 시스템 내의 통신량과 전체 연산 결과를 복구하기 위한 복구 한계치 사이의 trade-off 관계를 다룬다. 우선, 전체 워커 중 임의의 워커들이 행렬에 대한 정보를 알아내기 위해 서로 할당받은 연산 업무에 대한 정보를 공유하는 상황에서 데이터 보안을 보장하는 분산 행렬 곱 기법을 제안하고 이에 따라 달성 가능한 통신량과 복구 한계치 짝의 영역을 도출한다. 또한, 정해진 통신량 하에서 달성 가능한 복구 한계치의 정보 이론적 한계를 유도한다. 추가적으로, 시뮬레이션을 통해 통신량과 복구 한계치 사이의 trade-off가 여러 가지 환경에서 마스터가 연산 업무를 할당하고 연산 결과를 기다리는 대기 시간에 어떠한 영향을 끼치는지 살펴보았다.

Key Words : Distributed computing, data security, straggling effects

ABSTRACT

In this paper, we consider a fundamental trade-off between communication load and recovery threshold in a distributed computing performing matrix multiplication where data security should be preserved. Firstly, we propose a new distributed matrix multiplication scheme in which multiple workers can collude to acquire information about input data, and derive the achievable pairs of communication load and recovery threshold. We also derive a fundamental limits on recovery threshold for given communication load in an information-theoretic sense. In addition, we reveal the distribution of the waiting time at the master in various distributed computing scenarios to show the effects of the tradeoff between communication load and recovery threshold.

Ⅰ. 서 론

머신 러닝 기법의 발달에 따라 대용량 데이터의 활용도가 점차 높아지면서 여러 개의 컴퓨팅 노드 또는 워커(worker)를 활용한 분산 컴퓨팅에 대한 관심이 늘어나고 있다. 분산 컴퓨팅에서는 하나의 연산 업무를 여러 개의 워커가 나누어 처리함으로써 전체 연산을 수행하는 속도를 높일 수 있다. 하지만 이러한 분산

※ 이 연구는 금오공과대학교 학술연구비로 지원되었음(2019-104-169).
◆ First and Corresponding Author : Kumoh National University of Technology, School of Electrical Engineering, hc.yang@kumoh.ac.kr, 조교수, 정회원
논문번호 : 202007-160-B-RE, Received July 20, 2020; Revised August 10, 2020; Accepted August 18, 2020

컴퓨팅 환경에서는 연산을 가장 늦게 처리하는 워커가 연산을 완료할 때까지 기다려야하기 때문에 전체 성능이 저하되는 문제가 발생할 수 있다.[1] 이러한 현상을 낙오 효과(straggling effect)라고 한다. 또한, 여러 개의 워커에 연산 업무를 나누어 제공하고, 연산을 마친 워커가 연산 결과를 반환하는 과정에서 생기는 통신 부하 또한 분산 컴퓨팅의 속도를 저하시키는 요인이 될 수 있다.

본 논문에서는 행렬 곱 연산을 수행하는 분산 컴퓨팅 환경을 다룬다. 행렬 곱 연산을 여러 개의 워커가 나누어 수행하는 경우 연산을 부호화하여 각 워커에 할당함으로써 낙오 효과를 줄일 수 있다는 것이 밝혀졌다.[2] 예를 들어, N개의 워커가 존재하는 분산 컴퓨팅 환경에서 연산 행렬을 (N,K) maximum distance separable (MDS) 부호를 사용하여 부호화할 경우, 전체 N개의 워커 중 K개 워커의 결과만으로도 전체 행렬 곱 연산의 결과를 복구할 수 있다. 이러한 결과를 바탕으로 분산 컴퓨팅에서 행렬 곱 연산을 수행하는 경우 전체 연산 결과를 얻는 속도를 높이기 위해 필요로 하는 워커의 연산 결과 수, 즉 복구 한계치(recovery threshold)와 연산 할당 및 연산 결과 반환에 필요한 통신량에 관해 다양한 연구가 이뤄졌다.[3-5]

본 논문에서는 추가적으로 연산 대상인 행렬에 대한 정보를 워커에게 공개하지 않도록 데이터 보안을 보장하는 분산 컴퓨팅 환경을 고려하였다. 특히, 분산 컴퓨팅 환경에서는 데이터 보안이 더욱 더 중요할 수 있는데, 이는 연산 업무를 워커에 할당하고 연산 결과를 반환하는 과정에서 데이터 전송을 엿듣는 도청자가 존재할 수 있기 때문이다. 또한, 분산 컴퓨팅 환경에 따라 데이터 유출에 대한 위험이 있으나 계산 능력을 활용하고자 하는 워커가 존재하는 경우 데이터 보안이 보장되어야 할 것이다. 최근에는 개인의 위치 정보나 의료 정보 등 개인 정보 보호가 필요한 데이터에 대한 머신 러닝의 활용도가 높아져 데이터 보안의 중요성이 점차 증가하고 있다. 이에 따라 데이터 보안을 보장하는 분산 컴퓨팅 환경에서 행렬 곱 연산을 수행하는 기법에 관한 연구가 꾸준히 진행되고 있다.[6]-[10]

본 논문에서는 구체적으로 데이터 보안을 보장하며 행렬 곱 연산을 수행하는 분산 컴퓨팅 환경에서 컴퓨팅 시스템 내의 통신량과 전체 연산 결과를 복구하기 위한 복구 한계치 사이의 정보 이론적 trade-off 관계를 다뤘다.[1] 특히, 전체 워커 중 임의의 워커들이 행렬에 대한 정보를 알아내기 위해 서로 할당받은 연산 업무에 대한 정보를 공유하는(colluding) 상황에서 데이터 보안을 보장하고자 한다. 우선, 데이터 보안을 보장하는 분산 행렬 곱 기법을 제안하고 이에 따라 달성 가능한 연산 결과 반환 통신량과 복구 한계치 짝의 영역을 최초로 도출하였다. 또한, 정해진 통신량 하에서 달성 가능한 복구 한계치의 정보 이론적 한계를 유도하였다. 이에 더해 분산 컴퓨팅의 각 과정, 즉 마스터(master)가 연산 업무를 부호화하고 이를 개별 워커에 전송하며 워커가 할당된 연산을 수행하고 연산 결과를 다시 마스터에 반환하여 전체 연산 결과를 복구하는 과정에서 필요로 하는 마스터와 워커의 연산 복잡도와 계산량을 분석하였다. 마지막으로 여러 가지 환경에서 마스터가 연산 업무를 할당하고 연산 결과를 기다리는 대기 시간(waiting time)에 대한 실험을 수행하여 통신량과 복구 한계치 사이의 trade-off가 마스터의 대기 시간에 어떠한 영향을 끼치는지 살펴보았다.

본 논문에서 두 정수 a,b에 대해 $[a:b]$는 집합 $\{a,a+1,...,b\}$를 나타낸다. 또한, 행렬 집합 $\{A_1,A_2,...,A_n\}$은 $A_{[1:n]}$으로 축약해서 표기하기로 한다.

Ⅱ. 본 론

본 장에서는 본 논문에서 다루는 데이터 보안이 보장되는 분산 컴퓨팅 시스템에서 행렬 곱 연산을 수행하는 환경에 대해 소개하고, 데이터 보안을 보장하는 새로운 분산 행렬 곱 기법을 제안한다. 또한, 정해진 통신량 하에서 얻을 수 있는 복구 한계치의 정보 이론적 한계를 밝히고자 한다.

2.1 시스템 모델

본 논문에서는 마스터가 보안 데이터 $A\in F^{N\times N}$과 $B\in F^{N\times N}$에 대한 행렬 곱 연산 $C=AB$을 수행하고자 하는 분산 컴퓨팅 시스템을 고려하였다. 마스터는 행렬 곱 연산을 여러 개의 작은 연산으로 나누어 P개의 워커 $\mathrm{W}_1,...,\mathrm{W}_\mathrm{P}$에 할당한다. 개별 워커는 행렬 A,B의 원소들의 N^2/m개의 선형 결합을 저장할 수 있다. 구체적으로, 마스터는 워커 W_n에게 연산 $\widetilde{C}_n=\widetilde{A}_n\widetilde{B}_n$을 할당한다. 우선, 부호화된 A,B의 부행렬 집합을 아래와 같이 정의한다.

$$\widetilde{\boldsymbol{A}}=\widetilde{\mathrm{A}}_1,\widetilde{\mathrm{A}}_2,...,\widetilde{\mathrm{A}}_\mathrm{P},\ \widetilde{\boldsymbol{B}}=\widetilde{\mathrm{B}}_1,\widetilde{\mathrm{B}}_2,...,\widetilde{\mathrm{B}}_\mathrm{P} \tag{1}$$

1) 본 논문 내용의 일부는 2020년도 한국통신학회 동계학술발표회에서 발표되었음.[11,12]

만약 E개의 서로 정보를 공유하는 워커가 $\widetilde{\boldsymbol{A}}$, $\widetilde{\boldsymbol{B}}$ 중 임의의 E개의 원소를 통해 행렬 A,B에 대한 정보를 얻어낼 수 없을 때, 행렬 A,B는 안전하게 부호화(encoding)되었다고 가정한다. 이를 수식으로 나타내면 다음과 같다.

$$I(\widetilde{A}_{\varepsilon};A)=0, I(\widetilde{B}_{\varepsilon};B)=0,\\ \text{for } \varepsilon\subset[1:P], |\varepsilon|=E. \tag{2}$$

모든 $n\in[1:P]$에 대한 부행렬 $\widetilde{A}_n, \widetilde{B}_n$는 워커의 데이터 저장 한계를 만족하여야 한다. 즉, 부행렬 $\widetilde{A}_n, \widetilde{B}_n$의 원소의 개수는 N^2/m보다 클 수 없다. 워커 W_n은 할당된 연산 $\widetilde{C}_n=\widetilde{A}_n\widetilde{B}_n$을 계산하여 결과를 마스터에게 전송한다. 마스터는 낙오 효과에 대처하기 위해 전체 워커 중 계산 결과를 빠르게 전송하는 일부 워커의 결과만을 기다리며, 주어진 일부 워커의 연산 결과를 이용해 최종 연산 결과 $C=AB$를 복구한다. 그림 1은 이러한 분산 컴퓨팅 시스템을 묘사하고 있다.

행렬 곱 연산을 위한 데이터 보안을 보장하는 분산 컴퓨팅 시스템에서 달성 가능한 통신량과 복구 한계치의 집합을 정의하고자 한다. 안전하게 부호화된 행렬 A,B에 대해, 통신량 $L_{(\widetilde{\boldsymbol{A}},\widetilde{\boldsymbol{B}})}$는 워커 W_n에서 마스터로 전송하는 연산 결과 $\widetilde{C}_n$의 원소의 개수로 정의하였다. 또한, 복구 한계치 $K_{(\widetilde{\boldsymbol{A}},\widetilde{\boldsymbol{B}})}$은 마스터가 최종 연산 결과 C를 복구할 수 있는 임의의 연산 결과 $\widetilde{C}_n$의 최소 개수로 정의하였다. 만약 통신량 L과 복구 한계치 K를 동시에 만족하는 부호화된 A,B의 부행렬 집합 $\widetilde{\boldsymbol{A}}$, $\widetilde{\boldsymbol{B}}$이 존재한다면 통신량 L과 복구 한계치 K의 짝 (L,K)가 달성 가능하다고 정의하였다. 달성 가능한 (L,K)의 영역 $R(L,K)$은 다음과 같이 나타낼 수 있다.

$$R(L,K)=\{(L,K)\,|\,(L,K)\text{ is feasible for }given\ (\widetilde{\boldsymbol{A}},\widetilde{\boldsymbol{B}})\} \tag{3}$$

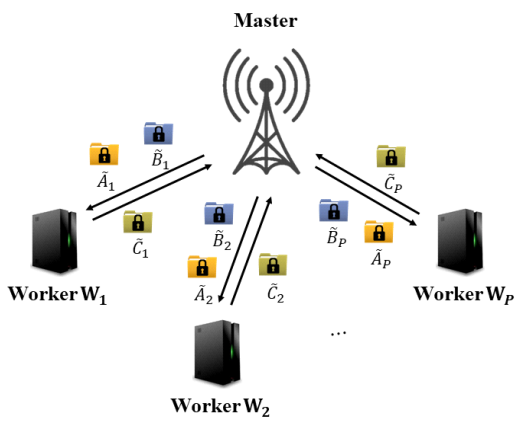

그림 1. 데이터 보안을 보장하는 분산 컴퓨팅 시스템
Fig. 1. Secure distributed computing system

2.2 데이터 보안을 보장하는 분산 행렬 곱 기법

본 장에서는 $m=4$, $E=1$인 경우에 대한 예시를 우선 제시하고, 일반적인 경우에 대해 데이터 보안을 보장하는 분산 행렬 곱 기법을 제안하고자 한다.

2.2.1 기본 예시 ($m=4, E=1$)

기본 예시 $(m=4,E=1)$에 대해 행렬을 나누는 변수 t를 변경해가며 분산 행렬 곱 기법에서 달성 가능한 통신량 L과 복구 한계치 K를 보이고자 한다.

(1) 예시 1. $t=4$

행렬을 나누는 변수 $t=4$일 경우, 행렬 A,B를 아래와 같이 나눈다.

$$A=\begin{bmatrix}A_{1,1}\\A_{2,1}\\A_{3,1}\\A_{4,1}\end{bmatrix},\ B=[B_{1,1}\,B_{1,2}\,B_{1,3}\,B_{1,4}]. \tag{4}$$

최종 연산 결과 C는 아래와 같이 표현된다.

$$C=AB=\begin{bmatrix}A_{1,1}B_{1,1}&A_{1,1}B_{1,2}&A_{1,1}B_{1,3}&A_{1,1}B_{1,4}\\A_{2,1}B_{1,1}&A_{2,1}B_{1,2}&A_{2,1}B_{1,3}&A_{2,1}B_{1,4}\\A_{3,1}B_{1,1}&A_{3,1}B_{1,2}&A_{3,1}B_{1,3}&A_{3,1}B_{1,4}\\A_{4,1}B_{1,1}&A_{4,1}B_{1,2}&A_{4,1}B_{1,3}&A_{4,1}B_{1,4}\end{bmatrix}. \tag{5}$$

마스터는 워커 W_n에게 다음과 같은 연산 $\widetilde{C}_n=\widetilde{A}_n\widetilde{B}_n$을 할당한다.

$$\widetilde{A}_n=A_{1,1}+A_{2,1}x_n+A_{3,1}x_n^2+A_{4,1}x_n^3+R_Ax_n^4\\ \widetilde{B}_n=B_{1,1}+B_{1,2}x_n^5+B_{1,3}x_n^{10}+B_{1,4}x_n^{15}+R_Bx_n^{19} \tag{6}$$

행렬 R_A, R_B는 행렬 A,B의 정보를 워커로부터 보호하기 위해 생성된 랜덤 행렬이다. 마스터가 각각 워커에 할당하는 부행렬 $\widetilde{A}_n, \widetilde{B}_n$의 크기는 $\widetilde{A}_n\in F^{N/4\times N}$, $\widetilde{B}_n\in F^{N\times N/4}$이므로 워커의 저장 용량 제한을 만족한다.

워커 W_n은 연산 $\widetilde{C}_n = \widetilde{A}_n \widetilde{B}_n$을 계산한 후 연산 결과 $\widetilde{C}_n$을 마스터에게 전송한다. 연산 결과 $\widetilde{C}_n$는 모두 x에 관한 23차 다항식이고, 24개의 계수 중 16개는 최종 연산 결과 C의 16개의 부행렬 원소에 해당한다. 따라서, 마스터는 워커의 연산 결과 중 가장 빠른 24개의 결과를 통해 최종 연산 결과 C를 복구할 수 있다. 이 경우 복구 한계치 K는 24이다. 또한, 워커가 마스터에 전송하는 연산 결과 $\widetilde{C}_n$의 크기는 $\widetilde{C}_n \in F^{N/4 \times N/4}$이므로 통신량 L은 $N^2/16$이다. 결론적으로, 행렬을 나누는 변수 $t=4$일 때, 달성 가능한 통신량 L과 복구 한계치 K의 짝 (L,K)는 $(L,K)=(N^2/16, 24)$이다.

(2) 예시 2. $t=1$

행렬을 나누는 변수 $t=1$일 경우, 행렬 A,B를 아래와 같이 나눈다.

$$A=[A_{1,1}\,A_{1,2}\,A_{1,3}\,A_{1,4}],\ B=\begin{bmatrix}B_{1,1}\\B_{2,1}\\B_{3,1}\\B_{4,1}\end{bmatrix}. \tag{7}$$

위와 같이 행렬 A,B를 나눌 경우, 최종 연산 결과 C는 다음과 같이 표현된다.

$$\begin{aligned}C&=\left[A_{1,1}B_{1,1}+A_{1,2}B_{2,1}+A_{1,3}B_{3,1}+A_{1,4}B_{4,1}\right]\\&=[\sum_{j=1}^{4}A_{1,j}B_{j,1}]\end{aligned} \tag{8}$$

예시 1과 달리, 마스터는 워커 W_n에게 다음과 같은 연산 $\widetilde{C}_n = \widetilde{A}_n \widetilde{B}_n$을 할당한다.

$$\begin{aligned}\widetilde{A}_n &= A_{1,1}+A_{1,2}x_n+A_{1,3}x_n^2+A_{1,4}x_n^3+R_A x_n^4\\ \widetilde{B}_n &= B_{1,1}x_n^3+B_{2,1}x_n^2+B_{3,1}x_n+B_{4,1}+R_B x_n^4\end{aligned} \tag{9}$$

마스터가 각각 워커에 할당하는 부행렬 $\widetilde{A}_n, \widetilde{B}_n$의 크기는 $\widetilde{A}_n \in F^{N\times N/4}$, $\widetilde{B}_n \in F^{N/4\times N}$이므로 예시 1과 마찬가지로 워커의 저장 용량 제한을 만족한다.

워커 W_n의 연산 결과 $\widetilde{C}_n = \widetilde{A}_n \widetilde{B}_n$에서 x^3의 계수가 최종 연산 결과 C와 일치하는 것을 확인할 수 있다. 따라서, 마스터가 C를 복구하기 위해서는 x에 관한 8차 다항식 $\widetilde{C}_n$의 9개의 연산 결과를 워커로부터 받아 다항식의 계수를 구해야 한다. 따라서, 마스터는 워커의 연산 결과 중 가장 빠른 9개의 결과를 통해 최종 연산 결과 C를 복구할 수 있으며 복구 한계치 K는 9이다. 또한, 워커가 마스터에 전송하는 연산 결과 $\widetilde{C}_n$의 크기는 $\widetilde{C}_n \in F^{N\times N}$이므로 통신량 L은 N^2이다. 결론적으로, 행렬을 나누는 변수 $t=1$일 때, 달성 가능한 통신량 L과 복구 한계치 K의 짝 (L,K)는 $(L,K)=(N^2,9)$이다.

(3) 예시 3. $t=2$

마지막으로, 행렬을 나누는 변수 $t=2$일 경우 달성 가능한 통신량 L과 복구 한계치 K를 보이고자 한다. 이 경우, 행렬 A,B를 아래와 같이 나눈다.

$$A=\begin{bmatrix}A_{1,1}\,A_{1,2}\\A_{2,1}\,A_{2,2}\end{bmatrix},\ B=\begin{bmatrix}B_{1,1}\,B_{1,2}\\B_{2,1}\,B_{2,2}\end{bmatrix}. \tag{10}$$

최종 연산 결과 C는 아래와 같이 표현된다.

$$\begin{aligned}C&=\begin{bmatrix}A_{1,1}B_{1,1}+A_{1,2}B_{2,1}\,A_{1,1}B_{1,2}+A_{1,2}B_{2,2}\\A_{2,1}B_{1,1}+A_{2,2}B_{2,1}\,A_{2,1}B_{1,2}+A_{2,2}B_{2,2}\end{bmatrix}\\&=\begin{bmatrix}\sum_{j=1}^{2}A_{1,j}B_{j,1}&\sum_{j=1}^{2}A_{1,j}B_{j,2}\\\sum_{j=1}^{2}A_{2,j}B_{j,1}&\sum_{j=1}^{2}A_{2,j}B_{j,2}\end{bmatrix}.\end{aligned} \tag{11}$$

마스터는 워커 W_n에게 다음과 같은 연산 $\widetilde{C}_n = \widetilde{A}_n \widetilde{B}_n$을 할당한다.

$$\begin{aligned}\widetilde{A}_n &= A_{1,1}+A_{1,2}x_n+A_{2,1}x_n^2+A_{2,2}x_n^3+R_A x_n^4\\ \widetilde{B}_n &= B_{1,1}x_n+B_{2,1}+B_{1,2}x_n^6+B_{2,2}x_n^5+R_B x_n^9\end{aligned} \tag{12}$$

마스터가 각각 워커에 할당하는 부행렬 $\widetilde{A}_n, \widetilde{B}_n$의 크기는 $\widetilde{A}_n \in F^{N/2\times N/2}$, $\widetilde{B}_n \in F^{N/2\times N/2}$이므로 워커의 저장 용량 제한을 만족한다. 워커 W_n의 연산 결과 $\widetilde{C}_n = \widetilde{A}_n \widetilde{B}_n$는 다음과 같이 나타낼 수 있다.

$$\begin{aligned}\widetilde{C}_n &= A_{1,1}B_{2,1}+(A_{1,1}B_{1,1}+A_{1,2}B_{2,1})x_n\\&\quad+(A_{1,2}B_{1,1}+A_{2,1}B_{2,1})x_n^2+(A_{2,1}B_{1,1}+A_{2,2}B_{2,1})x_n^3\\&\quad+(A_{2,2}B_{1,1}+R_AB_{2,1})x_n^4+(A_{1,1}B_{2,2}+R_AB_{1,1})x_n^5\\&\quad+(A_{1,1}B_{1,2}+A_{1,2}B_{2,2})x_n^6+(A_{1,2}B_{1,2}+A_{2,1}B_{2,2})x_n^7\\&\quad+(A_{2,1}B_{1,2}+A_{2,2}B_{2,2})x_n^8+\ldots+R_AR_Bx_n^{13}\end{aligned} \tag{13}$$

마스터는 연산 결과 $\widetilde{C_n}$에서 x_n, x_n^3, x_n^6, x_n^8의 계수를 통해 전체 연산 결과 C의 4개의 부행렬 원소를 얻을 수 있다. 따라서, $\widetilde{C_n}$이 x에 관한 13차 다항식이므로 마스터가 최종 연산 결과 C를 복구하기 위해서는 가장 빠른 14개의 워커로부터 연산 결과 $\widetilde{C_n}$을 받은 후, 다항식의 계수를 얻어야 한다. 이 경우 복구 한계치 K는 14이다. 또한, 워커가 마스터에 전송하는 연산 결과 $\widetilde{C_n}$의 크기는 $\widetilde{C_n} \in F^{N/2 \times N/2}$이므로 통신량 L은 $N^2/4$이다. 결론적으로, 행렬을 나누는 변수 $t=2$일 때, 달성 가능한 통신량 L과 복구 한계치 K의 짝 (L,K)는 $(L,K)=(N^2/4, 14)$이다.

2.2.2 일반적인 경우로 확장

앞에서 보인 기본 예시에서의 데이터 보안을 보장하는 분산 행렬 곱 기법을 일반적인 경우로 확장하여 모든 m, E에 대해 행렬을 나누는 변수 t에 따라 달성 가능한 통신량 L과 복구 한계치 K를 보이고자 한다.

우선 행렬 A,B를 행렬을 나누는 변수 t, s를 활용하여 아래와 같이 표현한다.

$$A=\begin{bmatrix} A_{1,1} \cdots A_{1,s} \\ \vdots \ \ddots \ \vdots \\ A_{t,1} \cdots A_{t,s} \end{bmatrix},\ B=\begin{bmatrix} B_{1,1} \cdots B_{1,t} \\ \vdots \ \ddots \ \vdots \\ B_{s,1} \cdots B_{s,t} \end{bmatrix}. \tag{14}$$

이 때, $st=m$을 만족하도록 t, s를 결정한다. 최종 연산 결과 C는 다음과 같이 나타낼 수 있다.

$$C=\begin{bmatrix} \sum_{j=1}^{s} A_{1,j}B_{j,1} \cdots \sum_{j=1}^{s} A_{1,j}B_{j,t} \\ \vdots \ \ddots \ \vdots \\ \sum_{j=1}^{s} A_{t,j}B_{j,1} \cdots \sum_{j=1}^{s} A_{t,j}B_{j,t} \end{bmatrix}. \tag{15}$$

마스터는 워커 W_n에 아래와 같은 연산 $\widetilde{C_n}=\widetilde{A_n}\widetilde{B_n}$을 할당한다.

$$\widetilde{A_n}=\sum_{j=1}^{s}\sum_{i=1}^{t} A_{i,j}x_n^{s(i-1)+j-1}+\sum_{e=1}^{E} R_{A_e}x_n^{ts+e-1} \tag{16}$$
$$\widetilde{B_n}=\sum_{l=1}^{t}\sum_{k=1}^{s} B_{k,t}x_n^{(ts+E)(l-1)+s-k}+\sum_{e=1}^{E} R_{B_e}x_n^{t^2s+tE-e}.$$

이 때, 부행렬 $\widetilde{A_n}, \widetilde{B_n}$의 크기는 $\widetilde{A_n} \in F^{N/t \times N/s}$, $\widetilde{B_n} \in F^{N/s \times N/t}$이며, R_{A_e}, R_{B_e}는 행렬 A,B를 워커로부터 보호하기 위해 생성된 랜덤 행렬이고 그 크기는 부행렬 $\widetilde{A_n}, \widetilde{B_n}$과 같다.

위와 같이 마스터가 각각의 워커에 연산을 할당하기 위해 부행렬 $\widetilde{A_n}, \widetilde{B_n}$를 보냈을 때, 행렬 A,B의 정보가 서로 정보를 공유하는 임의의 E개의 워커로부터 보호됨을 밝히고자 한다.

보조정리 1. 행렬 A,B를 워커로부터 보호하기 위해 생성된 랜덤 행렬 $R_{A_{[1:E]}}, R_{B_{[1:E]}}$가 아래와 같은 조건을 만족하면, 서로 정보를 공유하는 E개의 워커 W_ε, $\varepsilon \subset [1:P]$, $|\varepsilon|=E$는 부행렬 집합 $\widetilde{A}_\varepsilon, \widetilde{B}_\varepsilon$로부터 행렬 A,B에 대한 정보를 얻을 수 없다. 즉, 식 (2)의 데이터 보호에 관한 제한이 만족된다.

$$\rho_{A_{[1:E]}} \geq H(\widetilde{A}_\varepsilon),\ \rho_{B_{[1:E]}} \geq H(\widetilde{B}_\varepsilon). \tag{17}$$

이 때, $\rho_{A_{[1:E]}}, \rho_{B_{[1:E]}}$는 각각 랜덤 행렬 집합의 엔트로피 $H(R_{A_{[1:E]}}), H(R_{B_{[1:E]}})$를 나타낸다.

증명: 일반성을 잃지 않고 임의의 워커 집합 W_ε, $\varepsilon \subset [1:P]$, $|\varepsilon|=E$과 행렬 A에 대해 보조정리 1을 증명하고자 한다. 아래 증명은 행렬 B에 대해서도 적용될 수 있다.

$$\begin{aligned} I(\widetilde{A}_\varepsilon;A) &= H(\widetilde{A}_\varepsilon)-H(\widetilde{A}_\varepsilon|A) \\ &= H(\widetilde{A}_\varepsilon)-H(\widetilde{A}_\varepsilon|A)+H(\widetilde{A}_\varepsilon|A,R_{A_{[1:E]}}) \\ &= H(\widetilde{A}_\varepsilon)-I(R_{A_{[1:E]}};\widetilde{A}_\varepsilon|A) \\ &= H(\widetilde{A}_\varepsilon)-H(R_{A_{[1:E]}}|A)+H(R_{A_{[1:E]}}|\widetilde{A}_\varepsilon,A) \\ &= H(\widetilde{A}_\varepsilon)-H(R_{A_{[1:E]}}|A) \\ &= H(\widetilde{A}_\varepsilon)-H(R_{A_{[1:E]}}) \end{aligned} \tag{18}$$

위 식에서 두 번째 줄과 다섯 번째 줄은 $\widetilde{A}_\varepsilon$가 행렬 A와 랜덤 행렬 집합 $R_{A_{[1:E]}}$에 관한 식이라는 사실에 기인한다. 위 식에서 만약 $H(\widetilde{A}_\varepsilon)=H(R_{A_{[1:E]}})$이면 $I(\widetilde{A}_\varepsilon;A)=0$임을 알 수 있다. 이는 만약 랜덤 행렬 집합의 엔트로피 $\rho_{A_{[1:E]}}=H(R_{A_{[1:E]}})$가 $H(\widetilde{A}_\varepsilon)$보다 크면 $I(\widetilde{A}_\varepsilon;A)=0$가 성립함을 의미한다. 따라서 $\rho_{A_{[1:E]}} \geq H(\widetilde{A}_\varepsilon)$일 때, 부행렬 집합 $\widetilde{A}_\varepsilon$로부터 행렬 A에 관한 정보를 얻을 수 없음이 증명되었다. ■

워커 W_n은 주어진 부행렬 $\widetilde{A_n}, \widetilde{B_n}$에 따라 할당된

연산 $\widetilde{C}_n = \widetilde{A}_n \widetilde{B}_n$을 수행하여 그 결과 $\widetilde{C}_n$을 마스터에게 전송한다. 계산 결과 $\widetilde{C}_n$은 다음과 같이 나타낼 수 있다.

$$\widetilde{C}_n = (\sum_{j=1}^{s}\sum_{i=1}^{t} A_{i,j} x_n^{s(i-1)+j-1} + \sum_{e=1}^{E} R_{A_e} x_n^{ts+e-1}) \times (\sum_{l=1}^{t}\sum_{k=1}^{s} B_{k,l} x_n^{(ts+E)(l-1)+s-k} + \sum_{e=1}^{E} R_{B_e} x_n^{t^2s+tE-e}) \tag{19}$$

이 때, x에 관한 다항식 $\widetilde{C}_n$의 차수는 부행렬 $\widetilde{A}_n, \widetilde{B}_n$의 최고차항 간의 곱의 차수로 구할 수 있으며 아래와 같이 계산된다.

$$\begin{aligned} x_n^{ts+E-1} \times x_n^{t^2s+tE-1} &= x_n^{(t+1)(ts+E)-2} \\ &= x_n^{(t+1)(m+E)-2} \end{aligned} \tag{20}$$

따라서, $\widetilde{C}_n$은 x에 관한 $((t+1)(m+E)-2)$-차 다항식이다. 만약 마스터가 워커로부터 $(t+1)(m+E)-1$개의 연산 결과 $\widetilde{C}_n$을 받는다면 다항식 $\widetilde{C}_n$의 모든 계수를 얻을 수 있다. 지금부터는 마스터가 $\widetilde{C}_n$의 계수로부터 최종 연산 결과 C의 t^2개의 부행렬 원소를 모두 얻을 수 있음을 보이고자 한다. 마스터가 C의 t^2개의 부행렬 원소를 얻는다면 최종 연산 결과 C를 복구할 수 있다. 각각의 워커가 전송하는 계산 결과 $\widetilde{C}_n$를 x에 관한 식으로 전개하면 아래와 같다.

$$\begin{aligned} \widetilde{C}_n = &\sum_{j=1}^{s}\sum_{i=1}^{t}\sum_{l=1}^{t}\sum_{k=1}^{s} A_{i,j}B_{k,l} x_n^{s(i-1)+j-1+(ts+E)(l-1)+s-k} \\ &+ \sum_{j=1}^{s}\sum_{i=1}^{t}\sum_{e=1}^{E} A_{i,j}R_{B_e} x_n^{s(i-1)+j-1+t^2s+tE-e} \\ &+ \sum_{l=1}^{t}\sum_{k=1}^{s}\sum_{e=1}^{E} R_{A_e}B_{k,l} x_n^{(ts+E)(l-1)+s-k+ts+e-1} \\ &+ \sum_{e=1}^{E}\sum_{e'=1}^{E} R_{A_e}R_{B_{e'}} x_n^{ts+e-1+t^2s+tE-e'} \end{aligned} \tag{21}$$

이 중, 랜덤 행렬 R_{A_e}, R_{B_e}를 포함하지 않는 항을 모아 전개하면 아래와 같이 표현된다.

$$\begin{aligned} &\sum_{j=1}^{s}\sum_{i=1}^{t}\sum_{l=1}^{t}\sum_{k=1}^{s} A_{i,j}B_{k,l} x_n^{s(i-1)+j-1+(ts+E)(l-1)+s-k} \\ &= \sum_{i=1}^{t}\sum_{l=1}^{t}(\sum_{j=1}^{s}\sum_{k=1}^{s} A_{i,j}B_{k,l} x_n^{s(t(l-1)+i)+E(l-1)+j-k-1}) \\ &= \sum_{i=1}^{t}\sum_{l=1}^{t}(\sum_{j=1}^{s} A_{i,j}B_{k,l} x_n^{s(t(l-1)+i)+E(l-1)-1} \\ &\qquad + \sum_{j=1}^{s}\sum_{k\neq j,k=1}^{s} A_{i,j}B_{k,l} x_n^{s(t(l-1)+i)+E(l-1)+j-k-1}) \end{aligned} \tag{22}$$

위 식에서 $j=k$일 때, 다항식 $\widetilde{C}_n$의 계수는 모든 $i,l \in [1:t]$에 대해 행렬 C의 부행렬 원소를 나타낸다. 또한 $-s < j-k < s$이므로 C의 부행렬 원소들은 모두 다항식 $\widetilde{C}_n$에서 독립적인 계수로 존재한다. 따라서, 마스터는 다항식 $\widetilde{C}_n$의 $j=k$이고 모든 $i,l \in [1:t]$에 해당하는 차수의 계수로부터 C의 t^2개의 부행렬 원소를 얻을 수 있다.

마스터가 워커로부터 $(t+1)(m+E)-1$개의 연산 결과 $\widetilde{C}_n$을 받아 다항식 $\widetilde{C}_n$의 모든 계수를 얻을 수 있고, 이에 따라 최종 연산 결과 C를 복구할 수 있으므로 복구 한계치 K는 $(t+1)(m+E)-1$이다. 또한, 워커가 마스터에 전송하는 연산 결과 $\widetilde{C}_n$의 크기는 $\widetilde{C}_n \in F^{N/t \times N/t}$이므로 통신량 L은 N^2/t^2이다. 결론적으로, 행렬을 나누는 변수 t에 따라 달성 가능한 통신량 L과 복구 한계치 K의 짝 (L,K)는 $(L,K) = (N^2/t^2, (t+1)(m+E)-1)$이다. 이와 같은 결과로 달성 가능한 (L,K)의 영역 $R(L,K)$에 대해 다음과 같은 정리를 도출할 수 있다.

정리 1. 마스터가 P개의 워커를 이용해 행렬 곱 연산 $C=AB$를 수행하고 서로 정보를 공유하는 임의의 E개의 워커로부터 행렬 A,B의 정보를 보호하는 분산 컴퓨팅 환경에서 각각의 워커가 A,B의 $1/m$을 저장할 수 있는 저장 용량 한계를 가질 때, 달성 가능한 (L,K)의 영역 $R(L,K)$는 다음과 같다.

$$R(L,K) = \left\{ \left(\frac{N^2}{t^2}, (t+1)(m+E)-1\right) \middle| t \in [1 : \frac{P+1}{m+E} - 1] \right\} \tag{23}$$

정리 1에 따르면, 마스터가 E개의 워커로부터 행렬 A,B의 정보를 보호하기 위해서는 정보 보호가 필요하지 않는 상황($E=0$)에 비해 같은 통신량 하에서 $(t+1)E$개의 연산 결과를 추가적으로 기다려야 한다. 따라서, 제안하는 기법에서 A,B의 정보를 보호하기 위해 필요로 하는 복구 한계치의 증가량은 $(t+1)E$

이다.

2.3 복구 한계치의 정보 이론적 한계

본 장에서는 통신량 L에 따라 달성 가능한 복구 한계치 K의 한계를 정보 이론적으로 증명하고자 한다. 복구 한계치 K의 정보 이론적 한계를 다음과 같이 정리할 수 있다.

정리 2. 마스터가 P개의 워커를 이용해 행렬 곱 연산 $C=AB$를 수행하고 서로 정보를 공유하는 임의의 E개의 워커로부터 행렬 A,B의 정보를 보호하는 분산 컴퓨팅 환경에서 각각의 워커가 A,B의 $1/m$을 저장할 수 있는 저장 용량 한계를 가질 때, 복구 한계치 K는 다음과 같은 정보 이론적 한계를 가진다.

$$K \geq \frac{N^2}{L}+E. \tag{24}$$

증명: 우선 두 행렬 A,B의 원소는 독립적이고 동일한 분포(independent and identically distributed; i.i.d.)를 가지는 랜덤 변수라 가정한다. 일반성을 잃지 않고, 마스터가 K개의 워커 $W_{[1:K]}$ 의 연산 결과로부터 전체 연산 결과 C를 복구할 수 있다고 하자. 이를 수식적으로 표현하면 다음과 같다.

$$H(C|\widetilde{C}_1,...,\widetilde{C}_K)=0. \tag{25}$$

추가적으로, 전체 연산 결과 C와 각 워커에서 계산된 연산 결과 $\widetilde{C}_n$ 사이의 상호 정보량(mutual information)은 다음과 같이 계산된다.

$$\begin{aligned} I(C;\widetilde{C}_n) &= I(AB;\widetilde{A}_n\widetilde{B}_n) \\ &= H(AB)-H(AB|\widetilde{A}_n\widetilde{B}_n) \\ &\leq H(AB)-H(AB|\widetilde{A}_n\widetilde{B}_n,\widetilde{A}_n,\widetilde{B}_n) \\ &= H(AB)-H(AB|\widetilde{A}_n,\widetilde{B}_n) \\ &\leq H(AB)-H(A|\widetilde{A}_n,\widetilde{B}_n) \\ &= H(AB)-H(A) \\ &\leq 0 \end{aligned} \tag{26}$$

위 식에서 셋째 줄은 추가 조건이 엔트로피를 감소시킨다는 사실[13]에 기인하며, 넷째 줄은 $\widetilde{A}_n\widetilde{B}_n$이 $\widetilde{A}_n$, $\widetilde{B}_n$로 결정된다는 사실에 기인한다. 또한, 여섯째 줄은 $\widetilde{A}_n$, $\widetilde{B}_n$이 행렬 A에 관한 정보를 포함하지 않는다는 데이터 보호 조건에 따른다. 상호 정보량의 정의에 따라 $I(C;\widetilde{C}_n)$은 항상 0보다 크거나 같으므로 아래 식이 성립한다.

$$I(C;\widetilde{C}_n)=0. \tag{27}$$

또한, 임의의 E개의 워커 집합 W_ε, $\varepsilon \subset [1:K]$, $|\varepsilon|=E$이 서로 정보를 공유한다고 가정할 때, 아래와 같은 식을 얻을 수 있다.

$$\begin{aligned} H(C) &= I(C;\widetilde{C}_1,...,\widetilde{C}_K)+H(C|\widetilde{C}_1,...,\widetilde{C}_K) \\ &= I(C;\widetilde{C}_1,...,\widetilde{C}_K) \\ &= H(\widetilde{C}_1,...,\widetilde{C}_K)-H(\widetilde{C}_1,...,\widetilde{C}_K|C) \\ &\leq H(\widetilde{C}_1,...,\widetilde{C}_K)-H(\widetilde{C}_\varepsilon|C) \\ &= H(\widetilde{C}_1,...,\widetilde{C}_K)-H(\widetilde{C}_\varepsilon) \end{aligned} \tag{28}$$

위 식에서 둘째 줄은 식 (25)에 의해, 다섯째 줄은 식 (27)에 의해 성립한다. 모든 E개의 워커 집합 W_ε, $\varepsilon \subset [1:K]$, $|\varepsilon|=E$에 대해 위 식을 더하여 평균을 취하면 아래와 같은 식을 유도할 수 있다.

$$\begin{aligned} H(C) &= H(\widetilde{C}_1,...,\widetilde{C}_K)-\frac{1}{\binom{K}{E}}\sum_{\varepsilon\subset[1:K],|\varepsilon|=E}H(\widetilde{C}_\varepsilon) \\ &\leq \left(1-\frac{E}{K}\right)H(\widetilde{C}_1,...,\widetilde{C}_K) \\ &\leq \left(1-\frac{E}{K}\right)\sum_{n=1}^{K}H(\widetilde{C}_n) \\ &= (K-E)H(\widetilde{C}_n) \end{aligned} \tag{29}$$

위 식에서 둘째 줄은 Han's 부등식[13]에 기인하고, 셋째 줄은 조건의 감소가 엔트로피를 감소시키지 않는다는 사실에 기인하며, 넷째 줄은 모든 $\widetilde{C}_n$이 같은 크기를 가지는 정칙(full-rank) 행렬이라는 사실에 기인한다. 통신량 L이 N^2/t^2일 때 부행렬 $\widetilde{C}_n$의 크기는 전체 행렬 C의 크기의 $1/t^2$이므로, 다음과 같은 식이 성립한다.

$$\begin{aligned} K &\geq t^2+E \\ &= \frac{N^2}{L}+E \end{aligned} \tag{30}$$

따라서, 정리 2가 증명되었다. ■

Ⅲ. 실 험

본 장에서는 데이터 보안을 보장하는 분산 컴퓨팅

시스템에서 마스터가 최종 연산 결과를 복구하는데 걸리는 전체 시간의 분포를 실험을 통해 보이고자 한다. 마스터에서 행렬 곱 연산을 부호화하는 과정과 워커로부터 받은 연산 결과를 통해 최종 연산 결과를 복구하는 과정에서 걸리는 시간은 확정적이므로, 마스터가 워커에게 연산을 할당하는데 걸리는 통신 시간, 워커가 할당된 연산을 계산하는데 걸리는 계산 시간, 워커가 마스터에게 연산 결과를 전달하는데 걸리는 통신 시간의 분포를 집중적으로 다룰 것이다.

3.1 복잡도 분석

마스터가 행렬 곱 연산을 부호화하는 과정과 최종 연산 결과를 복구하는 과정에서 소요되는 계산 복잡도를 분석하고, 각각의 워커가 연산을 수행하는데 필요한 계산량과 마스터와 워커간 통신량을 분석할 것이다.

3.1.1 연산 부호화 복잡도

마스터가 각 워커에 연산을 할당하기 위해 행렬 곱 연산을 부행렬 $\widetilde{A}_n$, $\widetilde{B}_n$로 부호화하는 과정이 필요하다. 우선, 마스터는 $\widetilde{A}_n$, $\widetilde{B}_n$을 부호화하기 위해 각각 $\frac{N}{t}\times\frac{N}{s}$ 크기의 행렬 또는 $\frac{N}{s}\times\frac{N}{t}$ 크기의 행렬을 $st+E$ 개 더한다. 이를 위한 계산 복잡도는 $O\left(2(st+E)\frac{N^2}{st}\right)=O\left(2(m+E)\frac{N^2}{m}\right)$와 같다. 따라서, 전체 P개의 워커를 위한 연산 부호화 복잡도는 $O\left(2P(m+E)\frac{N^2}{m}\right)$이다.

3.1.2 워커의 계산량

각 워커는 할당된 부행렬 $\widetilde{A}_n$, $\widetilde{B}_n$의 곱을 수행한다. $\widetilde{A}_n$, $\widetilde{B}_n$의 크기는 각각 $F^{\frac{N}{t}\times\frac{N}{s}}$ 과 $F^{\frac{N}{s}\times\frac{N}{t}}$ 이므로 행렬 곱을 위해 $\frac{N}{t}\times\frac{N}{s}\times\frac{N}{t}=\frac{N^2}{st^2}$의 계산이 필요하다. 따라서 각 워커의 계산량은 $O\left(\frac{N^3}{mt}\right)$이다.

3.1.3 마스터와 워커간 통신량

마스터는 각 워커에 연산을 할당하기 위해 각각 $F^{\frac{N}{t}\times\frac{N}{s}}$ 과 $F^{\frac{N}{s}\times\frac{N}{t}}$ 의 크기를 가지는 부행렬 $\widetilde{A}_n$, $\widetilde{B}_n$을 전송한다. 따라서, 마스터가 전체 P개의 워커에 전송하는 통신량은 $O\left(P\left(\frac{N^2}{st}+\frac{N^2}{st}\right)\right)=O\left(2P\frac{N^2}{m}\right)$이다. 각 워커는 연산 결과 $\widetilde{C}_n$을 마스터에 전송하며, $\widetilde{C}_n$의 크기는 $F^{\frac{N}{t}\times\frac{N}{t}}$ 와 같다. 마스터는 가장 빠른 $(t+1)(m+E)-1$개의 워커의 연산 결과만을 기다리므로 워커에서 마스터로 연산 결과를 전송하는데 필요한 통신량은 $O\left(((t+1)(m+E)-1)\frac{N^2}{t^2}\right)$이다.

3.1.4 연산 결과 복구 복잡도

마스터에서 최종 연산 결과 C를 복구하기 위해서는 $\widetilde{C}_n$의 전체 $\frac{N^2}{t^2}$개의 행렬 원소에 대해 $((t+1)(m+E)-2)$-차 다항식의 계수를 구하는 보간법(interpolation) 문제를 풀어야 한다.[14] k차 다항식의 보간법 문제의 계산 복잡도는 $O(klog^2k)$이므로 마스터에서 필요로 하는 연산 결과 복구 복잡도는 아래와 같다.

$$O\left(\frac{N^2}{t^2}((t+1)(m+E)-1)\log^2((t+1)(m+E)-1)\right) \quad (31)$$

3.2 마스터의 대기 시간 분포

본 절에서는 마스터가 워커의 연산 결과를 기다리는 대기 시간의 분포를 실험을 통해 보이고자 한다. 마스터의 대기 시간은 마스터가 워커에게 연산을 할당하는데 걸리는 통신 시간, 워커가 할당된 연산을 계산하는데 걸리는 계산 시간, 워커가 마스터에게 연산 결과를 전달하는데 걸리는 통신 시간의 합으로 정의한다.

우선, 워커 W_n이 할당된 연산을 계산하는데 걸리는 시간을 T_n이라고 할 때, T_n은 다음과 같은 분포를 따르는 것이 밝혀져 있다.[15]

$$\Pr(T_n<t)=1-e^{-\frac{\mu}{L_t}(t-\alpha L_t)},\ \forall t\geq\alpha L_t \quad (32)$$

이 때, μ는 워커의 낙오 정도를 나타내는 낙오 상수이고 α는 워커의 계산 능력을 나타내는 계산 상수, L_t는 워커에 할당된 계산량에 관한 상수로 앞서 워커의 계산량을 밝힌 바와 같이 임의의 상수 γ를 도입하여 $L_t=\gamma\frac{N^3}{mt}$와 같이 표현할 수 있고, 행렬을 나누는 변수 t에 따라 결정된다. 워커의 계산 시간이 위와 같은 분포를 따를 때, 그 중 가장 빠른 K개의 워커가 연산을 끝내는데 걸리는 시간을 $T_{(K)}$라 하면, $T_{(K)}$의

평균 $E[T_{(K)}]$는 다음과 같이 나타낼 수 있다.

$$E[T_{(K)}] = \alpha L_t + \frac{H_P - H_{P-K}}{\mu / L_t} \simeq \alpha L_t + \frac{L_t}{\mu} log\left(\frac{P}{P-K}\right) \quad (33)$$

이 때, $H_P = \sum_{i=1}^{P} \frac{1}{i}$이고 $K = (t+1)(m+E) - 1$이다.

만약 마스터가 모든 P개의 워커와 동시에 통신이 가능하다고 가정하면 마스터의 대기 시간 T_t은 아래와 같이 표현된다.

$$T_t = T_d + T_{(K)} + T_u \quad (34)$$

이 때, T_d는 마스터가 워커에 부행렬 $\widetilde{A}_n$, $\widetilde{B}_n$을 전송하는 통신 시간으로 임의의 상수 β_d를 도입하여 $T_d = \beta_d \frac{2N^2}{m}$으로 나타낼 수 있다. 또한, T_u는 워커가 마스터에 연산 결과 $\widetilde{C}_n$을 전송하기 위한 통신 시간을 나타내며, 마찬가지로 임의의 상수 β_u를 도입하여 $T_u = \beta_u \frac{N^2}{t^2}$으로 나타낼 수 있다. 그 결과, 마스터의 평균 대기 시간 $E[T_t]$은 다음과 같이 계산된다.

$$E[T_t] = \frac{2\beta_d N^2}{m} + \alpha L_t + \frac{L_t}{\mu} log\left(\frac{P}{P-K}\right) + \frac{\beta_u N^2}{t^2} \quad (35)$$

그림 2와 그림 3은 마스터가 모든 P개의 워커와 동시에 통신이 가능하고 $P=25$, $N=2^9$, $m=4$, $E=1$일 때, 변수 t에 따른 마스터의 대기 시간의 누적 분포 함수를 보여준다. 마스터가 모든 P개의 워커와 동시에 통신이 가능한 서비스 환경은 마스터가 여러 워커와 주파수 분할 다중화(Frequency Division Multiplexing)를 통한 무선 통신을 수행하는 상황을 고려할 수 있다. 그림 2는 상대적으로 통신 시간의 비중이 작은 경우 ($\beta_d = \beta_u = 4 \times 10^{-5}$)를 나타낸다. 이 때, 변수 t를 작게 하여 복구 한계치 K가 작을 때, 낙오 효과가 감소하여 다른 t 값을 가질 때에 비해 마스터의 대기 시간이 줄어드는 것을 확인할 수 있다. 반면 그림 3은 상대적으로 통신 시간의 비중이 큰 경우 ($\beta_d = \beta_u = 3 \times 10^{-2}$)를 나타낸다. 이 경우에는 복구 한계치 K가 크더라도 변수 t를 크게 하여 마스터와 워커간 통신량을 줄이는 것이 마스터의 대기 시간을 감소시킬 수 있는 방안이라는 것을 알 수 있다.

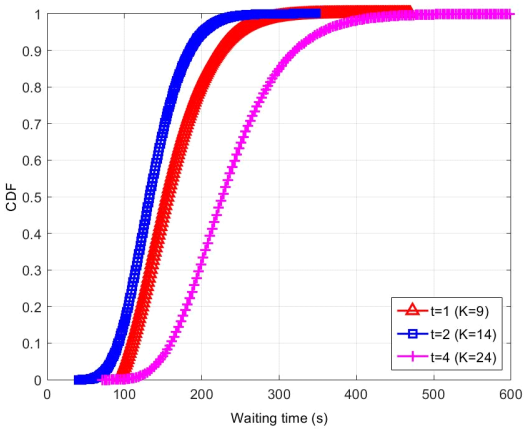

그림 2. 마스터가 모든 P개의 워커와 동시에 통신이 가능할 때 마스터의 대기 시간 분포 ($\beta_d = \beta_u = 4 \times 10^{-5}$)
Fig. 2. Waiting time at the master if the master communicates with P workers at the same time ($\beta_d = \beta_u = 4 \times 10^{-5}$)

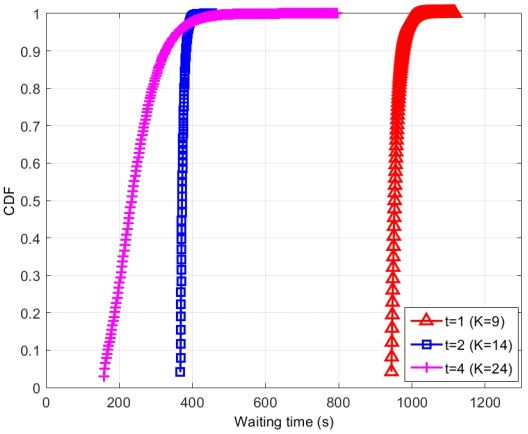

그림 3. 마스터가 모든 P개의 워커와 동시에 통신이 가능할 때 마스터의 대기 시간 분포 ($\beta_d = \beta_u = 3 \times 10^{-2}$)
Fig. 3. Waiting time at the master if the master communicates with P workers at the same time ($\beta_d = \beta_u = 3 \times 10^{-2}$)

한편, 마스터의 대기 시간에 한계가 존재하는 경우 통신량과 복구 한계치 사이의 trade-off에 따른 성능 변화를 추가로 고려할 수 있다. 그림 3에 따르면 통신량을 줄이고 복구 한계치를 크게 하는 변수 t를 설정

하는 것이 마스터의 평균적인 대기 시간을 감소시킬 수 있다는 것을 알 수 있다. 하지만 마스터의 대기 시간의 한계를 600(s)으로 설정하여 마스터의 대기 시간 한계를 만족할 수 있는 확률을 고려할 경우, $t=2$일 때 가장 좋은 성능을 보인다. 이는, 복구 한계치를 크게 하는 방법이 마스터의 평균적인 대기 시간을 감소시킬 수 있지만 낙오 효과에 따른 영향이 증가하여 마스터의 대기 시간 한계를 만족하지 못하는 확률 또한 증가하기 때문이다. 따라서, 분산 컴퓨팅 환경에서 고려하는 성능 지표(마스터의 평균적인 대기 시간 또는 마스터의 대기 시간 한계를 만족할 수 있는 확률)에 따라 적절한 변수를 설정하는 것이 중요하다.

추가적으로, 행렬 곱 연산 서비스를 필요로 하는 행렬 입력이 마스터에 도달하는 상황을 고려하여, 트래픽에 따른 연산 처리 성능을 확인하였다. 그림 4는 마스터가 모든 P개의 워커와 동시에 통신이 가능하고 $P=25$, $N=2^9$, $m=4$, $E=1$, $\beta_d=\beta_u=4\times10^{-5}$인 분산 컴퓨팅 환경에서 마스터에서 대기 중인 행렬 곱 연산의 개수 변화를 나타낸 것이다. 행렬 곱 연산을 필요로 하는 입력의 도달 시간이 다음과 같은 푸아송 분포(Poisson's distribution)를 따른다고 가정하였다.

$$f(x;\lambda)=\frac{\lambda^x e^{-\lambda}}{x!} \tag{36}$$

그림 4는 $\lambda=170$인 경우를 나타낸다. 이에 따르면 그림 2에서 나타난 바와 같이 $t=2$일 때 마스터의 평균 대기시간이 짧으므로 $t=1$ 또는 $t=4$일 때에 비해 마스터에서 대기 중인 행렬 곱 연산의 수가 적어 좋은 성능을 나타내는 것을 확인할 수 있다. 또한, $t=1$인 경우 다른 값에 비해 낙오 효과가 작아 마스터에서 대기 중인 행렬 곱 연산의 수는 $t=2$에 비해 손해가 크지 않음을 알 수 있다.

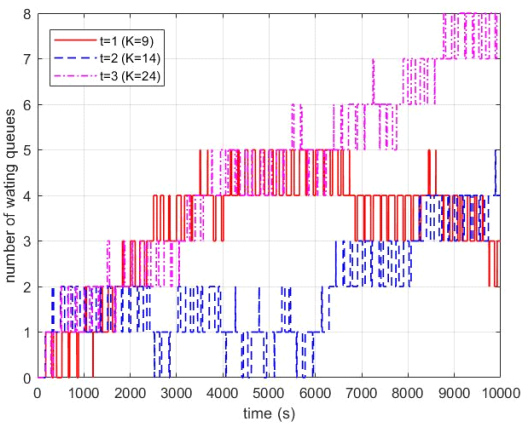

그림 4. 마스터에서 대기 중인 행렬 곱 연산 개수 변화
Fig. 4. The number of matrix multiplication tasks waited at the queue of the master

또 다른 서비스 환경으로 마스터가 여러 워커와 시간 분할 다중화(Time Division Multiplexing)를 통한 무선 통신을 수행하는 상황을 고려할 수 있다. 이 경우, 마스터가 한 타임 슬롯마다 하나의 워커와 통신이 가능하다고 가정하면 각각의 워커가 부행렬 $\widetilde{A}_n$, $\widetilde{B}_n$를 전송받아 연산을 시작하는 시점이 서로 다르므로, 마스터의 대기 시간 T을 정확히 결정하기 힘들다. 대신, 마스터의 대기 시간 T은 아래와 같은 범위를 가진다고 할 수 있다.

$$T_d+T_{(K)}+T_u \le T_t \le PT_d+T_{(K)}+KT_u \tag{36}$$

그림 5와 그림 6은 마스터가 한 타임 슬롯마다 하나의 워커와 통신이 가능하고 $P=25$, $N=2^9$, $m=4$, $E=1$일 때, 변수 t에 따른 마스터의 대기 시간의 누적 분포 함수를 보여준다. 그림 5는 상대적으로 통신 시간의 비중이 작은 경우($\beta_d=\beta_u=4\times10^{-5}$)를 나타낸다. 앞서 그림 2에서는 변수 t를 작게 하여 복구 한계치 K가 작을 때 마스터의 대기 시간이 가장 줄어드는 것을 확인할 수 있었지만, 그림 5의 경우 마스터의 대기 시간 중 통신 시간

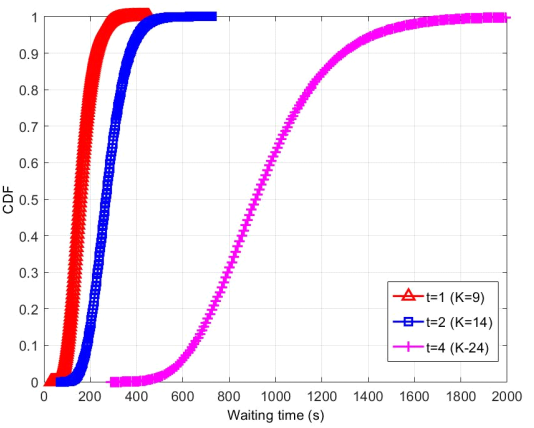

그림 5. 마스터가 한 타임 슬롯마다 하나의 워커와 통신이 가능할 때 마스터의 대기 시간 분포 ($\beta_d=\beta_u=4\times10^{-5}$)
Fig. 5. Waiting time at the master if the master communicates with each worker at a time slot ($\beta_d=\beta_u=4\times10^{-5}$)

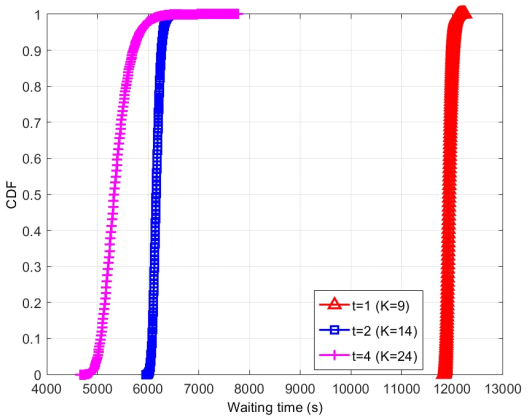

그림 6. 마스터가 한 타임 슬롯마다 하나의 워커와 통신이 가능할 때 마스터의 대기 시간 분포 ($\beta_d = \beta_u = 3\times10^{-2}$)
Fig. 6. Waiting time at the master if the master communicates with each worker at a time slot ($\beta_d = \beta_u = 3\times10^{-2}$)

이 차지하는 비중을 정확히 예측하기 힘들기 때문에 같은 경향성을 가지지 않는다는 것을 알 수 있다. 그림 6은 상대적으로 통신 시간의 비중이 큰 경우 ($\beta_d = \beta_u = 3\times10^{-2}$)를 나타낸다. 이 경우, 마스터가 어떠한 워커와 먼저 통신할 것인지 정해지지 않았지만 기본적으로 통신 시간이 차지하는 비중이 크기 때문에 그림 3에서와 마찬가지로 변수 t를 크게 하여 마스터와 워커간 통신량을 줄이는 것이 마스터의 대기 시간을 감소시킬 수 있음을 확인할 수 있다.

마지막으로, 행렬 곱 연산을 수행하는 분산 컴퓨팅에서 통신량과 복구 한계치 사이의 trade-off를 그래프로 표시하였다. 그림 7은 $N=100, m=10, P=120$인 분산 컴퓨팅 환경에서 통신량과 복구 한계치 사이의 trade-off를 나타낸다. 만약 통신량을 줄이고자 할 경우 복구 한계치가 늘어나 분산 컴퓨팅에서의 낙오 효과에 의한 영향을 크게 받게 되고, 복구 한계치를 줄이고자 할 경우 통신량이 늘어나 워커가 마스터에게 연산 결과를 전달하는 통신 시간이 늘어난다. 또한, 데이터 보안을 고려하지 않는 경우($E=0$)와 비교해 개별 워커로부터 데이터 보안을 보장하는 경우 ($E=1$) 필요로 하는 복구 한계치의 증가도 확인할 수 있다.

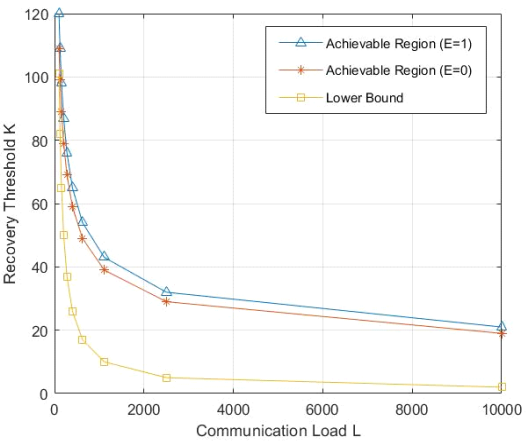

그림 7. 통신량과 복구 한계치 사이의 trade-off ($N=100, m=10, P=120$)
Fig. 7. Trade-off between communication load and recovery threshold ($N=100, m=10, P=120$)

Ⅳ. 결 론

본 논문에서는 데이터 보안을 보장하는 행렬 곱 연산을 수행하는 분산 컴퓨팅 시스템에서 마스터와 워커간 통신량과 마스터가 최종 연산 결과를 복구하기 위한 복구 한계치 사이의 이론적 trade-off 관계를 다뤘다. 먼저, 데이터 보안을 보장하는 분산 행렬 곱 기법을 제안하였으며, 제안한 기법으로 달성 가능한 통신량과 복구 한계치의 영역을 보였다. 또한, 정해진 통신량 하에서 복구 한계치의 정보 이론적 한계를 수식을 통해 밝혔다. 이에 따라 통신량과 복구 한계치가 서로 상충 관계를 가져 분산 컴퓨팅 환경에 따라 적절한 통신량과 복구 한계치를 설정함으로써 분산 컴퓨팅 성능을 향상시킬 수 있다는 것을 확인하였다.

마지막으로, 실험을 통해 통신량과 복구 한계치 사이의 trade-off가 마스터의 대기 시간에 어떤 영향을 끼치는지 확인하였고, 적절한 변수를 설정함으로써 마스터의 대기 시간을 감소시킬 수 있음을 보였다.

본 논문에서 다룬 데이터 보안을 보장하는 분산 컴퓨팅 시스템은 데이터의 크기가 큰 연산을 여러 개의 워커를 통해 나누어 처리하는 환경에서 연산 시간을 효과적으로 감소할 수 있으며, 이에 따라 향후 머신러닝 또는 빅데이터 연산을 처리하는 서비스를 제공하는 미래의 통신 네트워크에 활용될 수 있을 것이다.

References

[1] J. Dean and L. Barroso, "The tail at scale," *Commun. 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 Trans. Info. Theory*, vol. 64, no. 3, pp. 1514-1529, Mar. 2018.

[3] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Polynomial codes: an optimal design for high-dimensional coded matrix multiplication," in *Proc. 32th Annual Conf. NIPS*, pp. 4403-4413, Dec. 2017.

[4] S. Dutta, M. Fahim, F. Haddadpour, H. Jeong, V. Cadambe, and P. Grover, "On the optimal recovery threshold of coded matrix multiplication," *IEEE Trans. Info. Theory*, vol. 66, no. 1, pp. 278-301, Jan. 2020.

[5] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Straggler mitigation in distributed matrix multiplication: fundamental limits and optimal coding," *IEEE Trans. Info. Theory*, vol. 66, no. 3, pp. 1920-1933, Mar. 2020.

[6] H. Yang and J. Lee, "Secure distributed computing with straggling servers using polynomial codes," *IEEE Trans. Info. Forensics and Secur.*, vol. 14, no. 1, pp. 141-150, Jan. 2019.

[7] Q. Yu, S. Li, N. Raviv, S. M. M. Kalan, M. Soltanolkotabi, and A. S. Avestimehr, "Lagrange coded computing: optimal design for resiliency, security, and privacy," in *Proc. 22nd Int. Conf. AISTATS*, Okinawa, Japan, Apr. 2019.

[8] Q. Yu and A. S. Avestimehr, "Entangled polynomial codes for secure, private, and batch distributed matrix multiplication: breaking the "cubic" barrier," arXiv preprint arXiv:2001.05101, Jan. 2020.

[9] R. G. L. D'Oliveira, S. E. Rouayheb, and D. Karpuk, "GASP codes for secure distributed matrix multiplication," *IEEE Trans. Info. Theory*, vol. 66, no. 7, pp. 4038-4050, Jul. 2020.

[10] M. Aliasgari, O. Simeone, and J. Kliewer, "Private and secure distributed matrix multiplication with flexible communication load," *IEEE Trans. Info. Forensics and Secur.*, vol. 15, no. 2, pp. 2722-2734, Feb. 2020.

[11] H. Yang, "A tradeoff between communication load and straggling effect in distributed computing with data security," in *Proc. Symp. KICS*, pp. 90-91, Feb. 2020.

[12] H. Yang, "Data processing time in distributed computing systems with data security," in *Proc. Symp. KICS*, pp. 92-93, Feb. 2020.

[13] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd Ed., NJ, Wiley, 2006.

[14] K. S. Kedlaya and C. Umans, "Fast polynomial factorization and modular composition," *SIAM J. Computing*, vol. 40, no. 6, pp. 1762-1802, 2011.

[15] G. Liang and U. C. Kozat, "TOFEC: achieving optimal throughput-delay trade-off of cloud storage using erasure codes," in *Proc. IEEE Conf. Comput. Commun.(INFOCOM)*, Toronto, Canada, Apr. 2014.

양 희 철 (Heecheol Yang)

2013년 2월 : 서울대학교 전기·정보공학부 졸업

2018년 2월 : 서울대학교 전기·컴퓨터공학부 박사

2019년 9월~현재 : 금오공과대학교 전자공학과 조교수

<관심분야> 무선통신, 머신러닝, 이동통신

[ORCID:0000-0002-2802-2143]