7J-01

# AMD EPYC 上での敵対的生成ネットワークの OpenMP による並列処理

Parallelization of Generative Adversarial Networks by OpenMP on AMD EPYC

中村 優志 †　井出 俊輔 ††　吉田 明正 ††
Yuji Nakamura　Shunsuke Ide　Akimasa Yoshida

## 1　はじめに

近年，深層学習で注目されている生成モデルの一つとして敵対的生成ネットワーク（GAN: Generative Adversarial Networks）がある．GAN は生成器と識別器の二つのニューラルネットワークが競い合うように学習することで，本物データと見分けがつかない偽物データの生成が可能である．深層学習の問題点の一つとして，膨大な学習時間があげられる．特に，GAN により精度の高い偽物データを生成するには多くの時間を要する．このような問題点を解決するために，マルチコアによる並列処理が有効 [1][2][3] と考えられる．

本稿では，GAN プログラムを C 言語実装し，OpenMP[4] の指示文を加えて並列プログラムを作成した．AMD EPYC 7443P 上での性能評価の結果，高い実効性能が得られ，提案手法の有効性が確認された．

## 2　敵対的生成ネットワーク (GAN)

敵対的生成ネットワーク（GAN: Generative Adversarial Networks）とは，生成器（Generator）と識別器（Discriminator），二つのニューラルネットワークが競い合うように学習する生成モデルの一種である [5]．図 1 に示すように，生成器は偽物を作り，識別器を騙すことが目的である．ランダムなノイズから偽物を作成し，偽物が識別器を騙せるように学習が進む．識別器は偽物を見破る側であり，生成器が作成した偽物を見破ることが目的である．本物と生成器が作成した偽物の両者を教師データとし，偽物を見破れるように学習する．

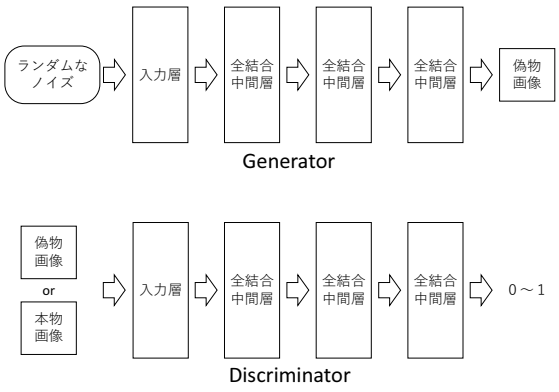


図 1　GAN の構成.

### 2.1　GAN の OpenMP によるループ並列処理

本研究で C 言語実装した GAN プログラムにおいて dot() 関数は処理時間の 48 パーセントを占めている．この行列計算を高速化することで GAN プログラム全体の高速化が可能である．dot() 関数は三重ループで構成されており，後述する#pragma omp parallel for 指示文を最外側のループに記述する．また，内側ループのインデクス変数および計算結果を格納する一時変数を private 指示節でプライベート化することで，イタレーション間のデータ依存を防いでいる．

### 2.2　GAN の OpenMP による粗粒度並列処理

本研究で実装した GAN プログラムは，大きく分けて三つの処理に分割することができる．一つ目はノイズから偽物を生成し識別器の学習，二つ目は本物を使った識別器の学習，三つめは統合したモデルにより生成器の学習となる．この三つの処理を粗粒度並列処理することで GAN プログラムの高速化が可能である．しかし，この三つの処理は一つのニューラルネットワークを用いて学習しているのでこのまま粗粒度並列処理を実行するとデータ依存が発生する．本稿ではこの問題を解決するために，三つの処理それぞれにニューラルネットワークを用意し，学習一回ごとに各ニューラルネットワークの重みとバイアスの平均値をとり共有するプログラムを作成した．粗粒度並列化したプログラムを図 2 に示す．

```
1  int main(int argc, char *argv[]) {
2      …
3      for (i=0;i<n_learn;i++) {
4          #pragma omp parallel sections
5          {
6              #pragma omp section
7              ノイズから偽物を生成し識別器の学習
8              #pragma omp section
9              本物を使った識別器の学習
10             #pragma omp section
11             統合したモデルにより生成器の学習
12         }
13         各ニューラルネットワークの値の共有
14     }
15     …
16 }
```

図 2　GAN プログラムの粗粒度並列処理.

## 3　OpenMP による並列処理

本章では，並列プログラム作成に用いた OpenMP[4] について述べる．OpenMP とは，共有メモリ型 PC での並列プログラミングを可能にする API であり，C/C++ および Fortran で使用可能である．OpenMP の特徴は指示文を挿入するだけで並列化を可能にするため，新たな開発環境のインストールの必要がなく，デバックが容

†明治大学大学院先端数理科学研究科
Graduate School of Advanced Mathematical Sciences, Meiji University
††明治大学総合数理学部
School of Interdisciplinary Mathematical Sciences, Meiji University

易である点である．

OpenMP はプログラムに OpenMP 指示文を加えることで並列化を行っている．OpenMP 指示文は指示文と指示節から構成されており，構造によって役割が異なり，実行される処理も異なる．基本的な OpenMP 指示文は #pragma omp parallel であり，中括弧で囲まれた範囲を並列化させる．

### 3.1 OpenMP によるループ並列処理

OpenMP でループ並列処理をする際は，#pragma omp parallel for 指示文を用いる．この指示文の直後の for ループを各スレッドに分割して割り当てて並列処理を行う．二重以上の for ループの最外側を並列化した場合は，内側のループは外側のループの反復によって割り当てられたスレッドで処理される．

OpenMP には様々な指示節がある．ここでは本稿で用いた private 指示節について述べる．使用方法は指示文の後に private() を記述する．括弧内に指定された変数が各スレッドでローカル化される．

### 3.2 OpenMP による粗粒度並列処理

OpenMP で粗粒度並列処理を行う際は，#pragma omp parallel sections を用いる．この指示文は#pragma omp section とともに用いる．#pragma omp section の次の一文またはブロックを一つのスレッドに割り当てて並列処理を行う．なお本稿では，ループ並列処理の時と同様に粗粒度並列処理の際も private 指示節を用いている．

## 4 GAN の OpenMP による並列処理の性能評価

本性能評価では，まずループ並列処理の性能評価を行う．次に main() 関数での粗粒度並列処理の性能評価を行う．本稿で実装した GAN プログラム [5] は MNIST の手書き数字を教師データとし，1 万回の学習を行っている．

### 4.1 性能評価環境

本性能評価では，表 1 に示す並列システムを使用する．

表 1 性能評価に用いる並列システム．

| | |
|---|---|
| CPU | AMD EPYC 7443P 24 コア |
| メモリ | 128GB |
| OS | Ubuntu 20.04LTS |
| 処理系 | gcc 8.4.0 |

### 4.2 ループ並列処理の性能評価

本節では，GAN プログラムの行列計算部分に対してループ並列処理を適用し性能評価を行う．1 コア，2 コア，4 コア，6 コア，12 コアでそれぞれ実行し実行時間を比較する．

実行結果は図 3 のようになっており，1 コア実行を基準に，2 コアで 1.66 倍，4 コアで 2.52 倍，6 コアで 2.96 倍，12 コアで 3.71 倍の速度向上が達成された．

### 4.3 粗粒度並列処理の性能評価

本節では，main() 関数に粗粒度並列処理を適用し性能評価を行う．1 コア，2 コア，3 コアでそれぞれ実行し実行時間を比較する．

実行結果は図 4 のようになっており．1 コア実行を基準に，2 コアで 1.25 倍，3 コアで 1.51 倍の速度向上が達成された．

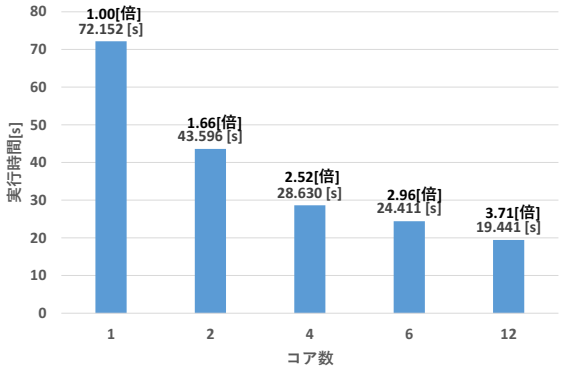


図 3 ループ並列処理の性能評価．

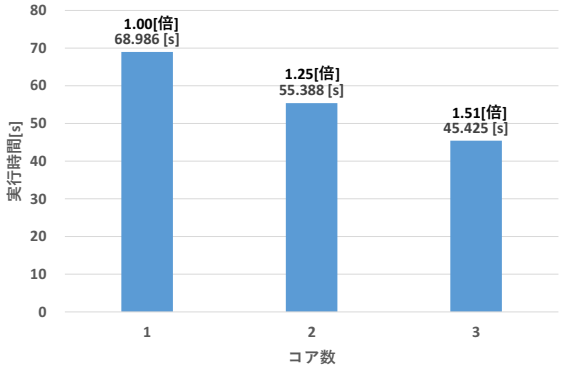


図 4 粗粒度並列処理の性能評価．

## 5 おわりに

本稿では，AMD EPYC 7443P 上での敵対的生成ネットワークの OpenMP による並列処理手法を提案した．AMD EPYC 7443P 上での性能評価の結果，ループ並列処理では逐次実行に比べて最大で 3.71 倍の高速化を達成した．粗粒度並列処理では逐次実行に比べて最大で 1.52 倍の高速化を達成した．以上の結果から提案手法の有効性が確認された．今後の課題としては，ループ並列処理と粗粒度並列処理による階層的並列処理があげられる．

## 参考文献

[1] Ryuichi Yamamoto, Eunwoo Song, Jae-Min Kim. Parallel Wavegan: A Fast Waveform Generation Model Based on Generative Adversarial Networks with Multi-Resolution Spectrogram, ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)

[2] Emiliano Perez, Sergio Nesmachnow, Jamal Toutouh, Erik Hemberg, Una-May O 'Reily. Parallel/distributed implementation of cellular training for generative adversarial neural networks, 2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW).

[3] Corentin Hardy, Erwan Le Merrer, Bruno Sericola. MD-GAN: Multi-Discriminator Generative Adversarial Networks for Distributed Datasets, 2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS).

[4] 菅原清文．C/C++プログラマーのための OpenMP 並列プログラミング，カットシステム, 2009.

[5] 我妻幸長．はじめてのディープラーニング 2，SB Creative, 2020.