計測自動制御学会東北支部 第 355 回研究集会 (2025.11.8)
資料番号 355-5

時空間並列性を活用したVision TransformerのFPGA実装

An FPGA Implementation of Vision Transformer Exploiting Spatial and Temporal Parallelism

○松井友宏，ウィッデヤスーリヤハシタムトゥマラ，田中大介，張山昌論

Tomohiro Matsui, Hasitha Muthumala Waidyasooriya,
Daisuke Tanaka，Masanori Hariyama

東北大学

Tohoku University

キーワード： Vision Transformer (Vision Transformer)，再構成可能コンピューティング (reconfigurable computing)，FPGA (Field Programmable Gate Array)，機械学習 (machine learning)

連絡先： 〒 980-8579　仙台市青葉区荒巻字青葉 6-3-09 東北大学大学院 情報科学研究科
張山・ウィッデヤスーリヤ研究室
松井友宏，Tel.: 022-795-7155 E-mail: matsui.tomohiro.t6@dc.tohoku.ac.jp

1. はじめに

Transformer アーキテクチャ[1] は、機械学習分野における画期的な進歩を示すものであり、その柔軟性と高い拡張性により、自然言語処理からコンピュータビジョンに至るまで、幅広い応用に活用されている。Vision Transformer（ViT）[2] は、その中でも特に強力な画像解析モデルとして登場した。ViT は自己注意（self-attention）メカニズムを活用することで、視覚データ中の長距離依存関係（long-range dependencies）を効果的に捉え、畳み込みニューラルネットワーク（CNN）に代わる有力な手法として注目を集めている。Vision Transformer の計算で特徴的なのは、極めて高いデータ並列性（data parallelism）である。大量のデータが外部メモリから繰り返し読み込まれ、大規模な並列演算によって処理され、その後再びメモリに書き戻される。この一連の処理が多数の層で繰り返されるため、頻繁かつ膨大なメモリアクセスが発生する。GPU は高い外部メモリ帯域と多数の演算コアを備えており、このような処理に適しているが、一方で極端に多い外部メモリアクセスによって消費電力が大幅に増加するという問題がある。

エネルギー効率を改善する有望な方法の一つは、外部メモリアクセスの削減である。しかし、単にメモリアクセス回数を制限すると、並列計算に必要なデータ供給が不足し、演算性能のボトルネックを生じる可能性がある。したがって、計算の並列性を損なうことなく、消費電力を抑えるためには、より洗練されたアーキテクチャ的な工夫が必要である。この課題に対処するため、著者らは先行研究[3] において、Multi-head attention 機構における「時間的並列性（temporal parallelism）」を活用するアーキテクチャを提案した。本論文では、そのアプローチを Vi-

sion Transformer 全体に拡張し、時間的並列性と空間的並列性（spatial parallelism）の両方を探究する。

本研究では、Vision Transformer の計算全体を対象として、マクロブロックレベルからマイクロ演算単位に至るまで、複数の粒度で空間的・時間的並列性を分析する。特に、中間結果を外部メモリに書き戻すことなく再利用することで、データ再利用効率を最大化する技術に焦点を当てている。時間的並列性は主に、外部メモリアクセスを削減しつつ十分な並列処理量を維持するために用いられる。一方で、空間的並列性は、データフローの最適化やパイプラインのボトルネック段階の処理時間短縮に活用される。このように、メモリアクセスを積極的に削減しつつ深くパイプライン化されたアーキテクチャを実装するには、Field-Programmable Gate Array（FPGA）のような柔軟なハードウェアプラットフォームが最適である。FPGA はカスタム化された省電力アクセラレータ設計を可能にする。本研究では、提案アーキテクチャを Intel Agilex 7 FPGA 上に実装した。実験結果として、本設計は高性能に最適化された 20 コア CPU 実装を上回る性能を示した。

2. 空間的および時間的並列性

2.1 空間的および時間的並列性

Fig. 1 は、2 種類の基本的な並列処理手法、すなわち空間的並列性と時間的並列性を示している。Fig. 1(a) に示すように、空間的並列性では、複数の演算要素（Processing Elements: PEs）を用いて、大量のデータを同時並行的に処理する。それぞれの PE が異なるデータに対して並行して計算を行うことで、スループット（処理速度）を向上させることができる。多くの場合、同じハードウェアユニットを異なる種類の計算に時間的に再利用することもある。空間的並列性を効果的に活用するためには、システム

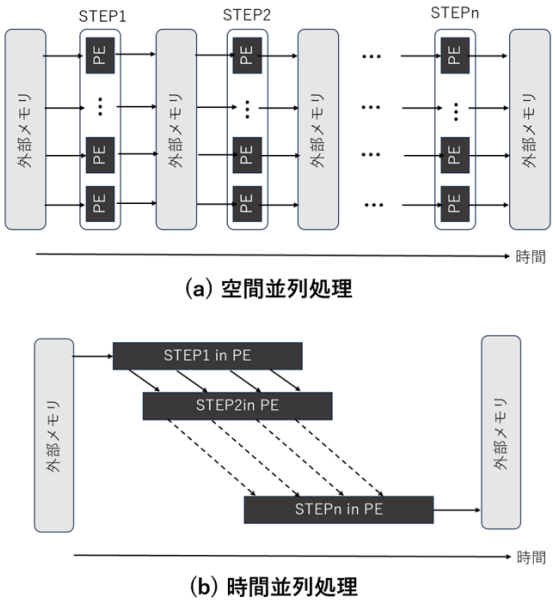

Fig. 1　空間的並列処理と時間的並列処理

が外部メモリへの並列アクセスを行う必要がある。しかし、これにより高いメモリアクセス帯域幅が要求される。同様に、中間結果も同時に外部メモリへ書き戻す必要がある。これは、内部メモリ資源が通常、全ての処理データを保持するには不十分であるためである。十分なメモリ帯域が確保できる場合、空間的並列性によって高い処理速度を実現することが可能である。しかし、外部メモリへの頻繁かつ大規模なアクセスは、非常に高い消費電力を引き起こす原因となる。特に、GPU を用いた処理ではこの問題が顕著である。

一方、時間的並列処理は、ある程度の並列性を維持しながらも、外部メモリアクセスを大幅に削減することに重点を置いている。Fig. 1(b) に示すように、異なる処理をパイプライン状に順番にスケジューリングし、各段階で得られた中間結果を外部メモリに書き戻さずに、次の段階で直接再利用する。これにより、演算処理が連続的かつ効率的に行われる。各ステップでは、外部メモリからアクセスまたはストリームされるデータ量はごく一部に限られ、そのデータが複数の PE 間で再利用される。最終的な結果のみが、全ての計算パイプラインを通過した後で

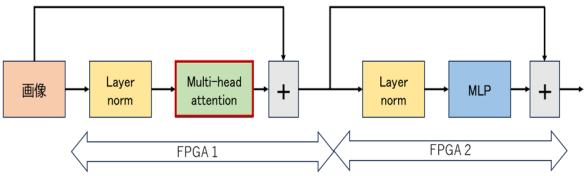

Fig. 2　Vision Transformer Encoder の処理フロー

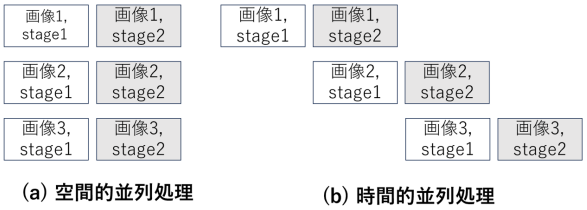

Fig. 3　マクロレベルの Vision Transformer 並列処理

外部メモリに書き戻される。このアプローチにより、外部メモリアクセスを劇的に削減でき、結果として消費電力を大幅に低減できる。しかし、こうした中間データの再利用を可能にするための演算スケジューリングを正確に設計することは非常に難しい。特に、CPU や GPU のような汎用プロセッサでは、スレッド間で頻繁な同期が必要となり、その際に大きなオーバーヘッドが発生する。このオーバーヘッドが、せっかく削減したメモリアクセスによる性能向上効果を相殺してしまうことがある。それに対して、FPGA のようなハードウェアプラットフォームは、時間的並列性の活用に非常に適している。FPGA では、演算処理のスケジューリングを細かく制御でき、同期のための複雑な制御を必要とせずに中間データをシームレスに再利用することが可能である。この結果、外部メモリアクセスの大幅な削減が実現し、電力消費を大きく低減できる。

2.2　Vision Transformer における並列性の分析

ここでは、数百枚程度の画像からなるバッチ入力を処理するケースを考える。Fig. 2 は、Vision Transformer のマクロレベルでの計算フローを示しており、大きく 2 つのステップに分けることができる。第 1 ステップ：レイヤ正規化の後に Multi-Head Attention を実行する。第 2 ステップ:再び レイヤ正規化を行い、その後に多層パーセプトロン（Multi-Layer Perceptron: MLP）の計算を行う。これらの 2 つのステップは、Fig. 3 に示すように、空間的並列処理または時間的並列処理のいずれかで実行できる。Fig. 3(a) に示すように、空間的並列性では、複数の埋め込み画像を同時に処理し、それぞれのデータに並行してアクセスする。この手法は、十分なメモリ帯域幅が確保できる場合には、非常に高いスループットを実現できる。一方で、Fig. 3(b) に示すように、時間的並列性では、2 つのステップを異なる埋め込み画像に対してパイプライン形式で順次処理する。この方法では、中間データを再利用することによって外部メモリアクセスを大幅に削減できるが、得られる並列性の度合いは限定的であり、大きな性能向上を実現するには不十分な可能性がある。したがって、より高い性能を達成するためには、各ステップの内部構造をさらに詳細に分析し、より細かい粒度での時間的並列処理を特定することが重要である。

2.3　Multi-head-attention の並列処理

Fig. 4 は、ステップ 1 の計算フローを示しており、このステップはレイヤ正規化、Multi-Head Attention、および加算の処理から構成されている。Multi-Head Attention は、複数の独立した「ヘッド（head）」から構成されており、それぞれのヘッドは入力データの異なる側面に焦点を当てて学習を行う。これにより、モデルは入力シーケンス内の異なる位置関係に同時に注目で

き、多様な関係性を効果的に捉えることができる。Multi-head-attention 機構では、入力シーケンスはそれぞれのヘッドに対して、異なる線形変換を用いて複数のクエリ（Q)、キー（K)、およびバリュー（V）に射影される。各ヘッドは独立して、クエリとキーを比較してアテンションスコアを計算し、そのスコアをバリューに適用してコンテキストベクトルを生成する。

すべてのヘッドから得られたコンテキストベクトルは、その後連結され、線形変換を通して処理されることで、アテンションブロック全体の最終出力が得られる。以下の式は、自己注意機構における単一ヘッドの計算処理を表している。

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \tag{1}$$

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{X}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{X}\mathbf{W}_V \tag{2}$$

ここで、X は位置エンコーディング（positional encoding）が付加された後の入力データを表す。また、W_Q, W_K および W_V は、それぞれ クエリ（Query）、キー（Key）、およびバリュー（Value） の射影に対応する重み行列である。これらの演算は、バッチ内の各入力画像に対して個別に行われ、すべてのアテンションヘッドに対して繰り返し実行される。マルチヘッド・アテンション機構は、非常に高い空間的並列性を備えている。すべてのアテンションヘッドは同時に並列処理することが可能であり、さらに各ヘッドの内部でも、バッチ内のすべての入力データを並行して処理できる。加えて、クエリ（Q)、キー（K)、バリュー（V）を計算するための行列積演算も同時に実行可能である。それぞれの行列積演算自体も細かい粒度で並列化が可能であるため、この段階は空間的並列処理による高速化に非常に適している。実際、複数のアテンションヘッドをまたいで複数の画像を同時に処理することで、非常に高い並列性を実現できる。しかしこの手法では、埋め込み画像データと重み行列の両方を外部メモリから同時にアクセスする必要があるため、メモリ帯域幅の要求が非常に大きくなるという問題がある。一方、時間的並列処理では、各ヘッドの処理を順番に開始し、全体の 5 つの計算ブロックがパイプライン的に動作するようにスケジューリングされる。この方法では、空間的並列処理と比較して達成できる並列性の度合いは低いものの、一度に 1 つのヘッド分のデータだけをメモリから読み出せばよいため、外部メモリアクセスを大幅に削減できる。

総じて言えば、時間的並列性はスループットをある程度犠牲にする代わりに、より高い電力効率を実現する。一方で、空間的並列性は高い性能を実現できるが、より大きなメモリ帯域を必要とする。最適な設計は、利用可能なメモリ帯域幅と電力制約とのトレードオフによって決まる。

2.4 MLP の並列処理

Fig. 5 は、ステップ 2 における MLP 計算の構造を示している。この計算は 5 つの計算ステージで構成されており、その中にはレイヤ正規化、行列積、および GELU 活性化関数が含まれている。Fig. 5 に示す通り、5 つのステージがパイプライン的に動作し、任意の時点でメモリから読み出す必要のあるデータ量を削減できる。この方法では、空間的並列処理と比較して並列度は低下するものの、一度に 1 枚の画像分のデータだけを扱えばよいため、外部メモリアクセスを大幅に削減できる。

3. Vision Transformer の FPGA アクセラレータ

第 II 章で説明したように、時間的並列性は、データの再利用効率を高めるための重要な要素であり、これが消費電力の削減に直結する。し

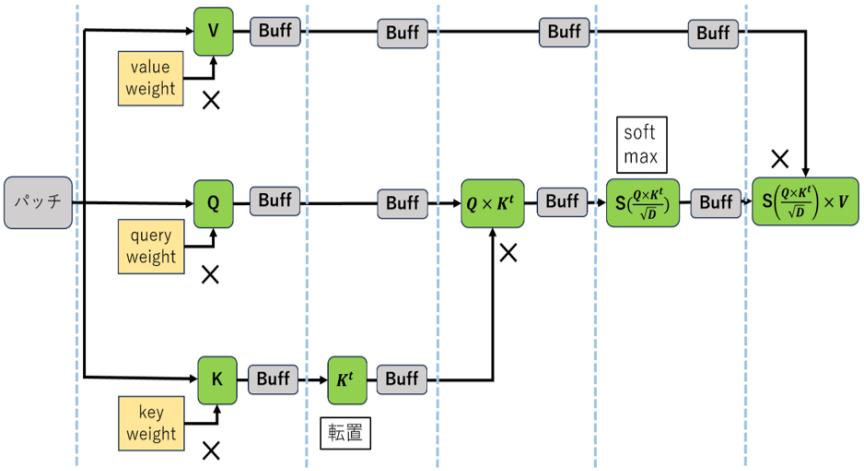

Fig. 4　Multi-head-attention の FPGA 処理の流れ

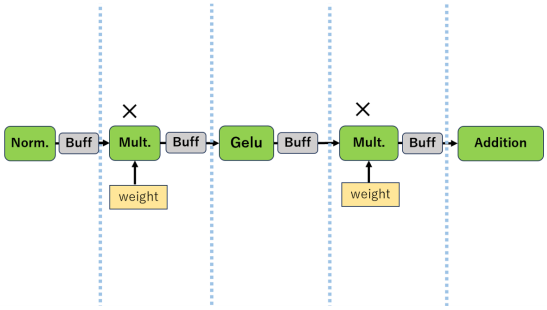

Fig. 5　MLP の FPGA 処理の流れ

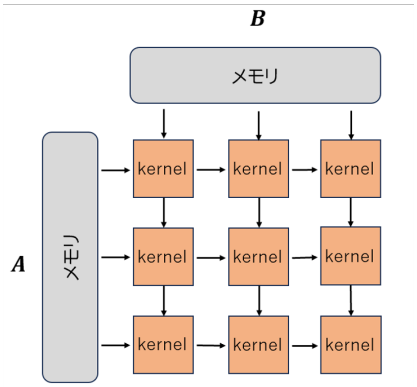

Fig. 6　MLP の FPGA 処理の流れ

たがって本研究では、可能な限り時間的並列性を優先的に利用する設計方針を採用している。一方で、空間的並列性は、マクロパイプライン内のボトルネックとなるステージの処理時間を短縮するために活用される。たとえば、マルチヘッド・アテンションにおける行列積は最も大きなボトルネックであるため、複数の行列積演算を同時に処理することで、空間的並列性を利用して処理時間を短縮している。

行列積の実装には、Fig. 6 に示すように [4] で用いられたシストリックアレイ構造を採用している。各行および列における PE の数は、処理速度や FPGA リソース制約など、設計要件に応じて調整可能である。すべての PE はパイプライン的に同時並行して動作するため、この構造では高いスループットを実現しつつ、アレイ全体でデータ再利用を最大化することができる。このアプローチは、計算効率を高めるだけでなく、外部メモリアクセスを削減して電力消費を抑えるという効果もある。同様の手法で、レイヤ正規化、行列転置、および GELU 活性化といった他のコンポーネントも、複数のマイクロステージに分割して実装している。このような分割設計により、中間データの効率的な再利用が可能となり、メモリアクセスの最小化とともに、性能および電力効率の向上が実現されている。

このように、時間的並列性と空間的並列性を組み合わせたハイブリッド並列戦略により、低消費電力を維持しながらさらなる高速化を実現できる。このような大規模なマルチ FPGA アクセラレータシステムの設計と評価は、今後の課題として残されている。しかしながら、本アーキテクチャは、高スループットかつ高エネルギー効率な Vision Transformer 実装に向けた明確な

性能スケーリングの道筋を示している。

3.1 評価

評価のために BittWare IA-840F FPGA ボードを使用した。このボードには Intel Agilex-7 AGF027 FPGA が搭載されている。FPGA カーネルのコンパイルには Intel OneAPI バージョン 2024.1 と Quartus Prime Pro 23.1 を使用した。比較対象として用いた CPU は、Intel Xeon Silver 4316（20 コア）である。CPU 上の行列演算を高速化するために Intel MKL（Math Kernel Library） を使用した。Table 1 にエンコーダのパラメータを示す。すべての計算は 単精度浮動小数点（FP32） で実行した。

Table 1　Vision Transformer 主要パラメータ

Image size	32
Number of channels	3
Patch size	8
MLP size	3072
Hidden size	768
Number of heads	12
Batch size	200

Table 2　FPGA Resource Utilization Summary

Resource	**Utilization**	**Percentage**
Registers	1,747,339	–
Logic	715,410	78%
DSP	3,563	42%
RAM blocks	3,465	26%
Memory	7.26 MB	22%
Clock frequency	292 MHz	–

マルチヘッドアテンション計算における FPGA リソース使用率を Table 2 にまとめる。主要な制約要因はロジックブロックの使用量であり、一方で浮動小数点演算に用いる DSP（Digital Signal Processor）の使用率は 42% に留まった。

また提案手法は、CPU 実装や従来 FPGA 実装[3] と比較して高速の処理を実現した。この性能向上は主に、本アーキテクチャで採用した高い並列処理度によるものである。現時点では、主なボトルネックは行列積にある。FPGA リソースの活用をさらに最適化することで、演算ユニット（PE）の数を増やし、さらなる性能向上が可能と考えられる。しかし、高い並列度を持つ完全な Vision Transformer 全体を単一 FPGA 上に実装するのは、現状ではリソース容量を超えている。そのため今後は、マルチ FPGA 構成による拡張を進め、並列性をより一層高めるとともに、Vision Transformer 全体のエンドツーエンド加速を実現する予定である。

4. まとめ

本研究では、Vision Transformer における時間的並列性と空間的並列性の両方を分析し、外部メモリアクセスと電力消費を削減するために時間的並列性を優先した深いパイプライン構造を提案した。提案アーキテクチャに基づき、マルチヘッドアテンション計算を FPGA 上に実装し、その性能を評価した。実験結果から、提案アクセラレータは並列マルチコア CPU 実装よりも高い性能を示し、さらなる高速化の可能性を有することが確認された。また、同一ハードウェアプラットフォーム上での従来 FPGA 実装[3] よりも優れた性能を達成している。提案したアーキテクチャは本質的にスケーラブルであり、計算パイプラインを分割することで、複数の FPGA に効率的に分散実行できる構造を備えている。今後の研究では、この手法を Vision Transformer 全体に拡張し、マルチ FPGA 環境で実装することを目指す。これにより、大規模トランスフォーマモデルに対して、高スループットかつ省電力なアクセラレーションを実現することを目的とする。

参考文献

1) Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.

2) Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929, 2020.

3) Hasitha Muthumala Waidyasooriya, Masanori Hariyama, and Daisuke Tanaka. FPGA-Based Deep-Pipelined Architecture for Vision Transformer's Multi-Head Attention. In 25th Workshop on Synthesis And System Integration of Mixed Information Technologies, pages 160–163, June 2024.

4) Hasitha Muthumala Waidyasooirya, Takuro Fukuda, and Masanori Hariyama. Scalable Architecture Targeting HBM-Based FPGAs for Complex Matrix Multiplication. In 23rd International Conference on Parallel and Distributed Computing, Applications and Technologies, December 2022.