5A-01

分散深層学習のためのワイヤースピードでの In-Network Computing の検討

田仲 顕至† 有川 勇輝† 川合 健治† 加藤 順一† 伊藤 猛† Huy Cu Ngo†
森田 和孝‡ 三浦 史光‡ 坂本 健† 重松 智志†

日本電信電話株式会社 NTT 先端集積デバイス研究所†
日本電信電話株式会社 NTT ソフトウェアイノベーションセンタ‡

1. はじめに

ディープラーニング（DL）は、その適用領域の広さから、様々なアプリケーションの開発が検討されている。DL モデルの学習には大量のデータと膨大な計算リソースが必要となることが知られており、その 1 つの解決策として大規模分散 DL が検討されている。中でも、データ並列同期更新型の分散深層学習が良好な性能を示している[1, 2]。このアプローチでは、各ワーカーノードで計算された勾配を共有するために、ワーカーノード間の集団通信（Allreduce）が発生する。Allreduce でやり取りされる勾配のメッセージサイズは数十 kB を超え、中には数 GB に達するものもある[3]。よって、大きなメッセージサイズでの Allreduce のスループットを向上させ、待ち時間を短縮することは重要である。

2. 既存手法

低遅延なノード間 Allreduce を実現する方法として、Mellanox 社は Scalable Hierarchical Aggregation Protocol（SHArP）を提案している[4]。SHArP はネットワークスイッチをデータの集約ノードとみなし、ノード間でのデータ転送中に reduce 処理を行うことができる。こうしたネットワーク側への集団通信の中間処理のオフローディングは In-Network Computing と呼ばれ、SHArP では 5 KB 未満のメッセージサイズでのレイテンシ短縮に成功している。しかしながら、SHArP は reduce 処理の前にスイッチ内のバッファにすべてのデータを格納するストアアンドフォワードバッファリングを行うため、データ格納にレイテンシが発生する[5]。そのために、メッセージサイズが増加するにつれて遅延が増大してしまい、分散深層学習などの用途では、低遅延なワイヤスピードでの Allreduce が困難になる[6]。

In-Network Computing at Wire Speed for Distributed Deep Learning
† NTT Device Technology Laboratories, NTT Corporation
‡ NTT Software Innovation Center, NTT Corporation

3. 提案手法

本稿では、大メッセージサイズでの In-Network Computing をワイヤスピードで実行するためのアーキテクチャを提案する。提案アーキテクチャでは、集約ノードにカットスルーバッファリングを実装することで、従来のデータ格納遅延を低減した。図 1（a）に、従来のストアアンドフォワードバッファリングとカットスルーバッファリングの違いを示す。カットスルーバッファリングでは、各ワーカーノードから送信されたデータフレームの先頭が到着した直後にベクトル和を開始する。これによりデータ格納時間が隠蔽され、低遅延化が可能となる。我々はこの処理をフィールドプログラマブルゲートアレイ（FPGA）に実装した。図 1（b）に FPGA 内での処理を示す。FPGA 内にカットスルーバッファリングを実装するために、データ受信バッファに閾値を設定し、バッファリングされたフレームの数が閾値を超えると、ベクトル和が開始される。このアーキテクチャにより、従来のストアアンドフォワード型に比べて低遅延な Allreduce を実現できる。

集約ノードの低遅延化に加えて、ワーカーノードにおける勾配データの転送スループット増大にも着手した。提案手法では、GPGPU・CPU・NIC に複数の送受信バッファを持たせ、GPGPU-CPU 間と CPU-NIC 間の転送を非同期に行った。これによりデータ転送スループットが向上し、大きなメッセージサイズの転送遅延を低減できる。

4. 評価

提案手法の評価のために、我々は Xeon CPU E5-2603 と、64 GB RAM、Tesla K20 GPU、10G Ethernet Controller からなるワーカーノードを構築し、VC709 FPGA にて集約ノードを実装した。また、SHArP 提案論文[10]で述べられているデータを線形に外挿し、参照データとして比較した。

図 1(c)はメッセージサイズを変えながら Allreduce を実行した際の提案系のスループットである。提案系のスループットは、メッセージ

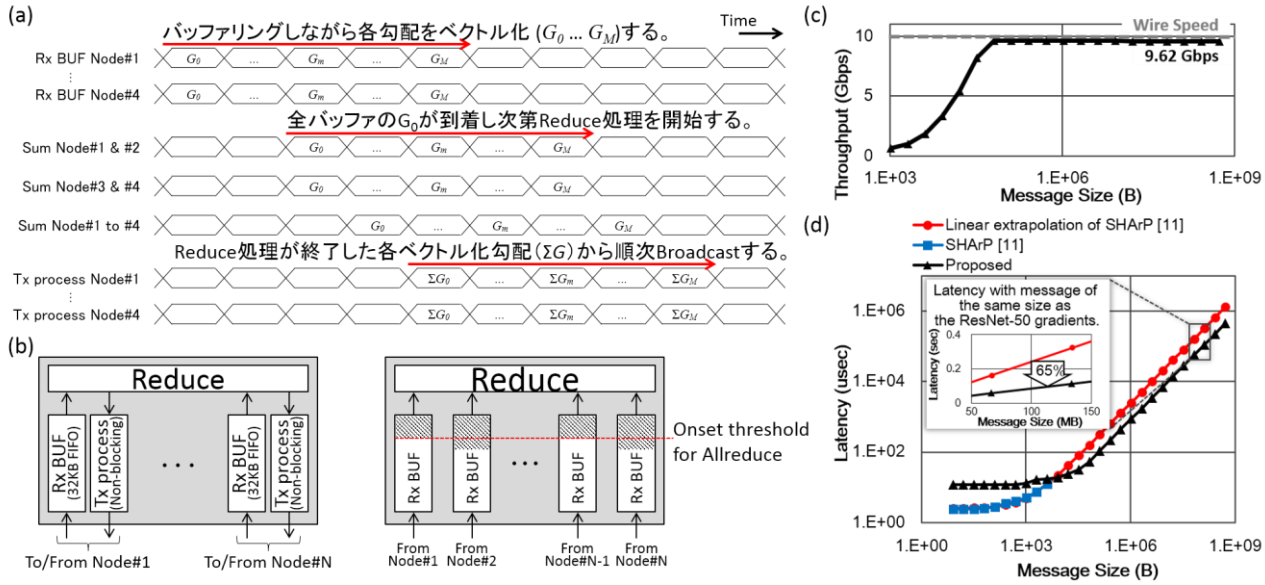

図 1: (a) 従来と提案バッファリングの比較 (b) 実装したデータフローの模式図 (c) 提案系の Allreduce スループット (d) 評価系と参照系の Allreduce 遅延の比較

サイズが 64KB 以上で 9.62Gbps であり、Allreduce がほぼワイヤスピードで動作することが示された。

また、図 1(d)に参照系と評価系のレイテンシの比較を示す。評価系は GPU メモリから 10 Gbps イーサネットを使用してデータを転送するのに対し、参照系では 100-Gbps InfiniBand を使用してメインメモリからデータを転送しているため、8 KB 未満のメッセージサイズでは評価系の遅延が参照系よりも大きくなる。 しかし、8 KB を超えるメッセージサイズでは、評価系の方が低遅延となる。一例として、ResNet-50 の勾配（約 100 MB）[3]と同じメッセージサイズでの Allreduce のレイテンシを比較したところ、評価系は参照系より 65%低遅延であった。

5. まとめと今後の課題

本研究はワイヤースピードで In-Network Computing を実行するためのハードウェアアーキテクチャとデータ転送方法を検討した。結果として、DL で用いられる大きなメッセージサイズの Allreduce において、従来よりも低遅延化することができた。今後は、参照系と同等の 100 G Ethernet を用いて評価を行う。加えて、ワーカーノードの数を増やした場合や、集約ノードを多段に構成した場合の性能のスケールに関して評価していく。

参考文献

[1] Tal Ben-Num, and Torsten Hoefler: Demystifying Parallel and Distributed Deep Learning: An In-Depth Concurrency Analysis, arXiv:cs.LG/1802.09941, (2018).

[2] Takuya Akiba, Shuji Suzuki, and Keisuke Fukuda: Extremely Large Minibatch SGD: Training ResNet-50 on ImageNet in 15 Minutes, Deep Learning at Supercomputer Scale (NIPS'17 Workshop), arXiv:cs.DC/1711.04325, (2017).

[3] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J. Dally: Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training, 6th International Conference on Learning Representations (ICLR'18), arXiv:cs.CV/1712.01887, (2017).

[4] Gil Bloch, Devendar Bureddy, Richard L. Graham, Gilad Shainer, and Brian Smith: Towards A Data Centric System Architecture: SHARP. Supercomputing Frontiers and Innovations: an International Journal, Vol. 4, No. 4, pp. 4-16 (2017).

[5] Gil Bloch, Diego Crupnicoff, Benny Koren, Oded Wertheim, Lion Levi, Richard Graham, and Michael Kagan: Aggregation protocol, Patent No. US20170063613A1, (2017).

[6] Mohammadreza Bayatpour, Sourav Chakraborty, Hari Subramoni, Xiaoyi Lu, Dhabaleswar K. (DK) Panda: Scalable Reduction Collectives with Data Partitioning-based Multi-Leader Design, In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC'17), (2017).