

ABCI上でのジョブ実行履歴の分析による 深層学習計算の傾向把握

滝澤 真一朗^{1,a)} 坂部 昌久¹ 谷村 勇輔¹ 小川 宏高¹

概要: HPC システムを設計するときには、対象とするシステムで実行する計算の特性を考慮して、ハードウェアやジョブスケジューリング等の資源管理の設計を行う。深層学習の普及に伴い、深層学習用の HPC システムの導入が進んでいくことが期待されるが、深層学習計算の歴史は新しく、計算の特性についての情報は少ない。本研究は、今後の深層学習用の HPC システムのハードウェアや資源管理を設計する際の一助となることを目的に、2019 年度に ABCI で実行された深層学習の訓練計算の特性について分析を行った。その結果、並列計算の割合は少なく、1 ノード以下の資源を利用する計算が 97.6%を占めること、94%の計算が 24 時間以内に完了していること、GPU 利用率の低いジョブも多くある、等の知見が得られた。

1. はじめに

画像認識、物体認識、機械翻訳、およびそれらを用いた応用問題等、実社会で求められている課題を計算機で高精度に解けることから、深層学習技術への期待が高まっている。最近の深層学習計算の実行環境としては、GPU を搭載したサーバや、それら複数のサーバからなるクラスターが広く使用されている。高性能な GPU は導入費用が高いことから、計算機資源としては、Amazon Web Services や Google などのクラウドや、産総研が運用する ABCI や東工大 Tsubame などの GPU を搭載したスパコンが使われている。一方で、研究室や会社内での計算需要の高まりにより、独自にオンプレミスのシステムの導入が進むことも期待されている。

深層学習計算向けに、高性能な GPU サーバからなる HPC システムを導入する場合、高い費用対効果を実現するためには、システムで実行する予定の深層学習ワークロードの特徴や規模、要求性能を元にハードウェア設計を行う必要がある。GPU 資源を管理する資源管理システムや、計算の実行管理を行うジョブスケジューリングシステムについても、仮想マシンやコンテナを用いて CPU や GPU、IO デバイス等を仮想化するソフトウェアが多数提案されているため、対象ワークロードが要求する機能・性能を考慮し取舍選択を行い、要求要件を満たすシステムを構築する必要がある。これから実行する予定の未知のワークロードの特徴予測のためには、既存システムにおける類似ワー

クロードの特徴分析情報が参考になる。しかしながら、特に大規模並列学習を含む、深層学習計算は歴史が浅いため、ワークロードに関して十分な知識が蓄積されていない。Microsoft や Facebook などの人工知能技術を活用する大手企業による、社内研究開発システムにおける深層学習計算のワークロード分析結果が公開されつつあるが [1–3]、組織ごとにワークロードやシステム構成が異なるため、知識蓄積のためには、事例は多いことが望ましい。

我々は、ABCI で実行された深層学習ワークロードを並列度、実行時間、GPU 利用状況の観点から分析を行った。ABCI は、国内における人工知能技術を用いた技術開発や産業を発展させることを目的としたシステムであり、産学官の研究者・開発者により人工知能の基礎研究から応用分野まで幅広い分野で利用されている。一方で公的機関である産総研が運用するため、ABCI 上での有償サービスのホスティングは認められていない。ABCI は、1 台のサーバを細かい単位で分割して利用できる点を除き、典型的なスーパーコンピュータのハードウェア・ソフトウェア構成を持つ大規模 HPC システムである。ABCI で 2019 年度に実行されたジョブから深層学習の訓練計算の特徴を持つジョブを抽出し、時系列に蓄積した全 GPU の利用状況データと結合することで、ワークロードを作成した。ワークロード分析の結果、以下が観測された。並列度に関しては高並列のジョブは少なく、全ジョブの 97.6%が 1 ノード以下のジョブであった。実行時間については、94%のジョブが 24 時間以内に完了しており、複数ノードジョブでは並列度と実行時間に弱い負の相関があった。GPU 利用状況について、複数 GPU が割り当てられたジョブでは 88%以上の

¹ 国立研究開発法人産業技術総合研究所

^{a)} shinichiro.takizawa@aist.go.jp

ジョブが全 GPU を使用しており、GPU 利用率はジョブ毎に異なり、利用率の低いジョブも多くあった。

2. 対象システム: ABCI

産総研が運用する、AI 橋渡しクラウド (AI Bridging Cloud Infrastructure, ABCI) [4–6] を深層学習ワークロード分析の対象とする。本章では、ワークロード分析に関連する ABCI の機能について説明する。

2.1 概要

ABCI は世界トップレベルの計算能力とデータ処理能力を有し、産学官共同の AI 研究開発を加速するためのオープンなプラットフォームであることを目指し運用している。そのための、AI 研究開発用の様々なソフトウェア、コンテナ技術、ビッグデータ蓄積・処理ソフトウェアをサポートし、学術・産業に広く利用されている。

ABCI は 1088 台の計算ノードから構成され、それらは InfiniBand EDR にて互いに接続されている。各ノードは 2 基の Intel Xeon Gold 6148 (計 80 コア)、4 基の NVIDIA V100、384GiB メモリ、1.6TB の NVMe SSD を搭載する。ABCI における、利用者からの計算要求はバッチジョブスケジューラにより管理されており、利用者が要求する資源が利用可能になった時点で計算を開始する。バッチジョブスケジューラには Univa Grid Engine を用いている。利用者は表 1 に示す資源タイプから 1 種類選択し、バッチジョブ、あるいは対話的に計算ノードを使用することができる。資源タイプ毎に異なる資源量、利用制限が設けられている^{*1}。資源タイプ Full 以外の 4 タイプを使用するジョブは、1 台の計算ノード上で複数同時に実行される可能性がある。ただし、オーバプロビジョニングはせず、表中の資源量が保証されている。

2.2 利用状況、および資源情報モニタリング

利用者が ABCI 上で実行したジョブの状況を把握するため、ジョブや資源利用状況のモニタリングを行っている。ABCI のモニタリングシステムを構成するシステムから、本研究に関連するシステムを図 1 に示す。ABCI では利用者が所有するデータとシステム運用上取得するデータ（以降、運用データ）について明確な区別をつけており、ABCI モニタリングシステムは後者を収集するシステムである。前者には利用者が持ち込んだプログラムやジョブスクリプト、ジョブの標準入出力、入出力データが含まれ、ABCI ではこれらは収集していない。

運用データには ABCI のサービス使用状況と各種機器のセンサーデータが該当する。サービス使用状況にはジョブ

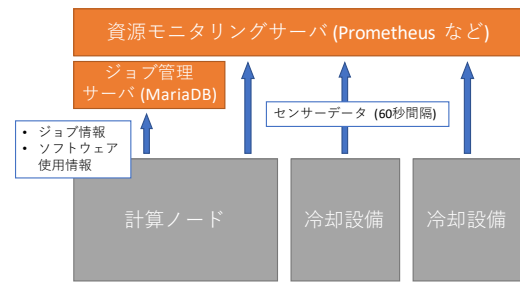


図 1: ABCI Monitoring System

情報とソフトウェア利用情報が含まれる。ジョブ情報としては、利用者が投入しバッチジョブスケジューラが実行管理したジョブの属性情報を収集しており、本研究では以下の情報を利用している。

- ユーザ ID
- ジョブ種別
- 資源タイプ種別・数量
- 投入, 実行開始, 終了時刻
- 実行時間
- 実行ノードリスト
- 使用 GPU リスト
- 終了コード

ABCI では、各種コンパイラや GPU プログラミング用ライブラリ、MPI ライブラリ等の利用頻度が高いと想定されるソフトウェアを Environment Modules [7] で提供しており、ジョブ毎に使用したモジュールをソフトウェア利用情報として収集している。これらサービス利用状況は、ジョブ毎に分類し、関係データベースの 1 つである MariaDB に蓄積している。

センサーデータとして、(1) 計算ノードの CPU、GPU 等の利用率や周波数、温度と、(2) 電力・冷却設備等の付帯設備の状況を示すデータを、1 分間隔の時系列データとして蓄積している。目的に応じて 2 種類のサービスを使い分けられている。短期的なデータ蓄積及び、障害等のアラート通知用に Zabbix を、長期データ蓄積に Prometheus [8] を採用している。本研究では Prometheus に蓄積した計算ノードの GPU 利用率を使用している。

2.2.1 計算ノードモニタリングの詳細

計算ノードのセンサーデータを蓄積する Prometheus の構成を述べる。Prometheus は管理サーバ 1 台で動作しており、このサーバから全 1088 台の計算ノードに問合せを行なう。稼働当初より、データベースのサイズは大きくなることが想定されていたため、データベースは ABCI の GPFS 分散ファイルシステム領域に構築した。2019 年間の運用実績より、1 年間で 1.4TB 以上の容量となることが判明したため、データベースは年度単位で分割することとした。Prometheus は蓄積するデータ量は期間で指定する仕様であり、過去のデータを失わないよう長期間に設定し

^{*1} 表はバッチジョブ実行時の資源制限を示している。対話的利用の場合は異なる。全ての資源タイプの標準の実行時間制限は 1 時間である。

表 1: ABCI Resource Types

Resource Type	#CPU cores	#GPUs	Memory (GiB)	Local SSD (GiB)	#Maximum Instances /Job	Walltime Limit (Max, hour)
Full	40	4	360	1440	512	72
G.large	20	4	240	720	1	72
G.small	5	1	60	180	1	168
C.large	20	0	120	720	1	72
C.small	5	0	30	180	1	168

ている*2.

計算ノードには Prometheus の Node Exporter を稼働させ、それから各種情報を得ている。Node Exporter はサーバの各種情報を Metrics として出力するデーモンとして動作し、標準設定で多数の Metrics を出力する。一方で、Prometheus は各 Exporter が出力する Metrics をそのまま格納するが、後から削除する手段が提供されていない。不必要なデータの蓄積を避けるよう、Node Exporter では我々が取得したい情報のみを出力するよう設定した。また、Node Exporter には GPU 情報を Metrics として出力する手段はない。GPU 情報を出力する Exporter は有志により多数開発されているが、Exporter ごとに別プロセス、別ポートで動作させる必要があるため、それらは使用せず、Node Exporter の機能の 1 つである Textfile Collector を用いて、nvidia-smi から得られる情報を集約している。Textfile Collector とは、定期的に更新されるテキストファイルの内容を Metrics として Prometheus に出力する機能である。CRON で 1 分間隔で nvidia-smi を実行し、その出力ファイルを Textfile Collector に与えている。GPU 情報については、Prometheus による観測時から、最大で 1 分の遅延が生じることとなる。

Prometheus と Node Exporter 間の通信は管理用 Ethernet 上で行われる。Prometheus はバージョン 2.7.1 から使い始め、現在は 2.18.1 が動作している。Node Exporter も同様に、バージョン 0.17.0 から使い始め、現在は 0.18.1 が動作している。

3. ワークロードの詳細

2019 年 4 月 5 日から 2020 年 3 月 31 日の 1 年間に ABCI 上で実行されたジョブから、深層学習の訓練計算を行っているジョブを抽出し、それらジョブ集合からなるワークロードを構成した。ただし、グラウンドチャンレンジ等、特定用途で資源占有して実行されたジョブは除外している。本ワークロードのことを「ABCI 深層学習ワークロード」と呼ぶ。

ジョブ抽出方法を以下に述べる。ジョブスクリプトやジョブの標準入出力を解析することでより正確なジョブの抽出を行うことが可能であるが、上述のデータ収集ポリシ

より、ABCI では実施できない。そのため、運用データをもとに、以下のすべての条件を満たすジョブを深層学習の訓練計算ジョブとして抽出した。

GPU を搭載する資源タイプを使用 訓練計算には GPU がよく利用されているため、GPU を搭載する資源タイプの使用を宣言するジョブを対象とした。具体的には表 1 中の Full, G.large, G.small である。

バッチジョブとして実行 インタラクティブジョブの実行時間には不使用時間も含まれるため、バッチジョブのみを対象とした。

CUDA を使用 GPU を使用するためには、ABCI が提供するいずれかのバージョンの CUDA を使用する必要がある。

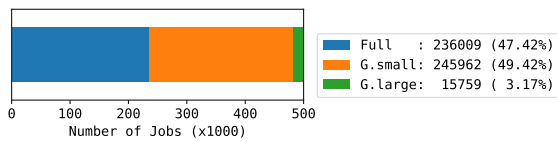
実行時間 300 秒以上 GPU 利用率を複数の観測値で評価することを目的に、300 秒以上の実行時間のジョブを対象とした。一般に訓練計算には長時間を要するため、これにより除外されるジョブによる、分析への影響は少ないと考える。

1 つ以上の GPU の利用率が 5% 以上 実際に GPU を用いた計算を行っているジョブを対象とした。具体的にはジョブに割り振られたノードが搭載する各 GPU に対して、次の計算より求めた利用率を評価している。(1) Prometheus からジョブ実行期間中の GPU 利用率を取得する。(2) 利用率の平均値 (μ) と標準偏差 (σ) を求め、 $[\mu - 2 \times \sigma, \mu + 2 \times \sigma]$ の範囲に収まる最大の値を求める。

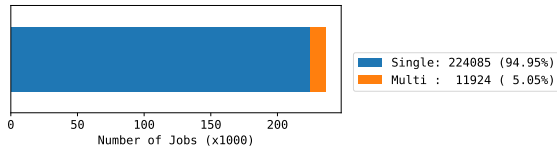
以上の抽出方法を採用するため、生成したワークロードは真に深層学習の訓練計算のみを集めたものである保証はない。例えば、GPU を使用する 300 秒以上のシミュレーションプログラムも該当する。しかしながら、ABCI は AI の研究開発目的に利用されているシステムであるため、抽出されたジョブの多くが訓練計算であると考えられる。

上記の期間に実行された総ジョブ数は 2,501,437 ジョブあったが、この抽出方法により深層学習ジョブとして分類されたジョブは 497,730 ジョブであった。ジョブ投入者数は 389 名であった。ジョブ投入者間のジョブ数の偏りは大きく、上位 6% の投入者のジョブが全体の 80% を占めていた。

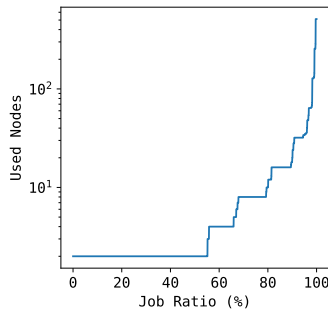
*2 現在の設定値は 10 年



(a) By Resource Types



(b) By Single or Multi Node of Full Jobs



(c) Used Nodes of Multi Nodes Full Jobs

図 2: Breakdown of Degree of Parallelism by Job Count

4. ワークロード分析と考察

深層学習の訓練計算による HPC システムの利用状況の把握を目的に、ABCI 深層学習ワークロードの分析を、5 つの視点より行った。

4.1 並列度

深層学習のアルゴリズムの研究開発では、複数 GPU、複数計算ノードを使用した分散深層学習についても研究されているが、深層学習技術の応用分野における分散深層学習の普及は限定的にも思われる。実際、著者らが産総研で運用する GPU クラスタ「AAIC」のワークロードを分析した結果では、単一 GPU を使用するジョブが支配的であった [9]。ABCI には企業ユーザも多くいるため、ABCI 深層学習ワークロードにおける並列度を調査することで、最先端のアルゴリズムの研究開発から、深層学習技術に応用したサービス・製品開発に求められる GPU・ノード数の規模を把握できると考えている。なお、本分析では、ジョブ投入時に利用者が申告した資源タイプとその数量を元に分析を行っている。

図 2 に結果を示す。図 2a は資源タイプごとのジョブ数の内訳であり、この結果より、1 GPU 搭載の G.small の利用が最も多く、ほぼ 50% であることがわかる。4 GPU 搭載されている Full と G.large の合計も 50% だが、実際に各ノードに搭載された 4 GPU 全て使用されているかどうか

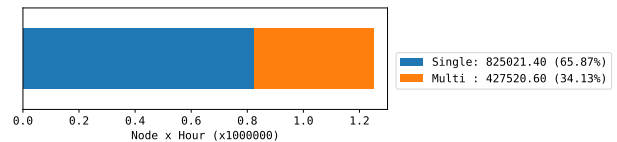


図 3: Breakdown of Single or Multi Node of Full Jobs by Node x Hour

かは、この結果からは判別できない。同じ GPU 数の Full と G.large の利用数に大きな差がある理由は、G.large では 1 ジョブは 1 ノードしか利用できないこと、これら資源タイプ間で CPU・メモリ等の性能差に対して価格差が小さいことが理由と考えている。図 2b に Full を使用するジョブにおける、1 ノード、複数ノード使用ジョブの数の割合を示す。1 ノードジョブが約 95% と支配的であることがわかる。また、この結果より、複数ノードジョブは全ジョブに対して 2.4% であることがわかった。図 3 は、Full を使用する 1 ノード・複数ノードジョブにおける、ノード時間積の合計値を示す。ジョブ数は少ないが、複数 Full 資源を使用するジョブによるノード時間積は、Full を使用するジョブの 34% を上回る。図 2c は、Full を使用する複数ノードジョブにおける、使用ノード数ごとのジョブの分布を示す。2 ノードジョブが最も多く、全体の 55% を示す。また、複数ノードジョブの 90.7% が 2 の冪乗のノード数を指定していた。

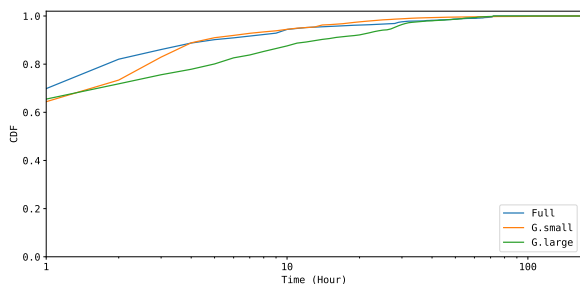
並列度の低いジョブが多数実行されていることより、ABCI 設計理念の 1 つでもあった「人工知能研究開発のためのキャパシティコンピューティングシステム」であることは十分達成できていると言える。また、近年出荷されている GPU サーバには複数台の GPU が搭載されたものも多いが、1 GPU ジョブが全体の半数となることから、ABCI で実施しているように、訓練計算ごとに GPU を割り振る資源管理の仕組みは重要になる。ABCI の設計に関して、1 ラック内には 34 ノード存在し、ラック内ではフルバイセクションで通信を行えるが、ラック間通信はラック内合算帯域の 1/3 となっている。今回の分析では、35 ノード以上を使用したジョブは 578 ジョブであり、全体の 0.12% であることから、ラック間帯域を制限したことは、費用対効果の点で正しい判断であったと考えている。

4.2 実行時間

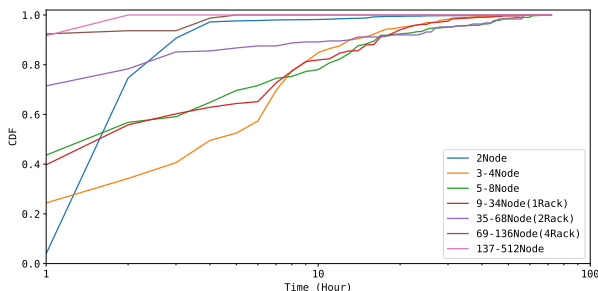
深層学習の訓練計算は、GPU を使用することで短縮はされるが、一般に長時間要するタスクであると認識されている。そのため、表 1 の通り、ABCI では 3 日以上ジョブの最大実行時間を許可している。また、事前予約サービスを使うことで、24 時間単位で最長 30 日、計算ノードを占有利用できる。実際の深層学習のアルゴリズムや応用分野の研究開発における、訓練計算の実行時間の傾向を調査する。

表 2: Walltime Statistics. Unit is Hour

Parallelism	Job type	Average	STD	Median	Min	Max
Single Node	G.small	2.514	7.533	0.495	0.083	167.385
	G.large	4.498	10.072	0.552	0.084	71.932
	Full 1Node	2.848	8.996	0.393	0.083	72.006
Multi Node	Full 2Node	2.212	3.040	1.711	0.084	71.998
	Full 3-4Node	6.024	7.102	4.013	0.084	53.502
	Full 5-8Node	6.304	11.034	1.347	0.084	72.006
	Full 9-34Node (1Rack)	5.538	8.355	1.582	0.083	70.662
	Full 35-68Node (2Rack)	4.108	10.779	0.345	0.085	56.007
	Full 69-136Node (4Rack)	0.441	0.902	0.173	0.084	4.346
	Full 137-512Node	0.316	0.364	0.219	0.084	1.943



(a) Distribution of Single Node Jobs



(b) Distribution of Multi Nodes Jobs

図 4: Distribution of Job Walltime

ジョブ実行時間について、図 4 に累積分布を、表 2 に統計を示す。1 ノードジョブの場合、G.large を使用しているジョブの実行時間が他に比べ長い、資源タイプによる傾向に大きな違いはない。図 4a より、全ての資源タイプに対して、約 70% のジョブが 1 時間以内に終了している。24 時間以内に終了しているジョブの割合は、Full で 96.6%、G.small で 98.3%、G.large で 93.9% であった。複数ノードジョブの場合、図 4b より、並列度に応じて傾向が大きく異なることがわかる。特に 2 ノードジョブにおいて、実行時間が 1~2 時間のジョブが 70% であることが特徴的である。一方で、共通して実行時間は長くはなく、複数ノードジョブの 98% は 24 時間以内に終了していた。

以上より、ABCI では 3 日や 7 日の最大実行時間を許可しているが、多くのジョブの需要はより短いことがわかった。また、ジョブの終了コードによると、実行時間を超過したジョブ数は 16,093 であり、全体の 3.23% であった。実

際に最大実行時間が足りずに時間超過したジョブ数はこれより小さいため、現在の最大値が足りないというジョブはごく限定的である。バッチジョブの最大実行時間は 24 時間とし、それを超えるジョブには事前予約サービスを利用させるサービス設計もできたと言える。

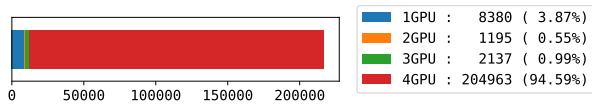
表 2 より、5 ノード以上のジョブにおいて、ノード数増加に反比例して実行時間が短縮されていることが見える。ノード数と実行時間の相関係数を求めることで、2 者の関係を定量的に評価する。Full を使用しているジョブに対して、表 2 と同じ並列度の区分で 8 分割し、各区分から 100 ジョブをランダムに選択した計 800 ジョブに対して、スピアマンの順位相関係数を求めた。結果、-0.386 であり、弱い負の相関を示すことがわかった。すなわち、表 2 の値から観測できた通り、並列度と実行時間には反比例の関係があり、利用者は複数ノードを使用する時は Strong Scale させて問題を解く傾向にあることがわかった。

4.3 GPU 利用数

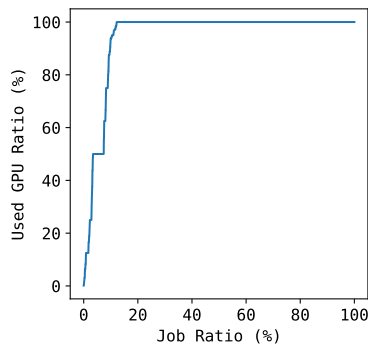
4.1 章の評価では調べられていない、ジョブで実際に使われた GPU の数を評価する。具体的には、ジョブスクリプトの終了コードが 0 であるジョブに対して、3 章で示した条件を満たす GPU の数をジョブ毎に数えた^{*3}。1 GPU しか搭載しない G.small ジョブは除き、G.large と Full の GPU 数の評価を行った。条件を満たすジョブ数は 216,675 であった。内、複数の Full を使用するジョブ数は 9,473 であった。

1 ノードジョブである G.large と Full 1 ノードの使用を宣言したジョブの結果を図 5a に示す。およそ 95% のジョブにて、全 4 GPU が使用されていることがわかる。3 以下の GPU を使用しているジョブには、ノードが搭載する GPU 以外の資源量が不足するために、使用 GPU 数を減らしているものもあると考えられる。例えば、強化学習において、訓練には 1 GPU しか使用しないが、Experience

^{*3} ジョブスクリプトの終了コードが 0 以外のジョブは、投入者の意図した動作をしていない可能性あり、正常な利用率を取得できないと考え除外した。



(a) Number of Used GPUs of 4 GPU Jobs



(b) Ratio of Used GPUs against Assigned GPUs of Multi Node Jobs

図 5: Breakdown of Used GPUs

Replay のために状態・行動・報酬等を蓄積するためのメモリ領域を大量に必要とするため、G.small で使用できるメモリでは足りずに Full を使用することはありうる。

図 5b に複数 Full を使用するジョブにおける、割り当てられた GPU 数に対する実際に使用された GPU 数の割合を示す。全 GPU を使用するジョブは約 88% あった。12% のジョブにて全 GPU が使用されていない理由としては、上述の通りの GPU 以外の資源量の不足による使用 GPU 数の削減や、分散学習での問題分割における不均衡が考えられる。

4.4 GPU 利用率

4.3 章で分析を行った同じジョブを対象に、深層学習の訓練計算による GPU 利用率を調査する。ここでは、ジョブ実行期間中に取得された時系列の GPU 利用率を平均化した値を評価する。複数 GPU を使用するジョブでは、上記の値を GPU 間平均するとともに、標準偏差を求めた。

GPU 利用率の平均値とジョブの割合を図 6 に示す。図 6a に示す 1 GPU ジョブでは、GPU 利用率に対して、ほぼ均等にジョブが分布していることがわかる。つまり、GPU 利用率が極端に低いジョブから高いジョブまで、ほぼ均等に存在している。図 6b の 4 GPU ジョブにおいても、1 GPU を使用しているジョブは類似の傾向を示す。4 GPU 使用しているジョブは極端に GPU 利用率の低ジョブの数は少ないが、一方で、2 または 3 GPU 使用しているジョブは GPU 利用率の低いジョブの割合の方が多い。図 7a に 4 GPU ジョブにおける、使用された GPU の利用率の標準偏差を示す。どのタイプのジョブも、75% のジョブの標準偏差は利用率 10% 以下であった。図 6c の複数ノードジョブは 80% の GPU 利用率を超えるジョブが 60% あり、他と

比べて利用率の高いジョブの割合が多い。図 7b で示すその標準偏差では、大きな例外値はあるものの、平均が小さいことから、同一ジョブにおける GPU 間の利用率のばらつきは平均的には小さいことがわかる。

GPU 利用率の低いジョブも多くあるため、システム全体の利用率向上のためには、1 つの GPU を複数のジョブで共有して使用することも考えた方がよい。その際には同一 GPU を共有するジョブ間での GPU メモリ領域の隔離などの、GPU メモリ利用による、他ジョブのメモリ枯渇発生を防ぐ機構も必要になる。今回は GPU メモリの分析は行っていないが、今後ジョブ毎の GPU メモリ利用状況や、GPU 利用率と GPU メモリ利用率の関連についても調査を行う予定である。

5. 関連研究

様々な観点から深層学習ワークロードの特性を分析する研究が行われている。Hazelwood らは、Facebook のオンラインサービスを提供する機械学習の訓練・推論ワークロードの特性を分析し、ワークロード実行に適したシステムを構築している [1]。Park らは、Facebook サービスでの深層学習の推論計算の特徴分類を行っており、推論性能向上のためのハードウェア・ソフトウェアのコードesignのための要求をまとめている [2]。Jeon らは、機械学習用の GPU クラスタの 2 ヶ月間のジョブ実行ログを分析し、スケジューリングにおける待ち時間の原因解析や複数 GPU ジョブにおけるローカリティの解析を行い、将来のスケジューリングシステム設計の指針を提案している [3]。Wang らは、Alibaba 社の深層学習用システムで実行された TensorFlow による訓練計算のワークロードの分析を、主に性能分析の観点から行っている [10]。Pumma らは、Caffe の分散実行時の I/O 性能低下の原因を特定し、性能向上手法を提案している [11]。Chien らは、1 ノード上での SSD, HDD, ストレージクラスメモリ等の様々なストレージデバイスにおける TensorFlow の I/O 性能の評価を行っている [12]。Chowdhury らは、並列ファイルシステム BeeGFS を対象に LBANN, TensorFlow による分散深層学習の I/O 性能の評価を行っている [13]。

本研究では、ABCI という、分野を限定しない汎用的な AI 向けシステムで実行された多数の訓練計算ジョブの統計分析を、古典的な HPC システムのワークロード分析手法を用いて行っている。特定の深層学習アプリケーションの性能分析については上記関連研究が参考になるが、本研究成果は深層学習向け HPC システムのキャパシティ設計やスケジューリング設計に貢献できると考えている。

HPC システムのワークロード分析については、著者らの先行研究 [9] で紹介した通り [14–19]、多数の既存研究がある。本研究は、これら既存研究で採用されてきた手法の一部を用いている。分析対象のワークロード種別が異なるだ

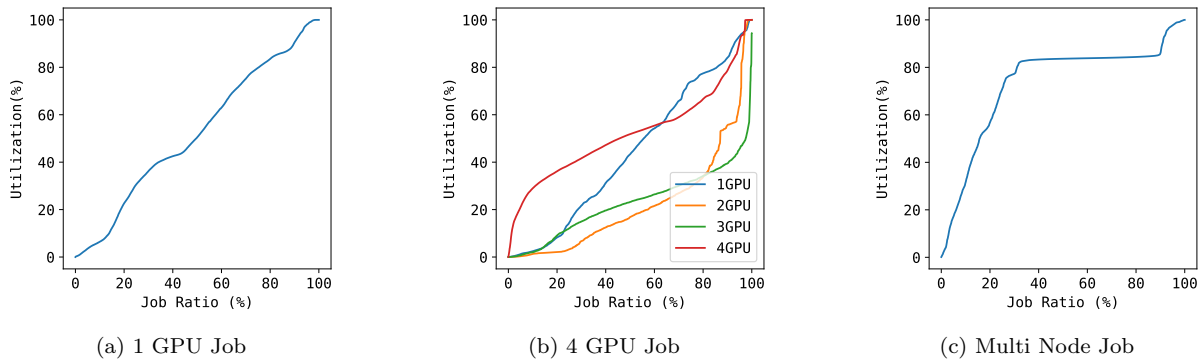


図 6: GPU Utilization

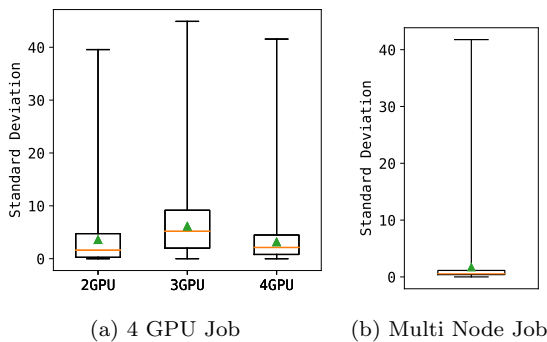


図 7: Standard Deviation of GPU Utilization

けではあるが、これほどの大量の大規模 HPC システムでの深層学習ジョブに対する分析を行っている研究は、我々の知る限りない。

6. まとめ

今後の深層学習計算用の HPC システム設計の指針となることを目的に、産総研が運用する ABCI にて 2019 年度に実行された深層学習の訓練計算ワークロードの分析を行った。分析は、並列度、実行時間、使用 GPU 数、GPU 利用率の観点から行い、次の知見が得られた。

- 全ジョブに対して、1 ノード以下の並列度の低いジョブが 97.6% を占める。具体的には、1 GPU ジョブが 49.42%、1 ノード (4 GPU) ジョブが 48.19% であった。
- 複数ノードジョブも並列度の低いジョブが多く、80% が 8 ノード以下のジョブである。
- 1GPU ジョブ、1 ノードジョブの 65% が 1 時間以内に終了している。それらジョブの 94% が 24 時間以内に終了する。複数ノードジョブの 98% が 24 時間以内に終了する。
- 複数ノードジョブでは、並列度と実行時間に弱い負の相関がある。利用者は Strong Scale させて問題を解く傾向にある。
- 複数 GPU が搭載されている場合、多くのジョブは全ての GPU を使用するが、一部の GPU しか使用しないジョブも観測された。ノードが搭載する GPU 以外

の資源を GPU 毎に分割したときに不足が生じる場合や、分散学習での問題分割での不均衡発生が理由と考えられる。

- GPU 利用率が低いジョブも多い。システム全体の利用率を向上させるには、単一 GPU を複数ジョブで共有することも考えた方がよい。

今後の課題として、GPU メモリ利用量や、ホストのメモリ使用量・I/O 量の蓄積も行い、深層学習の訓練計算におけるそれらの利用傾向の分析を行う。また、今回は GPU を使用する深層学習の訓練計算のみを対象としたが、CPU のみの資源タイプを含む、全ての資源タイプを対象としたワークロードの分析を行い、資源タイプの設計、最大実行時間、課金、スケジューリング等の改善を目的に運用にフィードバックする。さらには、ABCI を利用していた利用者が今後深層学習の研究開発に他クラウドを利用する、あるいはオンプレミスに HPC システムを導入する場合のシステム設計の支援となるよう、当該利用者・利用者組織による ABCI の利用傾向を可視化し提供するサービスを検討していきたい。

謝辞 本研究は JSPS 科研費 JP19H04121 の助成を受けたものです。

参考文献

- [1] Hazelwood, K., Bird, S., Brooks, D., Chintala, S., Diril, U., Dzhulgakov, D., Fawzy, M., Jia, B., Jia, Y., Kalro, A., Law, J., Lee, K., Lu, J., Noordhuis, P., Smelyanskiy, M., Xiong, L. and Wang, X.: Applied Machine Learning at Facebook: A Datacenter Infrastructure Perspective, *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 620–629 (2018).
- [2] Park, J., Naumov, M., Basu, P., Deng, S., Kalaiah, A., Khudia, D. S., Law, J., Malani, P., Malevich, A., Satish, N., Pino, J., Schatz, M., Sidorov, A., Sivakumar, V., Tulloch, A., Wang, X., Wu, Y., Yuen, H., Diril, U., Dzhulgakov, D., Hazelwood, K. M., Jia, B., Jia, Y., Qiao, L., Rao, V., Rotem, N., Yoo, S. and Smelyanskiy, M.: Deep Learning Inference in Facebook Data Centers: Characterization, Performance Optimizations and Hardware Implications, *CoRR* (2018).

- [3] Jeon, M., Venkataraman, S., Phanishayee, A., Qian, J., Xiao, W. and Yang, F.: Analysis of large-scale multi-tenant GPU clusters for DNN training workloads, *Proceedings of the 2019 USENIX Annual Technical Conference, USENIX ATC 2019* (2019).
- [4] 小川宏高, 松岡 聡, 佐藤 仁, 高野了成, 滝澤真一郎, 谷村勇輔, 三浦信一, 関口智嗣: AI 橋渡しクラウドー AI Bridging Cloud Infrastructure (ABCI) ーの構想, 第 160 回ハイパフォーマンスコンピューティング研究会 (2017).
- [5] 小川宏高, 松岡 聡, 佐藤 仁, 高野了成, 滝澤真一郎, 谷村勇輔, 三浦信一, 関口智嗣: 世界最大規模のオープン AI インフラストラクチャ AI 橋渡しクラウド (ABCI) の概要, 第 165 回ハイパフォーマンスコンピューティング研究会 (2018).
- [6] : ABCI, <https://abci.ai/>.
- [7] : Environment Modules, <http://modules.sourceforge.net/>.
- [8] : Prometheus, <https://prometheus.io/>.
- [9] 滝澤真一郎, 小川宏高, 高野了成: 大規模 GPU クラスタにおける深層学習ワークロードの傾向把握, 第 170 回ハイパフォーマンスコンピューティング研究会 (2019).
- [10] Wang, M., Meng, C., Long, G., Wu, C., Yang, J., Lin, W. and Jia, Y.: Characterizing Deep Learning Training Workloads on Alibaba-PAI, *arXive*, (online), available from (<http://arxiv.org/abs/1910.05930>) (2019).
- [11] Pumma, S., Si, M., Feng, W.-c. and Balaji, P.: Towards Scalable Deep Learning via I/O Analysis and Optimization, *2017 IEEE 19th International Conference on High Performance Computing and Communications; IEEE 15th International Conference on Smart City; IEEE 3rd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 223–230 (2017).
- [12] Chien, S. W. D., Markidis, S., Sishtla, C. P., Santos, L., Herman, P., Narasimhamurthy, S. and Laure, E.: Characterizing Deep-Learning I/O Workloads in TensorFlow, *2018 IEEE/ACM 3rd International Workshop on Parallel Data Storage & Data Intensive Scalable Computing Systems (PDSW-DISCS)*, IEEE, pp. 54–63 (2018).
- [13] Chowdhury, F., Zhu, Y., Heer, T., Paredes, S., Moody, A., Goldstone, R., Mohror, K. and Yu, W.: I/O Characterization and Performance Evaluation of BeeGFS for Deep Learning, *Proceedings of the 48th International Conference on Parallel Processing* (2019).
- [14] Li, H., Groep, D. and Wolters, L.: Workload Characteristics of a Multi-cluster Supercomputer, *Job Scheduling Strategies for Parallel Processing*, pp. 176–193 (2005).
- [15] You, H. and Zhang, H.: Comprehensive Workload Analysis and Modeling of a Petascale Supercomputer, *Job Scheduling Strategies for Parallel Processing*, pp. 253–271 (2013).
- [16] Rodrigo, G. P., Östberg, P.-O., Elmroth, E., Antypas, K., Gerber, R. and Ramakrishnan, L.: Towards understanding HPC users and systems: A NERSC case study, *Journal of Parallel and Distributed Computing*, Vol. 111, pp. 206–221 (2018).
- [17] Feng, J., Liu, G., Zhang, J., Zhang, Z., Yu, J. and Zhang, Z.: Workload Characterization and Evolutionary Analyses of Tianhe-1A Supercomputer, *Computational Science – ICCS 2018*, pp. 578–585 (2018).
- [18] Schlagkamp, S., da Silva, R. F., Allcock, W., Deelman, E. and Schwiegelshohn, U.: Consecutive Job Submission Behavior at Mira Supercomputer, *Proceedings of the 25th ACM International Symposium on High-Performance Parallel and Distributed Computing, HPDC '16*, pp. 93–96 (2016).
- [19] Schlagkamp, S., da Silva, R. F., Renker, J. and Rinke-nauer, G.: Analyzing Users in Parallel Computing: A User-Oriented Study, *2016 International Conference on High Performance Computing Simulation (HPCS)*, pp. 395–402 (2016).