# 豊橋技術科学大学における小規模研究用システムの運用

中村 純哉 [1)], 小林 真佐大 [1)], 土屋 雅稔 [1)]

1) 豊橋技術科学大学 情報メディア基盤センター

{junya, kobayashi, tsuchiya}@imc.tut.ac.jp

# Operation of Small Research System for Toyohashi University of Technology

Junya Nakamura[1)], Masahiro Kobayashi[1)], Masatoshi Tsuchiya[1)]

1) Information and Media Center, Toyohashi University of Technology

**概要**

科学研究における大規模計算処理の重要性は，シミュレーションの大規模化 · 高精度化および機械学習への応用という 2 つの要因から，近年ますます拡大している．それに伴って，大規模計算処理を行う研究用システムに対する要求も複雑化しており，多様な計算需要に応えられる柔軟な研究用システムが求められている．そのような研究用システムを実現するには，コンテナ環境と連携し，対話型ジョブに対応したジョブスケジューラと，仮想化アプリケーションの配信システムの 2 点が重要である．本稿では，この 2 点を実装した小規模研究用システムの運用状況について報告する．

## 1 はじめに

近年，科学研究における大規模計算処理（High Performance Computing; HPC）の役割は，重要になる一方である．特に，深層学習を始めとする機械学習の応用による影響は大きい．画像生成 [7]，自然言語理解 [4, 2]，ゲーム [3] などの各種の分野において，大規模な深層学習モデルの利用が広がっている．初期の小規模モデルは研究室規模の計算機環境でも学習できたが，近年の大規模モデルを学習するには，大規模な学習データを高速に入出力する I/O 性能と，大規模な計算を高速に実施する計算能力を兼ね備えた HPC 環境が必須である．

これらの大規模な計算需要に対応するため，アクセラレータとして GPU を搭載したクラスタシステムが広く採用されている [11, 1]．このようなシステムでは，(1) 従来のソフトウェアスタックと GPU を利用するソフトウェアスタックの性質の相違，(2) ソフトウェア作成時に利用する GPU に対する排他制御の必要性，という 2 つの理由から，従来のクラスタシステムとは異なった設計が必要になる．加えて，従来は共用ワークステーションを用意していた商用アプリケーションについては，利便性と維持コストの問題から，オンラインで利用できる環境を整備することが要請されている．これらの課題に対して，筆者らは，コンテナ環境と連携し，対話型ジョブに対応したジョブスケジューラと，仮想化アプリケーション配信システムによって対応することを提案している [10].

本稿は, 筆者らが提案する小規模研究用システム（以下，本システムという）の 2022 年の運用状況について報告する．本稿の構成は，以下の通りである．2 節で本システムの概要について説明し，3 節で本システムの利用状況について考察する．最後に，4 節で結論を述べる．

## 2 システム概要

この節では，本システムの概要について説明する．本システムの概要は，**図 1** の通りである．本システムは，アクセラレータとして GPU を搭載した小規模クラスタシステムと，仮想化アプリケーションの配信システムからなる．GPU を搭載したクラスタシステムの利用上の問題点を解決するため，ソフトウェアスタックをコンテナ化した上で，計算ノード上の対話型ジョブとしてソフトウェアの作成を行うという方針で設計している．加えて，クラスタシステムでは処理困難な，GUI によるインタラクティブな処理が重要なソフトウェアについては，仮想化アプリケーションを配信 · 提供している．

### 2.1 クラスタシステムのハードウェア仕様

この節では，クラスタシステムのハードウェア仕様について述べる．

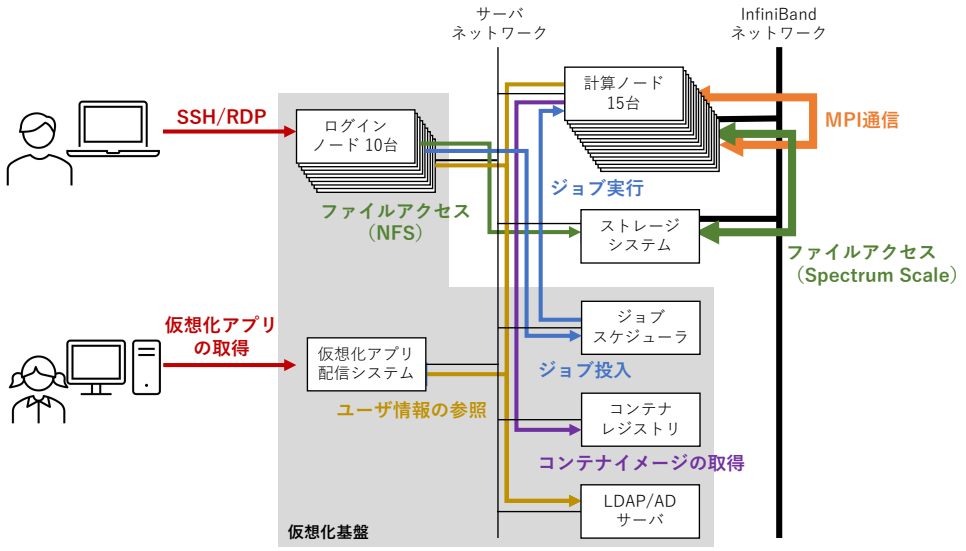


図 1　システム構成

表 1　システム構成

| 項目 | 台数 |
|---|---|
| ログインノード | 10 |
| 計算ノード | 15 |

表 2　ログインノード仕様

| 項目 | 仕様 |
|---|---|
| 機種名 | 仮想マシン（VMware） |
| CPU | Intel Xeon Gold 6148 2.40 GHz 4 コア |
| GPU | なし |
| メモリ | DDR4-2400 8 GiB |
| ローカルディスク | 200 GiB |
| OS | CentOS Linux 7 |

表 3　計算ノード仕様

| 項目 | 仕様 |
|---|---|
| 機種名 | DELL PowerEdge R740 |
| CPU | Intel Xeon Gold 6132 2.60 GHz 14 コア × 2 |
| GPU | NVIDIA Tesla V100 HBM2 16GB × 2 |
| メモリ | DDR4-2666 192 GiB |
| ローカルディスク | 256 GiB (RAID 5) |
| OS | RedHat Enterprise Linux 7 |
| コンテナ環境 | Singularity 3.3.0, Docker 18.09 |
| NIC | Intel X710 |
| インターコネクト | InfiniBand EDR |

表 4　ストレージシステム構成

| 項目 | 仕様 |
|---|---|
| ソフトウェア | DDN GridScaler |
| ファイルシステム | IBM Spectrum Scale |
| ストレージアレイ | DDN SFA14KX |
| 総容量 | 2PB |
| SSD | 1920 GiB × 13 |
| HDD | 8 TB NL-SAS × 325 (RAID 6, 8D+2P) |
| インターコネクト | InfiniBand EDR |

図 1 に示す通り，クラスタシステムは，ログインノード，計算ノード，インターコネクトおよびストレージシステムからなる標準的な構成である．**表 1** に，ログインノードおよび計算ノードの台数を示す．本学では，研究用途だけでなく，UNIX システムの CLI 演習用途にもログインノードを併用しているため，ログインノードを多数ユーザが同時に利用する状況が想定される．そのため，同等規模のクラスタシステムに比べて，多数のログインノードを用意している．

ログインノードの仕様を，**表 2** に示す．本システムでは，自家製アプリケーションの開発・デバッグ作業は，計算ノード上で実行する対話型ジョブによって対応する．そのため，ログインノードには，アクセラレータを搭載しない限定された構成の仮想マシンを採用している．

計算ノードの仕様を，**表 3** に示す．大規模な行列演算を行うことを考慮して，アクセラレータとして NVIDIA Tesla V100 を搭載している．また，複数ノードを同時に利用した並列演算および高速・大規模なファイルアクセスの必要性を考慮し，インターコネクトは InfiniBand EDR を採用している．

ストレージシステムの仕様を，**表 4** に示す．予算上の制約から，本システムのワーク領域と，本システムおよび教育用端末のホーム領域を，同一のストレージシステム上に確保せざるを得なかった．教育用端末のホーム領域については，ユーザが誤って削除したファイルを救済するため，スナップショット機能が必要である．そのため，スナップショット機能を備えた高速

並列ファイルシステムとして，IBM Spectrum Scale[*1]を採用している．

### 2.2 クラスタシステムのキュー構成

この節では，クラスタシステムのジョブを管理するジョブスケジューラのキュー構成について述べる．

キュー構成を，**表 5** に示す．本システムで用意されているキューは，教育用キューと研究用キューの 2 系統に大別できる．教育用キューは，ログインノードにログインできる利用者全員が，小規模ジョブを投入できるキューである．それに対して，研究用キューは，研究利用登録済み利用者のみが，大規模ジョブを投入できるキューである．

本システムのジョブスケジューラにジョブを投入する時，(1) 当該ジョブの種別（対話型ジョブまたはバッチジョブの別），(2) 当該ジョブを実行するコンテナ環境，(3) 当該ジョブが利用する予定の計算資源，という 3 種類の情報の指定を要する．キュー毎に指定できる 1 ジョブあたり計算資源の上限値を，表 5 に示す．GPU 数については，CPU コア数等の他の計算資源とは異なり，零の指定も可能である．コンテナ環境と連携し，予約されていない計算資源については，当該ジョブからは参照できないようになっている．よって，CPU を主として利用するジョブと，GPU を主として利用するジョブが，1 台の計算ノードを同時に共用することも可能である．

本システムでは，自家製アプリケーションの作成作業は，計算ノード上の対話型ジョブとして実行することを想定している．そのため，対話型ジョブは，ジョブ投入直後に利用を開始できる必要がある．ただし，対話型ジョブは，ユーザによるコマンド入力待ち時間が発生するため，計算資源の利用効率は低く，長時間の対話型ジョブにより計算ノードを占有する使い方は好ましくない．つまり，対話型ジョブは，実行時間を厳しく制限した上で，ジョブ投入から実行までの待ち時間を短くすることが適当である．この要請は，クラスタシステムを用いる演習や講習会におけるジョブと共通している．以上より，本システムでは，第 1 に，対話型ジョブの実行は，教育用キューのみに制限している．この制限は，長時間の対話型ジョブを禁止することを意図している．第 2 に，一部の計算ノードを，教育用キューの予約ノードとして確保し，研究用キューのジョブの実行を禁止している．この制限は，教育用キューおよび対話型ジョブの待ち時間を短縮することを意図している．

表 5　キュー構成

| | 教育用 | 研究用 |
|---|---|---|
| 実行時間 [時間] | $\leq 8$ | $\leq 336$ |
| ノード数 | $\leq 2$ | $\leq 13$ |
| CPU コア数 | $\leq 56$ | $\leq 364$ |
| GPU 数 | $\leq 4$ | $\leq 26$ |
| メモリ [GB] | $\leq 384$ | $\leq 2496$ |
| 対話型ジョブの可否 | 可 | 不可 |
| 予約ノードの有無 | 有（2 ノード） | 無 |

### 2.3 クラスタシステムのコンテナ環境

本システムでは，コンテナ環境として Docker[6] と Singularity[5] の両方に対応している．これは，本システム設計時（2018 年）におけるコンテナ環境の完成度から，片方に限定することは不安がある状況だったことが理由である．また，本システムで利用できるコンテナイメージは，専用のコンテナレジストリ（図 1）に登録されたコンテナイメージに限定しており，ユーザが作成したコンテナイメージを自由に実行することは許可していない．この制限は，本システムの設計時にはまだ，マルウェアなどの不正なコードが混入したコンテナイメージでも安全に実行できる機能[*2]が未実装だったことが理由である．

本システムで利用できるコンテナイメージの内訳と概要を，**表 6** に示す．本システムには，2022 年 10 月現在，68 種類のコンテナイメージがインストールされている．これらのイメージは，(1) NVIDIA NGC[*3]で公開されているコンテナイメージ（33 種類）と，(2) 管理者が登録したコンテナイメージ（35 種類）の 2 系統に大別できる．

### 2.4 仮想化アプリケーション配信システム

アプリケーション仮想化の実装としては，Numecent Cloudpaging[*4]，Microsoft Application Virtualization (App-V)[*5]，VMware ThinApp[*6]などがある．この中から，本論文の目的と合致する実装として，実行端末上の GPU を利用することができる Cloudpaging を採用する．Cloudpaging では，アプリケーションはパッケージ化された後，配信サーバを通じて配信され，実行端末にインストールされた軽量のエージェントソフトウェアを介して仮想化アプリケーションとして実行される．パッケージ化したアプリケーションの

[*1] https://www.ibm.com/products/spectrum-scale

[*2] Rootless Docker

[*3] https://catalog.ngc.nvidia.com/

[*4] https://www.numecent.com/cloudpaging/

[*5] https://docs.microsoft.com/ja-jp/windows/application-management/app-v/appv-for-windows

[*6] https://www.vmware.com/jp/products/thinapp.html

表 6　コンテナイメージの内訳

| 分類 | 種類数 | 概要 |
|---|---|---|
| ログインノード | 2 | ログインノードと同等の環境 |
| 科学計算用 | 16 | 商用アプリケーションの実行環境 |
| 並列計算用 | 5 | MPI ライブラリを用いた並列計算環境 |
| CUDA | 13 | CUDA ライブラリを用いたプログラムの開発・実行環境 |
| PyTorch | 12 | 深層学習ライブラリ PyTorch による深層学習環境 |
| Anaconda | 1 | 科学計算向け Python 実装 Anaconda による開発・実行環境 |
| その他 | 19 | その他の機械学習ライブラリなど |
| 計 | 68 | |

5%〜10% 程度の容量があれば，アプリケーションを実行することができ，必要に応じてバックエンドで残りのパッケージがダウンロードされる．なお，一度ダウンロードされれば，オフライン環境でも実行可能である．

## 3　利用状況

本システムの仕様策定は，2018 年 4 月から 2019 年 3 月にかけて行った．調達は，予算の制約から分割して実施することになったが，本システムを構成するクラスタシステムのほとんどの部分は，2019 年 9 月に運用を開始した．仮想化アプリケーション配信システムについては，2021 年 11 月に運用を開始した．この節では，本システムの実際の利用状況を説明した上で，設計方針および仕様の妥当性について検討する．また，この節におけるデータの集計期間は 2022 年 1 月から 8 月である．

### 3.1　クラスタシステムの利用状況

この節では，クラスタシステムの利用状況について述べる．ログインノードに対する総ログイン時間の推移を**図 2** に示す．なお，教育研究目的での利用状況を調査するため，管理者によるログインは除外して集計した．この期間中におけるクラスタシステムの利用者は，学生 394 人，教職員 18 人，その他利用者 5 人，合計 417 人であった．本学では，卒業研究発表会は 12 月に，修士論文審査会は 2 月に行われる．図 2 から，総ログイン時間のピークは 1 月と 7 月に観察される．1 月のピークは修士論文審査会に向けての利用と考えられ，3 月の修了に伴い学生の利用が減少し，その後増加していることがわかる．

同様の傾向は，ジョブの実行状況からも読み取れる．ジョブ数の推移を**図 3** に示す．異常終了したと推定されるジョブを除外するため，実行時間が 5 分以内のジョブは除外して集計した．期間中に実行されたジョブは 27078 件，そのうち集計対象のジョブは 8737 件だった．図 3 より，1 月をピークに 3 月の修了に向けてジョブ数が減少している様子が見られ，その後増加している．ただし，5 月から 7 月はほぼ横ばいとなっている．また，総ログイン時間の推移（図 2）とジョブ数の推移（図 3）を比較すると，総ログイン時間の推移の内訳は学生が過半数を占めるのに対して，ジョブ数の推移の内訳は教職員が多数を占めることがわかる．特に，この傾向は学生が卒業 · 修了した 4 月以降に顕著である．

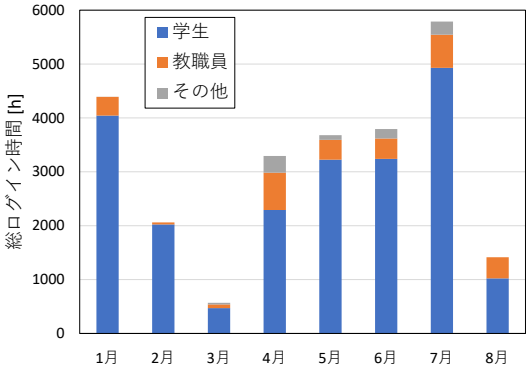


図 2　利用者種別ごとの総ログイン時間の推移

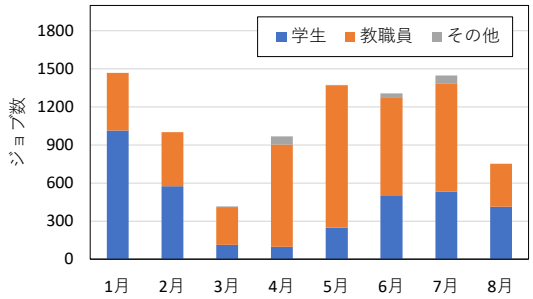


図 3　利用者種別ごとのジョブ数の推移

### 3.2　対話型ジョブの利用状況

ジョブ種別と GPU の関係を，**図 4** に示す．教育用キューに投入されたジョブについて観察すると，GPU を必要とする場合は，91% のジョブが対話型ジョブとして実行されていた．それに対して，GPU を必要としない場合は，20% のジョブのみが対話型ジョブとして

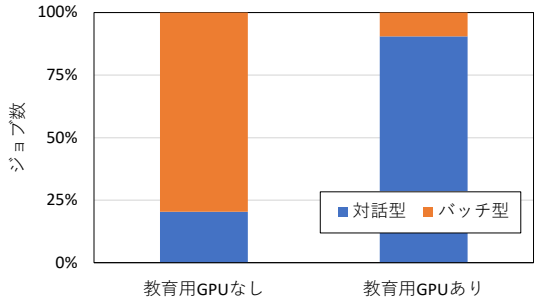


図 4　ジョブ種別と GPU の必要性の関係

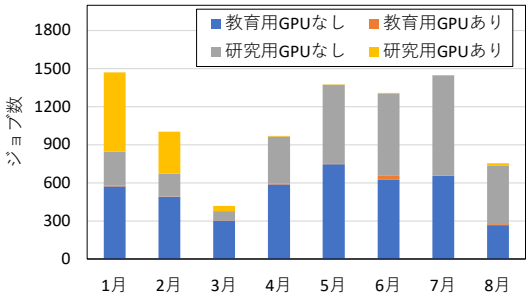


図 6　キューごとのジョブ数の推移

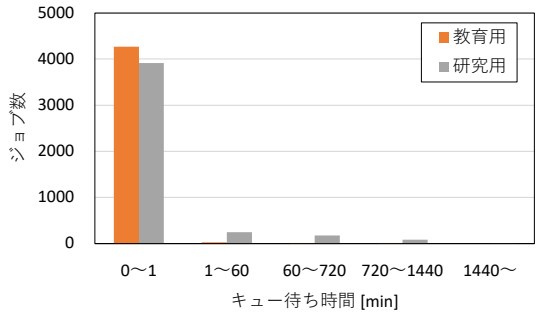


図 5　ジョブ種別と待ち時間の関係

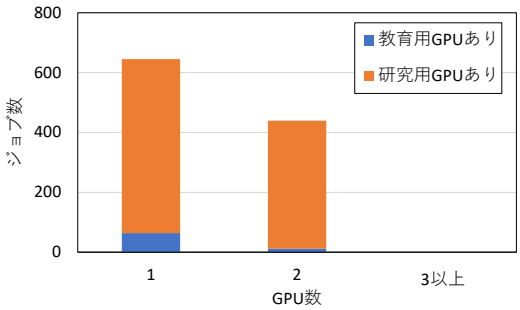


図 7　要求 GPU 数の分布

実行されていた．このように，短時間ジョブの実行状況において，GPU を必要とする場合と GPU を必要としない場合を比較すると，対話型ジョブの出現率に顕著な違いがある．これは，GPU を利用する自家製アプリケーションの大部分がスクリプト言語によって作成されており，開発・デバッグ作業のために，実際の実行環境と同一の環境を必要とするためと解釈できる．

ジョブ種別とキューに投入されてから実行開始までの待ち時間の関係を，**図 5** に示す．教育用キューに投入されたジョブについて見ると，99% のジョブは，キューに投入されてから実行開始までの待ち時間が 1 分未満だった．これは，教育用キューの待ち時間を短縮するために，予約ノードを用意した方針（2.2 節）の妥当性を示している．

### 3.3　GPU の利用状況

教育用キューと研究用キューに投入されたジョブを対象として，GPU の要不要に基づいて分類した場合のジョブ数の推移を**図 6** に示す．図 6 から，1 月と 2 月には，研究用に GPU が利用されているが，それ以降の月ではほぼ利用されていないことがわかる．これは，学生が 2 月の修士論文審査会に向けて GPU を利用していたが，3 月の修了に伴い GPU の利用者がいなくなったためであると考えられる．実際，図 3 のジョブ数の推移の内訳を見ると，1 月と 2 月は学生の利用が多数を占めるのに対して，3 月以降は教職員の利用が過半数を占めるのがわかる．したがって，次に GPU の利用が活発になるのは卒業研究が活発になる 10 月から 12 月であると考えられる．

### 3.4　プログラムの並列化状況

GPU を要求しているジョブを対象として，要求 GPU 数によってジョブを分類した結果を，**図 7** に示す．図 7 より，ジョブが要求する GPU 数は 1 基か 2 基のみであり，3 基以上は存在しないことがわかる．表 3 に示す通り，本システムの計算ノードはノードあたり 2 基の GPU を搭載している．近年の GPU 向けフレームワーク層の多くはノード内の複数 GPU に対応しているため，利用者は特別な対応をすることなく自然に，ノード内の複数 GPU を利用した並列化が利用できる．しかし，複数ノードにまたがる GPU の並列化については，本論文の執筆時点において，フレームワーク層によるサポートは充実していない．図 7 は，上記のような GPU 向けフレームワーク層の状況を反映していると考えられる．

要求ノード数によってジョブを分類した結果を**図 8** に，要求 CPU コア数によってジョブを分類した結果を**図 9** に示す．図 8 より，複数の計算ノードにまたがる並列化を行っている利用者は，非常に限られていることがわかる．同様の傾向は，図 9 からも確認できる．

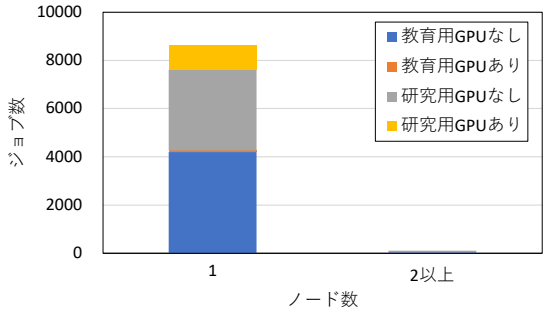


図 8　要求ノード数の分布

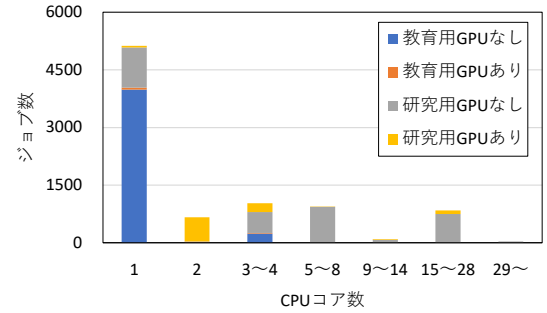


図 9　要求 CPU コア数の分布

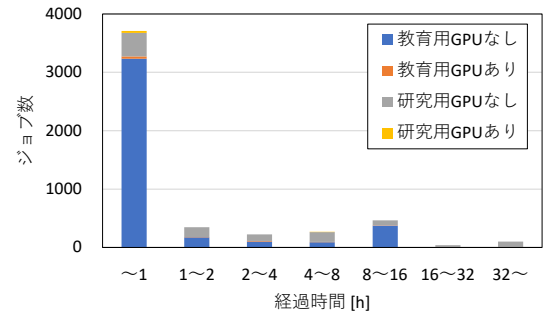


図 10　CPU コアを 1 つしか利用しないジョブの経過時間の分布

表 3 に示す通り，本システムの計算ノードはノードあたり 28 コアを有しているが，複数ノードを同時に確保する必要がある 29 コア以上の CPU コアを要求しているジョブは 0.4% しかない．なお，ノード内の複数 CPU コアを用いた並列化を行っているジョブ（すなわち，2 ～ 28 コアを要求しているジョブ）は，41% である．よって，本システムにおいては，ノード内での CPU の並列化はある程度利用されているものの，複数ノードにまたがる CPU の並列化は，ほとんど利用されていないことがわかる．したがって，複数の計算ノードにまたがる並列化を想定した設計方針を改めるか，または，複数の計算ノードにまたがる並列化を行えるように利用者支援を充実するか，どちらかの対応が必要と考えられる．

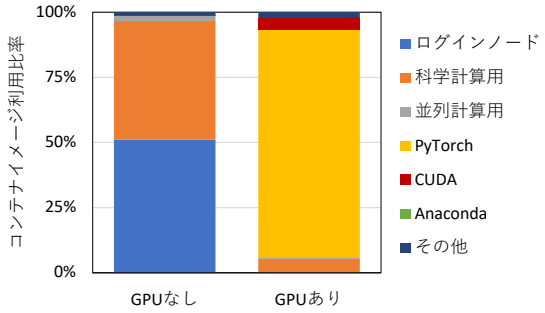


図 11　コンテナイメージの利用率

図 9 から，CPU コアを 1 つしか使わないジョブが多数実行されていることがわかる．そこで，1 コアのみ利用しているジョブを対象として，経過時間によってジョブを分類した結果を，**図 10** に示す．図 10 より，CPU コアを 1 つしか使わないジョブは，その経過時間もきわめて短いことがわかる．利用者に利用状況をヒアリングしたところ，多数のファイルを対象として集計を行う実験を行っており，対象となるファイル集合を保持しているマスタージョブと，ファイルごとの集計処理を行うスレーブジョブからなる，マスター・スレーブ型の並列化をジョブスケジューラを利用して実現しているという回答を得た．このような I/O 集約的な処理は，Hadoop などの専用フレームワークを利用することが多いが [9, 8]，高性能な並列ファイルシステムと汎用ジョブスケジューラの組み合わせも有効であることが判明した．

### 3.5　コンテナの利用状況

本システムでは，ジョブの種別に関係なく，全てのジョブはコンテナ環境で実行される．ジョブが参照しているコンテナの比率を，**図 11** に示す．本システムでは，2022 年 10 月現在，68 種類のコンテナイメージが利用できるが（表 6），図 11 は，それらのコンテナイメージを機能別に分類した結果である．

図 11 から，GPU を利用しないジョブの場合，ログインノードと同様の環境を提供するためのログインノード相当のコンテナが 50% 利用されていることがわかる．このジョブの多くは，I/O 集約的なジョブ（3.4 節）であり，ログインノード上で作成したソースを，そのまま計算ノード上で実行する使い方をしている．　2.1 節で述べた通り，本システムのログインノードは，仮想化基盤上の仮想マシンであり，ストレージシステムに対するファイルアクセスはイーサネットを経由した NFS プロトコルで行われる．それに対して，計算ノード上でバッチジョブとして実行した場合は，

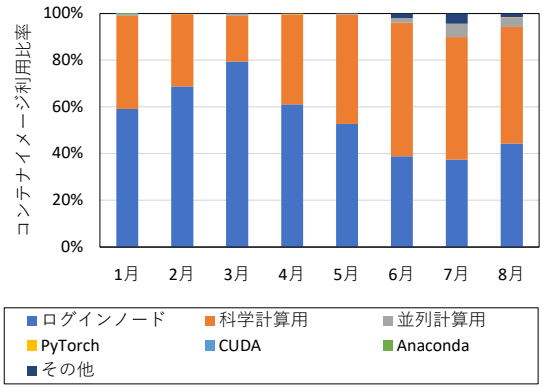


図 12　コンテナイメージ利用率の推移（GPU なし）

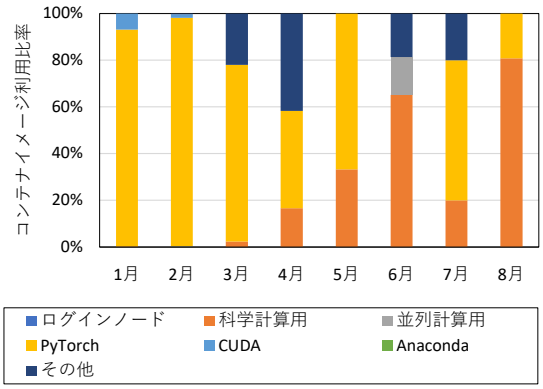


図 13　コンテナイメージの利用率の推移（GPU あり）

ファイルアクセスは InfiniBand を経由した Spectrum Scale プロトコルで行われる．そのため，I/O 集約的な処理を実行するには，計算ノード上でバッチジョブとして実行することが有利である．また，本システムでは計算資源として GPU を確保することなくジョブを実行することができるため（2.2 節），このような使い方によって計算資源が無駄になることはない． 商用アプリケーションがインストールされている科学計算用コンテナは 45%，MPI ライブラリがインストールされている並列計算用コンテナは 2% 利用されている．

**図 12** および**図 13** に，それぞれのコンテナイメージの利用割合の推移を示す．利用頻度の高いイメージは常に同じではなく，時期によって使われるイメージは大きく異なることがわかる．特に GPU ありキューの移り変わりは大きい．期間前半は PyTorch が主に利用されていたが，期間後半は科学計算用やその他イメージの利用頻度が増している．本学は小規模大学であるために，このように時期によって利用者の需要が変わってしまうと考えられる．

図 11 から，GPU を利用するジョブの場合，PyTorch がインストールされているコンテナが 87% 利用されており，最も利用率が高い．また，ライブラリ層のみがインストールされている CUDA コンテナは 4% 利用されている．CUDA コンテナの利用者は，フレームワーク層およびアプリケーション層を各自の領域に独自にインストールする必要があるが，フレームワーク層およびアプリケーション層を自由に選択できる．Python では，そのためのモジュール（venv）が用意されているため，そのような使い方も比較的容易である．このように，コンテナ技術の導入によって，利用者各自が必要とするソフトウェアスタックを柔軟に利用できていることがわかる．

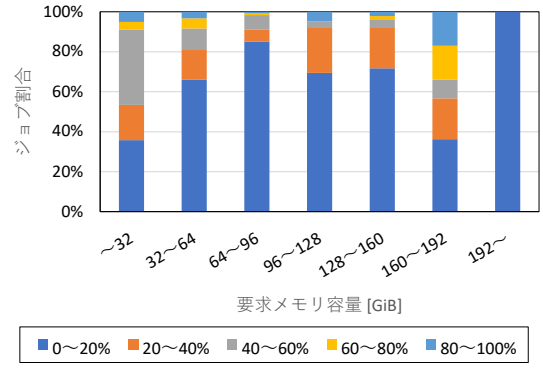


図 14　ジョブごとのメモリ利用状況

### 3.6　メモリの利用状況

ジョブ投入時に利用者が要求したメモリ容量と，実際に利用されたメモリ容量の関係を，**図 14** に示す．図では，要求メモリ容量に対する利用メモリ容量の割合（以下，実利用割合という）によってジョブを 20% ごとに 5 つのクラスに分類し，各クラスが全体に占める割合を示した．図 14 から，要求メモリ容量が 32 GiB 以下の場合は，要求したメモリ容量と実際に利用された容量の一致度が高く，適切に計算資源を利用できていることがわかる．一方で，64 GiB, 128 GiB，196 GiB など，切りの良い要求メモリ容量において，実利用割合の低いジョブが増加する傾向が見られる．これは実行するジョブの傾向に基づいて利用者が適切に要求メモリ容量を調整せずに，サンプルコードなどの記載をそのまま流用したためと考えられる．なかでも，192 GiB を指定して実行されるジョブが多く，全ジョブの約 5% を占めた．図 5 に見られるように，集計期間中の待ち時間は短かったため大きな問題とはならないが，利用者が増え多くのジョブが投入されるようになると，非効率な計算資源の指定はジョブの待ち時間を増加させ，利用者も不満を持つ．そのような場合に

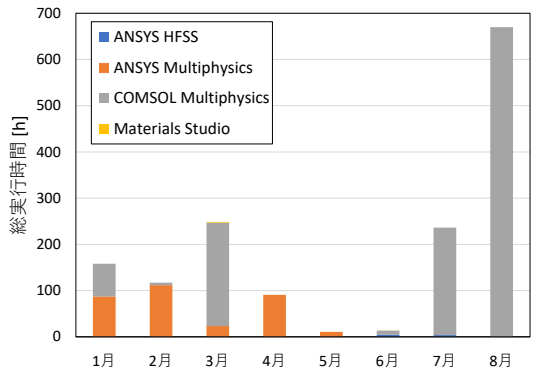


図 15　アプリケーションごとの総実行時間の推移

は，投入したジョブの実利用割合を利用者にフィードバックしたり，実利用割合が低い利用者に何らかのペナルティを課すなど，今後何らかの検討が必要となると予想される．

### 3.7　仮想化アプリケーション配信システムの利用状況

この節では，仮想化アプリケーション配信システム（以下，配信システムという）の利用状況について述べる．配信しているアプリケーションは，ANSYS HFSS，ANSYS Multiphysics，COMSOL Multiphysics，Materials Studio の 4 種類である．**図 15** にアプリケーションごとの配信システムの総実行時間を示す．なお，実行時間が 0 秒のデータに関しては集計から除外した．図 15 より，総実行時間は 5 月の利用が最も少なく，逆に 8 月の実行時間が他の月に比べて 2 倍以上に多いことがわかる．また，利用されたアプリケーションは ANSYS Multiphysics と COMSOL Multiphysics に集中している．基本的には，毎月一定時間実行されている．したがって，端末室ではなく研究室などのユーザの端末上で実行が可能な配信システムは有用であると考えられる．今後は，配信する商用アプリケーションの種類を充実させていくことを検討している．また，商用アプリケーションとクラスタシステムの連携についても，利用者の状況変化に応じて実装していく予定である．

## 4　むすびに

本論文では，商用アプリケーションおよび自家製アプリケーションを含む多様なジョブ類型を考慮した小規模研究用システムの 2022 年の利用状況について報告した．本システムは，アクセラレータとして GPU を搭載したクラスタシステムと，仮想化アプリケーションの配信システムからなる．自家製アプリケーションの作成・デバッグにあたっては，実行環境と同一の環境が必要であり，かつ，開発用 GPU の排他制御が必要となるため，ログインノード上ではなく，計算ノード上の対話型ジョブとして実行するように設計した．

## 参考文献

[1] TSUBAME3.0 ハードウェアおよびソフトウェアの仕様, 2019. https://www.gsic.titech.ac.jp/sites/default/files/spec30j.pdf.

[2] T. Brown, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, Vol. 33, pp. 1877–1901, 2020.

[3] Y. Chen, A. Huang, Z. Wang, I. Antonoglou, J. Schrittwieser, D. Silver, and N. de Freitas. Bayesian optimization in AlphaGo, 2018.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT2019, Vol. 1*, pp. 4171–4186, 2019.

[5] G. M. Kurtzer, V. Sochat, and M. W. Bauer. Singularity: Scientific containers for mobility of compute. *PLOS ONE*, 12(5):1–20, 05 2017.

[6] D. Merkel. Docker: lightweight linux containers for consistent development and deployment. *Linux Journal*, 239(2):2, 2014.

[7] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with CLIP latents, 2022.

[8] 土屋. HPC クラスタシステム上で動作する仮想マシンを用いた Hadoop クラスタの構築. 情報処理学会研究報告, 第 2013–HPC141–9 巻, pp. 1–4, 2013.

[9] 土屋. IaaS 基盤としても利用できる HPC クラスタシステムの構築. 第 3 回地域間インタークラウドワークショップ, 2013.

[10] 土屋, 中村, 小林. 多様なジョブ類型を考慮した小規模研究用システムの構築と運用. 学術情報処理研究, 25, 2022.

[11] 小川, 松岡, 佐藤, 高野, 滝澤, 谷村, 三浦, 関口. 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. 情報処理学会研究報告, 第 2018-HPC-165 巻, pp. 1–7, 2018.