Vol.2021-HPC-179 No.2
2021/5/14

VCP による CPU アーキテクチャが異なる HPC システムとクラウドの連携

佐賀一繁[1]　三浦信一[2]　丹生智也[1]　竹房あつ子[1,4]　横山重俊[3,1]　合田憲人[1,4]

概要：Virtual Cloud Provider (VCP) は，複数のクラウドプロバイダや学術機関のデータセンターを仮想的な 1 つのクラウドプロバイダとして連携させ，アプリケーション実行環境を構築するソフトウェアである．このような連携は，アプリケーションワークフローのステップ毎に適した計算資源の利用を可能とし，資源の利用効率などを高める．このとき，資源確保方法の違い，アプリケーション実行方法の違い，資源アーキテクチャの違いが課題となり，特に CPU アーキテクチャとシステムソフトの違いは，ノードにおけるアプリケーション実行やクラウド間連携機能に大きな影響を与える．そこで，Arm アーキテクチャであるスーパーコンピュータ「富岳」とインテル系アーキテクチャが主のクラウドプロバイダの連携を念頭に，富岳互換アーキテクチャである FX700 にて VCP のノード管理機能の評価を実施した．その結果 VCP ベースコンテナ内のアプリコンテナの実行機能(Docker in Docker)とノード情報収集機能に課題があることが判明し，コンテナエンジンの変更などの対策を実施した．その結果，FX700 ノードにおける VCP のノード管理機能の正常動作を確認した．本報告では，連携形態，課題，対策などを示す．

キーワード：HPC システム，ハイブリッドクラウド，富岳

Federation among HPC Systems and Cloud Providers That Use Different Architecture Type CPUs

KAZUSHIGE SAGA[1]　SHIN'ICHI MIURA[2]　TOMOYA TANJO[1]
ATSUKO TAKEFUSA[1,4]　SHIGETOSHI YOKOYAMA[3,1]　KENTO AIDA[1,4]

1. はじめに

従来，大学や研究所などの学術機関では，メールや業務用ソフトウェアなど,いわゆる SaaS を中心にクラウドサービスを導入してきた．しかし近年，研究や教育用に IaaS を利用するケースも増えつつある．さらに今後は，オンプレミスの計算資源とクラウドの計算資源を有機的に連携させたハイブリッドクラウドに発展していくものと考えられる．これは両者を組み合わせることで，オンプレミス計算資源不足時におけるクラウド計算資源による補完や，アプリケーションのワークフロー毎に最適な計算資源の選択が可能になるなど計算資源選択や組み合わせの自由度が増し，また全てをオンプレミスシステムのみで構成するよりも TCO 削減を期待できるなどの理由による．

国立情報学研究所が開発した Virtual Cloud Provider (VCP) [1]は，複数の資源プロバイダを統合して 1 つの仮想的なクラウドプロバイダとして構成する機能を持つ．資源プロバイダとして，オンプレミスの VM 管理ツール，複数の商用クラウドプロバイダ，学術機関が提供するプロバイダをサポートしており，ハイブリッドクラウドの構築が可能である．ハイブリッドクラウド上にアプリケーション環境を構築するには，まずこれら資源プロバイダから計算資源やストレージ資源を確保する必要がある．しかし，資源プロバイダ毎に資源管理インターフェースが異なるため，操作が非常に煩雑である．そこで，VCP を含むハイブリッドクラウドコントローラでは，資源管理インターフェースを抽象化するのが一般的である．利用する資源プロバイダの指定はできるため，資源確保時の状況に応じて最適な資源プロバイダを選択することはできる．

VCP を含むハイブリッドクラウドコントローラでは，バッチシステムベースの計算機システムを資源プロバイダとして利用できない場合が多い．また計算資源として異種 CPU の混在に対応していないことが多い．現在でも HPC システムはバッチシステムによる管理が主流であり，また今後は Arm 系 CPU を採用するシステムの増加が予想されることから，これらは改善すべき課題であると考える．そこで，VCP において，バッチシステムベースの HPC システムのサポートと，複数の CPU アーキテクチャの計算資源が混在するハイブリッドクラウドへの対応を目的に検討を行った．

本稿では，VCP から HPC システムをハイブリッドクラウドの計算資源プロバイダとして利用するための機能要件の検討結果と，理化学研究所が運用するスーパーコンピュータ「富岳」（以下，「富岳」という）[2]と互換アーキテクチャーである FX700 を用いた富岳クラウド検証環境を利用し，その計算ノードを VCP の計算資源として動作検証し

1 国立情報学研究所
National Institute of Informatics
2 理化学研究所
RIKEN
3 群馬大学
Gunma University
4 総合研究大学院大学
Graduate University for Advanced Studies

た結果を報告する．なお，本稿では計算ノードとは HPC システムの計算ノードを指し，計算資源とは特にことわらない限り HPC システムの計算ノードもしくはクラウドの計算インスタンスを指すものとする．

以下，第 2 章でハイブリッドクラウドを構成するための課題と VCP の機能について，第 3 章で HPC システムによるハイブリッドクラウド構成例を，第 4 章で，HPC システムとその計算ノードを，各々ハイブリッドクラウドの資源プロバイダと計算資源して利用するための機能要件を，第 5 章で FX700 の計算ノードを VCP の計算ノードとして利用する実証実験を，第 6 章で関連研究を，最後にまとめと将来課題について報告する．

2. ハイブリッドクラウドと VCP

利便性の高いハイブリッドクラウドを構成するには，以下の 3 つの基本的機能課題を解決する必要がある．

- 資源プロバイダ毎に異なる資源管理インターフェースに対するサポート
- 構成する資源プロバイダの計算資源間のアプリケーション環境のポータビリティの確保
- オンプレミス資源プロバイダとクラウドプロバイダ間の安全，高速な接続

その他，オンプレミス資源とクラウド資源間のデータ共有方法などの課題もあるが，利用機関により課題が異なるため，本稿では上記基本的機能課題についてのみ検討する．

VCP は，オーバレイクラウドとオーバレイネットワークの概念の下，オンプレミスの資源プロバイダを含む複数の現実の資源プロバイダを，1 つの仮想的なクラウドプロバイダとしてまとめ，資源の統一管理を可能としている．図 1 に概念的な構成を示す．なお，現実のプロバイダを以下単にプロバイダと記す．

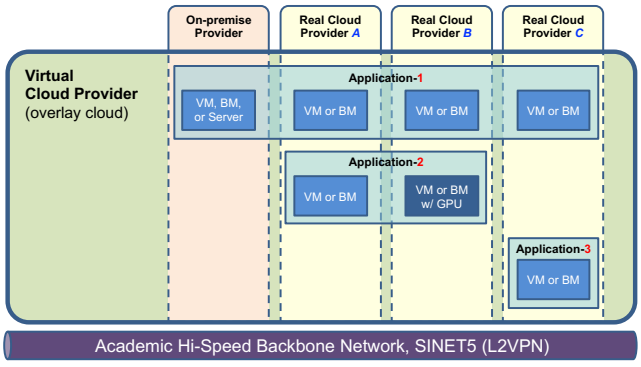

図 1　仮想クラウドプロバイダ
Figure 1　Virtual Cloud Provider

VCP によって統一管理された環境を仮想クラウド（VC）と呼ぶ．利用者が VCP を使用して VC 上の資源の確保や削除などを行うには，プロバイダの資源管理インターフェースではなく VCP のものを使用する．VCP の資源管理インターフェースは，複数のプロバイダの資源管理インターフェースが持つ基本的かつ共通な機能を，OSS である Terraform と独自開発の機能により抽象化しており，VC を構成する全てのプロバイダの資源を同じ操作方法で管理することができる．これにより，ハイブリッドクラウドの基本的機能課題の一つであるプロバイダ毎に異なる資源管理インターフェースの課題に対応することができる．なお，VCP の資源管理インターフェースでは，VC の資源を Node，Unit，UnitGroup の 3 階層で管理しており，各管理単位での資源の作成，追加，削除が可能である．

- Node: 単体の資源であり，計算資源とブロックストレージ資源の 2 種類がある．計算資源は，仮想マシン (VM) もしくはベアメタルマシン(BM) と，その上で動作する OS，VCP のベースコンテナ（後述）から構成されている．
- Unit: ひとつのプロバイダに属する同質（VM, BM，ベースコンテナなどが同じ）の Node の集合であり，複数の Node を一括して操作することが可能である．均一なノード構成を持つクラスタなどの管理に用いる．
- UnitGroup: 異なる Unit の集合．ただし，計算資源の Unit とストレージ資源の Unit は混在できないが，複数のプロバイダを跨ることができる．

この階層管理の仕組みにより，資源単体の管理だけでなく，これらをまとめた資源の一括管理が可能であり，より利便性の高い資源管理を実現している．

Node には資源タイプ名（AWS でのインスタンスタイプ名）を指定する必要があるが，資源プロバイダの違いにより同様な仕様の資源であっても資源タイプ名は異なる．そこで，VCP では資源タイプ名（フレーバと呼ぶ）も抽象化している．この抽象化された資源タイプ名と各プロバイダの資源名の対応は，VCP の構成ファイル内に記述する．また，Node が属するプロバイダの違いにより，同様な仕様の資源でも料金などの利用条件も異なる．このため，Node のプロバイダを指定できないと，コストの最適化などができない．そこで，Node の資源要件指定にはプロバイダ名を指定する仕様としている．これにより，明確にオンプレミスとクラウドの資源を意識して指定することが可能である．

VCP の計算資源構成を図 2 に示す．ベースコンテナにはシステム機能ならびに VCP のノード管理機能がインストールされており，ハードウエア構成や OS(kernel)機能に大きな違いがない限り，フレーバやプロバイダの違いに依存しない構造になっている．また，アプリケーション環境もコンテナ化することにより，異なるプロバイダの計算資源間でのポータビリティや，システム環境まで含めたアプリケーション環境の高い再現性を実現している．これは，上記ハイブリッドクラウドの基本的機能課題である，アプリ

ケーション環境のポータビリティの課題に対応することができる．

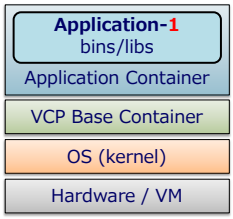

図 2　VC ノードの構成
Figure 2　VC node configuration

学術情報ネットワーク SINET5 は学術機関ならびに複数の商用クラウドプロバイダに接続しており，利用機関とクラウドプロバイダ間とをインターネットを経由せずに L2VPN で接続する SINET クラウド接続サービスを提供している[3]．また，VCP をサービスとして提供する学認クラウドオンデマンド構築サービス[4]は，利用機関に提供する VCP のコントローラ（VC コントローラ）を SINET に直接接続している．このため，同サービスを利用することで，利用機関，資源プロバイダ，VC コントローラの間をインターネットを経由しない L2VPN で接続することができ，安全，高速，低遅延の接続を実現できる．上記ハイブリッドクラウドの基本的機能課題である，オンプレミス資源プロバイダとクラウドプロバイダ間の安全，高速な接続に対応することができる．

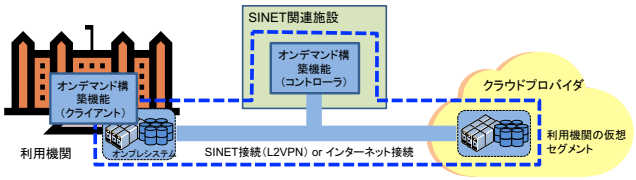

図 3　オンデマンド構築サービスのネットワーク接続
Figure 3　Network connection of on-demand configuration service

以上のように，VCP はハイブリッドクラウドを構築するための基本的機能を具備している．

3. HPC システムによるハイブリッドクラウド

HPC システムによるハイブリッドクラウドの構成には様々な形態が考えられる．代表的な 2 つの形態と VCP からの利用方法を説明する．

3.1 HPC システムのクラウド資源利用

HPC システムを主とする利用形態であり，クラウドバースティングと呼ばれるクラウドによる資源補完である．利用者は HPC システムのジョブ投入インターフェース経由でクラウド資源を利用する．

HPC システムは，年度初めは利用率が低くジョブの実行待ちはほとんど発生しないが，論文提出時期などでは利用率が非常に高く長時間ジョブ実行を待たされる傾向にある．これは，研究の進捗に影響を与える大きな問題であるが，オンプレミスの HPC システムだけでは解決が難しい．そこで高負荷になった時，HPC システムの計算ノードと同じソフトウェア構成を持つ計算資源をクラウドプロバイダ上に確保し，HPC システムのバッチシステムの計算ノードとして追加することで，負荷をクラウド計算資源に分散させることで問題を解決する．研究の進展などで急遽既存システムにない新たなタイプの計算資源が必要になったときにも，同じ方法で解決することができる．

最近のバッチシステムには，クラウドに負荷分散させる機能を持つものもある[5][6]．VCP で実現する場合の構成と動作を図 4 に示す．利用者は，システムの混雑具合やジョブ実行の急ぎ具合によって，利用するキューを選択してジョブを投入する．VCP がクラウド資源を確保する契機を知る方法には 2 種類あり，cron などで定期的にクラウド資源キューを監視する方法と，バッチシステムの外部ルーチン呼出機能を利用する方法が考えらえる．確保されたクラウド資源はバッチスシステムの設定により自動的に新たな資源として追加される．バッチシステムは，追加されたクラウド資源でジョブを実行する．削除についても同様の流れであり，クラウド資源キューに待機中のジョブが無くなったことを契機に VCP で資源の削除を行う．

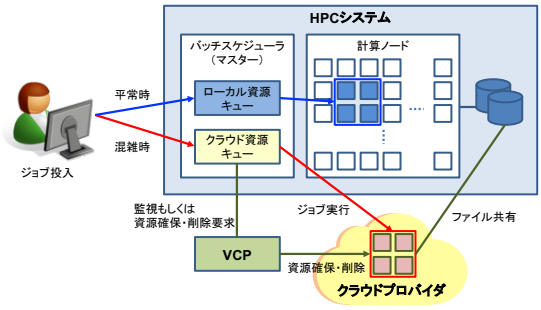

図 4　クラウド資源による資源補完
Figure 4　Resource complementation with cloud resources

3.2 資源プロバイダとしての HPC システム

HPC システムを資源プロバイダのひとつとして利用する形態である．利用者は，VCP の資源管理インターフェース経由で HPC システムの計算ノードを計算資源として利用する．

使い方は様々であるが，例えば RISM-FMO など HPC システムと大規模メモリシステムを併用する連成計算である．図 5 に構成，実行例を示す．連成方法も様々であるが，ここでは，より密な連成のために両者は直接通信を行うものとする．また，前後の処理も HPC システム以外の計算資源で実行するものとする．ワークフローシステムは，ワーク

フローの各ステップでの実行要件を満たす資源の確保を VCP に依頼し，確保した資源にアプリケーションステップを投入する．各ステップの実行が終了したら，ワークフローシステムは，VCP に資源の削除依頼を行う．

この形態では，アプリケーションワークフロー全体の実行時間の短縮やコストの低減を期待でき，また従来やむなく HPC システムで実行していた前後処理をクラウド計算資源に移行することで，前後処理を行なっていた分の計算ノードで他のアプリケーションを実行できるため．HPC システムの利用効率向上も期待できる．

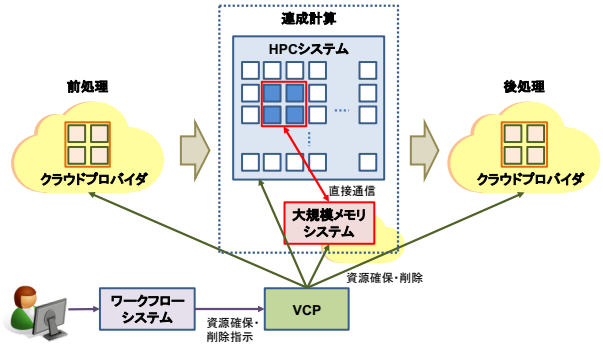

図 5　連成計算ワークフローの資源管理
Figure 5　Resource management of coupled simulation workflow

4. 機能要件

2 章で説明したように，VCP はハイブリッドクラウドを構築するための基本的機能を持っている．しかし，VCP がオンプレミスのプロバイダとしてサポートしているのは，単体サーバ用 VCP エージェントと VMware などの VM/BM 管理システムのみであり，HPC システムで多用されているバッチシステムはサポートしていない．これは，VCP の計算資源管理では，計算資源との直接通信と管理者権限を必要とするのに対し，従来の HPC システムの計算ノードは外部との直接的なネットワーク接続を持たず，また管理者権限を利用できないことが一般的なためである．

3.1 節で示した資源補完の利用例では，VCP が資源プロバイダとして利用しているのはクラウドプロバイダのみであり HPC システムは利用していない．また，HPC システムの計算資源の補完としてクラウド計算資源を利用するため，両者の CPU アーキテクチャは同じである．このため，この利用方法は既存の HPC システムと VCP で実現できる．既に研究に利用されている例があり[7]，また国立情報学研究所が開発したスクリプトも公開されている[8]．

一方，3.2 節に示した HPC システムの計算ノードをハイブリッドクラウドの計算資源として利用するには，HPC システム，VCP 双方に以下の課題があると考える．

(1) **HPC システムに求められる課題**
- (a) 計算ノードをハイブリッドクラウドの計算資源として確保，削除するためのインターフェース
- (b) HPC システム外部から計算ノードへの通信
- (c) 計算ノードでのコンテナ実行
- (d) HPC システムとクラウドなど他の計算資源プロバイダ間のデータ共有

(2) **VCP に求められる課題**
- (e) 計算資源の CPU アーキテクチャ別ベースコンテナイメージ
- (f) 計算資源アーキテクチャの違いによるコンテナの選択制御

(3) **共通課題**
- (g) HPC システムの利用者が，複数の機関に所属する場合，資源プロバイダ間を結ぶネットワークの利用機関別通信路制御

5. 富岳クラウド検証環境での VCP 動作検証

理化学研究所は，「富岳」のクラウド的な利用方法を検討しており，システム外部と計算ノードのネットワーク接続，コンテナのサポートも視野に入れている．また SINET クラウド接続サービスによる商用クラウドプロバイダとの直接接続も整備中である．また，国立情報学研究所と理化学研究所は，インタークラウド環境の研究として，VCP から「富岳」を利用する方法を検討している．

VCP で「富岳」の計算ノードをハイブリッドクラウドの計算資源として利用するには，4 章で説明した機能要件を満たす必要がある．今回は最も基本的な機能である，「富岳」の計算ノードが VCP の計算資源として動作するかの検証を実施した．これが動作しないと，VCP から「富岳」を利用することができない．動作には，少なくとも上記機能要件の(b), (c), (e)を満たす必要がある．

今回の検証では，「富岳」本体ではなく，理化学研究所の富岳クラウド検証環境として整備された富士通製 FX700 を使用した（以下，FX700 と記す）．「富岳」と FX700 の計算ノードの主な違いはノード間インターコネクト（富岳: Tofu D，FX700: InfiniBand），OS（富岳: RedHat 8，FX700: CentOS 8），バッチシステム（富岳: 富士通製，FX700: SLURM）である．ノード間インターコネクトの違いは，両者とも IP on インターコネクト機能を持っているため，VCP が使用する TCP/UDP レベルの通信では問題とならない．両者の OS は互換性があり機能的な差異はない．また，今回は計算ノード単体の検討であるため，バッチシステムの違いの影響はないと考えられる．なお，両者とも CPU は A64FX である．

5.1 検証環境

今回は，FX700 上に VCP 動作検証専用の 3 ノードを用

意し，上記要件(b), (c), (e)を満たす以下の環境を構築した．

(1) **HPC システム外部から計算ノードへの通信**

機能要件(b)HPC システム外部から計算ノードへの通信に対応する．「富岳」ならびに富岳クラウド検証環境は，その計算ノードと外部機関に設置された計算機を直接通信可能にする「富岳ダイレクト接続サービス」の提供を計画している．本サービスは国立情報学研究所が提供する SINET の L2VPN などを用いることを検討しているが，現時点では構築中である．そこで，FX700 のセグメントに VPN サーバを設け，VCP と検証用計算ノードの間のみ通信可能として検証を行なった．図 6 に今回の検証システムの全体構成を示す．

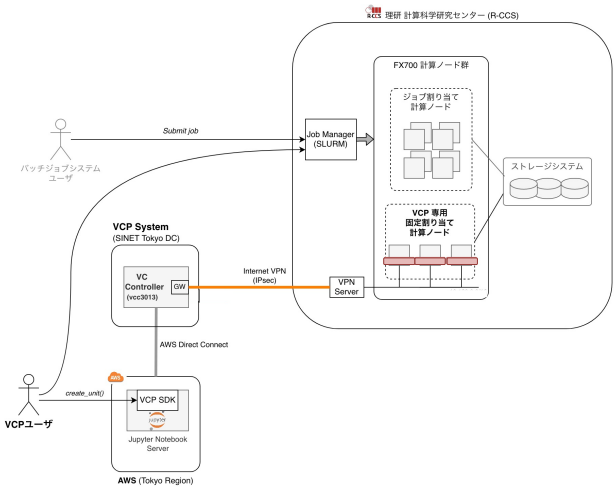

図 6　検証システムの構成

Figure 6　Verification system configuration

(2) **計算ノードでのコンテナ実行**

機能要件(c)計算ノードでのコンテナ実行に対応する．VCP は docker コンテナを採用し，図 2 に示すようにベースコンテナとアプリケーションコンテナを docker in docker で階層的に動作させる．しかし，「富岳」ならびに FX700 の OS である RedHat 8 ならびに CentOS 8 では，docker を排除しているため，ベースコンテナの起動には代替である podman を利用することにした．

(3) **Arm 系 CPU 用 docker コンテナイメージ**

機能要件(e)計算資源の CPU アーキテクチャ別ベースコンテナイメージに対応する．VCP は x86 系 CPU でのみ動作確認しており，「富岳」ならびに FX700 が採用している Arm 系 CPU では動作実績がない．そこで，Arm 系 CPU 用のベースコンテナならびにアプリケーションコンテナを作成した．なお，ベースコンテナ内でノード情報を収集する cAdvisor は Arm 系 CPU をサポートしていないため，同様な機能を持ち Arm 系 CPU をサポートしている Telegraf に変更した．また，アプリケーションコンテナとしては，初期動作確認用に stress コマンドを，アプリケーション動作確認用に TensorFlow を実装したものを用意した．

5.2 検証項目

以下項目の検証を実施した．

- ベースコンテナの起動確認
- stress コマンドを実装したアプリケーションコンテナの動作確認
- Telegraf によるノード情報収集確認
- TensorFlow を実装したアプリケーションコンテナ上での手書き数字認識 MNIST の学習動作確認

5.3 動作検証

(1) **ベースコンテナの起動確認**

ベースコンテナは docker コンテナイメージで作成されているが，podman での正常起動を確認した．

(2) **stress コマンドのアプリケーションコンテナ動作確認**

VCP ではアプリケーションコンテナを docker in docker 方式で起動するが，上記のようにベースコンテナの起動に docker が利用できないため，docker in podman 形式でアプリケーションコンテナの起動を試みた．結果として，正常にアプリケーションコンテナが起動し，stress コマンドが正常に実行されることを確認した．

(3) **Telegraf によるノード情報収集確認**

Telegraf で収集したノードの負荷情報を，データ可視化ツール grafana で表示し，上記 stress コマンド実行時の負荷が正常に収集・表示されることを確認した．表示画面のスクリーンショットを図 7 に示す．

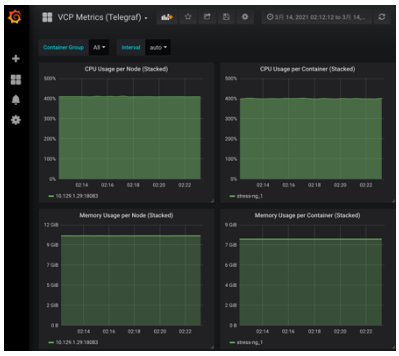

図 7　stress コマンド実行時の負荷表示

Figure 7　load display of stress command

(4) **TensorFlow アプリケーションコンテナの動作確認**

実際のアプリケーションの動作確認として，TensorFlow を実装したアプリケーションコンテナ上で MNIST の学習を行い，正常に学習されることを確認した．実行時のスクリーンショットを図 8 に示す．なお，この画面は学認クラウドオンデマンド構築サービスの NII 拡張 JupyterNotebook から実行した結果である．テストセットの認識率は，x86 系 CPU で実行した時と同等であり，正常に学習できている

と考えられる．

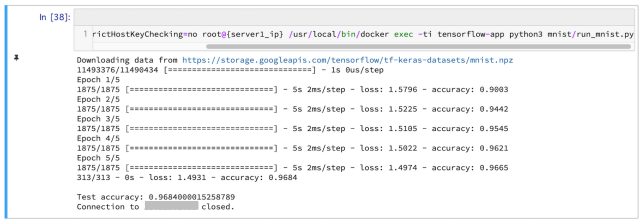

図 8　MNIST の学習結果
Figure 8　MNIST learning results

5.4 結論

今回の動作検証では，環境構築時に抽出した以下の課題以外は抽出されなかった．また，これらの課題に対して上記の対応をした結果，正常動作を確認できたので，対応策についても問題ないことが確認できた．このため，「富岳」の計算ノードは VCP の計算資源としての求められる基本機能の実行に問題ないことが確認できた．

- RedHat 8 ならびに CentOS 8 以降では docker が動作しない
- cAdvisor は Arm 系 CPU に対応していない

6. 関連研究

Steve Crago [9]らは，OpenStack で異種計算資源をサポートするための改造方法を示した．異種計算資源として，一般によくある GPU 有無だけでなく，異なる CPU アーキテクチャの計算資源もサポートしている．実装にあたり，オリジナルの OpenStack の API との互換性を重視するため，インスタンス名にアーキテクチャ情報などの拡張資源要件を追加している．例えば，彼らの実験環境で TILEPro64 アーキテクチャのインスタンスを使用する場合，資源のインスタンス名に cpuarch = TILEPro64 を追記する．CPU アーキテクチャの他に，使用するアクセラレータ（GPU など）の数，アクセラレータのアーキテクチャ（Fermi など），CPU に必要な機能（SSE4.2 など）が指定可能である．OpenStack は，配下の資源についてこれらの拡張要件記述を表示しないため，利用者は拡張要件を持つ拡張要件の存在を判別できない．そこで，彼らは libvirt を改造して拡張要件を表示可能としている．なお，計算資源としてはサーバを想定しており，バッチシステムで管理された資源はサポートされていない．

7. まとめ

本稿では，VCP がオンプレミスの単体サーバや VM 管理ツールとクラウドプロバイダとでハイブリッドクラウドを構築するための基本的機能を有していることを示し，また VCP を含むハイブリッドクラウドコントローラが，バッチシステムベースの HPC システムをサポートし，複数の CPU アーキテクチャの計算資源を混在可能とする機能要件を示した．これを VCP で実現するには，まず HPC システムの計算ノードを VCP から制御できるかを確認する必要があり，今回は富岳互換アーキテクチャである FX700 の計算ノードを VCP で制御する動作検証を実施し，事前準備で抽出した課題以外に問題ない事を確認した．また，事前準備で抽出した課題は，その対策が有効であることを確認した．

VCP で HPC システムを含むハイブリッドクラウドを構築するには，まだ多くの機能検討と開発が必要である．ハイブリッドクラウドには多くの利点があるため，今後も機能検討と開発を継続する．

謝辞　本研究にご協力いただいた株式会社アスケイドの那須野淳氏，羽鳥文子氏，増山隆氏に深く感謝いたします．また，本研究の一部は，JST-CREST,JPMJCR1501 の支援を受けたものである．

参考文献

[1] 竹房あつ子，横山重俊，政谷好伸，丹生智也，佐賀一繁，長久勝，合田憲人，SINET を活用したインタークラウド環境構築システムの開発，信学技報 117(153), 7-12, 2017.
“「富岳」について”，https://www.r-ccs.riken.jp/fugaku/（参照 2021-4-12）.
[2] “「富岳」について”，https://www.r-ccs.riken.jp/fugaku/（参照 2021-4-12）.
[3] “クラウド接続”，https://www.sinet.ad.jp/connect_service/service/cloud_connection（参照 2021-4-12）.
[4] “学認クラウドオンデマンド構築サービス”，https://cloud.gakunin.jp/ocs（参照 2021-4-12）.
[5] “IBM Spectrum LSF”，https://www.ibm.com/jp-ja/products/hpc-workload-management（参照 2021-4-12）.
[6] “slurm“，https://slurm.schedmd.com/elastic_computing.html （参照 2021-4-12）.
[7] 二階堂愛，ライフサイエンス研究の生産性を向上させるためのオンデマンドクラウド，学術情報基盤オープンフォーラム 2018
[8] “学認クラウドオンデマンド構築サービスのアプリケーションテンプレート”，https://github.com/nii-gakunin-cloud/ocs-templates （参照 2021-4-12）.
[9] Steve Crago, Kyle Dunn, Patrick Eads, Lorin Hochstein, Dong-In Kang, Mikyung Kang, Devendra Modium, Karandeep Singh, Jinwoo Suh, John Paul Walters, Heterogeneous cloud computing, IEEE International Conference on Cluster Computing, 2011