Vol.2014-HPC-145 No.4
2014/7/28

高性能かつスケールアウト可能なHPCクラウド AIST Super Green Cloud

高野 了成[1,a)]　谷村 勇輔[1]　竹房 あつ子[1]　広渕 崇宏[1]　田中 良夫[1]

概要：本論文では、「Build Once, Run Anywhere」という設計思想の基、プライベートクラウドと商用クラウドなど、異なるクラウドサービス間で可搬性を持つ仮想クラスタを提供する HPC クラウド技術を提案する。本技術を実装したプライベートクラウド AIST Super Green Cloud (ASGC) は、約 70 TFLOPS の計算性能を有し、2014 年 7 月から運用を開始する。ASGC は、ユーザに物理マシンの性能に匹敵する性能を有する仮想クラスタを提供する。仮想クラスタは負荷に応じて動的に拡大、縮小できる。さらに、一度構築した仮想クラスタを、外部のクラウドサービス、具体的には Amazon EC2 へのプロビジョニングも可能である。したがって、自前のプライベートクラウドでは実行できない大規模な計算も、EC2 を利用することで可能になる。本論文では、予備評価により、提案する HPC クラウドが実現可能であることを明らかにする。

1. はじめに

米国 XSEDE [1]、欧州 PRACE [2]、国内の HPCI [3] など、高速ネットワークで接続された複数のスパコンから構成される高性能計算プラットフォームが形成されている。ユーザはシングルアカウントで各スパコンを利用したり、スパコンで計算したデータをユーザ間で共有することができる。我々は、このような基盤上で、どのスパコンでも同じソフトウェア環境を提供し、アプリケーション実行環境を一度作ればどこでも動く利便性と、高い性能を備え、さらに商用クラウドを利用したスケールアウトも可能とする高性能計算プラットフォーム基盤の実現を目指している。

この目的を実現するためには、アプリケーション実行環境を仮想化することで可搬性を高めることが有効であり、我々は、HPC 用途においてもクラウドコンピューティング技術を活用する方針を採った。ユーザには、その利点である利便性やスケールアウト性を提供可能と期待できる。一方、運用者側の視点から見ると、HPC 環境は Web サーバ等の他の計算機資源とは独立した管理が必要である。HPC 計算資源のクラウド化により、HPC を含む計算機環境の統合的管理が実現できれば、運用コストを削減する効果が期待できる。

MPI や OpenMP などで記述された HPC アプリケーションをそのままクラウド環境で実行できる高性能計算プラットフォームを「HPC クラウド」と呼ぶ。従来の HPC 環境から HPC クラウドへの移行を促進するには、ユーザが各々の要求に合ったクラスタ環境を、クラウド上に容易に構築できること、さらに、その性能が物理マシン環境と遜色ないことが求められる。これまで我々は、前者の課題について、国内外の研究機関と共同で PRAGMA Cloud [4] の研究開発を進めている。これは、Gfarm ファイルシステム上に配置した VM イメージを用いて、UCSD Rocks や OpenNebula などのクラウド上に仮想クラスタを構築するシステムである。また、後者の課題について、HPC アプリケーション実行時における仮想化の影響について性能評価を行ってきた [5]。さらに、仮想化のオーバヘッドを大幅に削減することによる、既存の HPC ユーザの HPC クラウドへの移行の促進を狙い、クラウドミドルウェアから PCI パススルーや SR-IOV を利用するための機能拡張 [6] を実現してきた。

上記で示した研究開発を通じて、我々は HPC クラウドが実用的な性能を達成可能で、かつ組織内外の計算資源との連携も容易に可能であるという確証を得た。本論文では、この知見を活かし、「Build Once, Run Anywhere」という設計思想を掲げた、可搬性の高い HPC クラウドを提案する。さらに、提案する HPC クラウドの具現化を目指し、プライベート HPC クラウド AIST Super Green Cloud（以下、ASGC と記す）を導入し、2014 年 7 月に運用を開始した。ASGC は、ユーザに物理マシンの性能に匹敵する性能

[1] 独立行政法人 産業技術総合研究所 情報技術研究部門
Information Technology Research Institute, National Institute of Advanced Industrial Science and Technology (AIST)
[a)] takano-ryousei@aist.go.jp

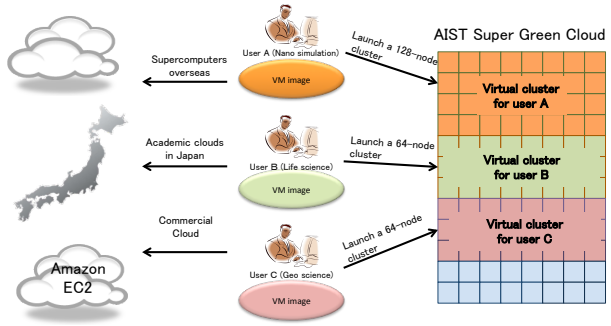

図 1 AIST HPC クラウドのビジョン「Build Once, Run Anywhere」

を提供する。さらに、一度構築した仮想クラスタは、外部のクラウドサービス、具体的には Amazon EC2 上での動作も可能であり、必要に応じて、ASGC で実行できない大規模な計算の実行も容易になる。我々の知る限り、プライベートクラウドと商用クラウドに対して可搬性を持つ仮想クラスタを提供する HPC クラウドサービスは他にない。

本論文の構成は以下のとおりである。2 節で ASGC の設計方針について述べる。その設計と実装を 3 節で、予備評価の結果を 4 節で示す。最後に 5 節でまとめを行い、今後の予定についても言及する。

2. 設計方針

2.1 HPC クラウド

我々が提案する HPC クラウドは、「Build Once, Run Anywhere」というビジョンを掲げている。これはユーザが一度構築した計算環境をプライベートクラウドだけではなく、そのまま外部のクラウドサービスにも可搬できることを意味している。具体的には、図 1 に示すように、ユーザは用途に応じた VM イメージ（以下、テンプレートと記す）を基に、プライベートクラウドや、パブリッククラウドを含む外部クラウドサービスに仮想マシンをプロビジョニングし、それらを単一のクラスタ計算機として利用できることを設計目標とした。これを仮想クラスタと呼ぶ。

なお、本 HPC クラウドが目指すのは、主にインタークラウドとして研究されている、地理的に離れた複数のクラウドに跨がった単一の計算環境を構築するものとは異なる。あくまで仮想クラスタは単一のクラウドに閉じた環境にプロビジョンされ、ユーザに提供される。また、クラウド間で仮想クラスタをライブマイグレーションさせる仕組みではなく、実行コンテキストは引き継がない。例えば、MPI プログラムでは実行時に動的にプロセス数を変えることは一般的ではない。したがって、大きなサイズの仮想クラスタを確保して、改めて MPI プログラムを実行することは不自然ではない。

2.2 仮想クラスタ

仮想クラスタとは、仮想マシンの集合であり、それらはプライベート IP ネットワークに接続されており、VLAN 等で各クラスタは隔離されている。さらに、プロビジョニングされるクラウドの差異を仮想化で極力吸収しつつ、通信デバイスに関しては、利用可能なデバイスから最良の通信性能を有するものを選択できることを考える。これは CPU の仮想化対応により CPU やメモリの仮想化オーバヘッドはほぼ問題にはならなくなっている一方で、I/O デバイスはいまだ仮想化オーバヘッドが無視できないからである。

I/O デバイスの仮想化オーバヘッドを削減する技術として、仮想マシンモニタをバイパスして、ゲスト OS からデバイスを直接操作できる PCI パススルーや SR-IOV などが存在する。具体的には、InfiniBand ネットワークを有するプライベートクラウドでは、PCI パススルーや SR-IOV を利用することで、ゲスト OS から InfiniBand デバイスを直接アクセスできるようにする。一方、仮想化デバイスしか提供されないパブリッククラウドでは、仮想化デバイスを用いる。

3. AIST Super Green Cloud (ASGC)

3.1 概要

我々が提案するスケールアウト型 HPC クラウドを AIST Super Green Cloud (ASGC) 上に実現した。

ASGC は、従来の共用高性能計算機が対象としていた HPC 用途に限定せず、大規模データ処理や Web サービス等も対象としたプライベートクラウドサービスである。HPC 以外の具体的な用途としては、現在産総研内でサービスしている GEO Grid [7] や、Songle [8]、Songrium [9] などの音楽視聴支援サービスなどが挙げられる。ただし、本論文では、HPC クラスタとしての利用に焦点を絞って説明する。以降、本節では、ASGC のハードウェア環境およびソフトウェア環境について述べる。

3.2 ハードウェア環境

ASGC は、Cray 社製ブレードサーバ H2312 が 155 ノードから構成される PC クラスタであり、論理ピーク性能は 69.44 TFLOPS である。各ノードは、CPU として 10 コアの Intel Xeon E5-2680v2 を 2 基、メモリ 128 GB、SSD 600 GB を搭載し、InfiniBand FDR および 10 ギガビットイーサネットで接続されている。主に InfiniBand ネットワークは MPI、イーサネットはリモートアクセスやストレージアクセスに用いられ、共にフルバイセクションバンド幅を有す。また、ASGC は共用ストレージを提供せず、ストレージは各ノードに搭載された SSD のみである。そのため必要に応じて、ユーザ所有のストレージを 10 ギガビットイーサネット経由で接続することで対応可能にして

表 1　ASGC の諸元

Compute Node	
CPU	10-core Intel Xeon E5-2680 v2/2.8GHz x2
Chipset	Intel C600
Memory	128 GB DDR3-1866
InfiniBand	Mellanox ConnectX-3
10 GbE	Intel X520-AD2
Disk	Intel SSD DC S3500 600 GB
Switch	
Infiniband	Mellanox SX6025
10 GbE	Extreme Networks BlackDiamond X8

いる。表 1 にハードウェア仕様をまとめる。

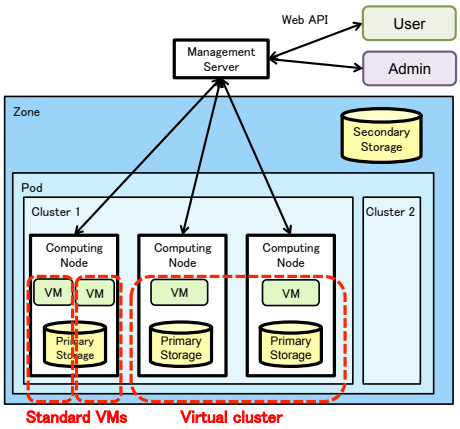

図 2　ASGC における CloudStack の構成

3.3　クラウドミドルウェア

IaaS (Infrastructure as a Service) サービスを提供するクラウドミドルウェアや商用サービスはいくつも存在するが、仮想クラスタを提供するものはない。また、Cycle computing や MIT StarCluster など、Amazon EC2 上で仮想クラスタを構築するソフトウェアは存在するが、クラウド間の可搬性は考慮されていない。

我々は、ASGC のクラウドミドルウェアとして Apache CloudStack [10] を選択した。CloudStack は、OpenStack と並ぶオープンソース・クラウドミドルウェアであり、データセンタ事業者によるクラウドサービス、研究機関によるアカデミッククラウドにおける国内外の利用実績も多い。しかし、CloudStack を HPC クラウドの基盤ソフトウェアとして利用するには、仮想クラスタの提供、および、PCI パススルーや SR-IOV による InfiniBand デバイスの対応が欠けている。これらの機能に関しては、我々は独自に開発を行った。さらに、モニタリングや課金集計等でも、CloudStack の提供する機能だけでは不足する。これらへの対応に関しても、下記で簡単に述べる。

3.3.1　Apache CloudStack

ASGC における CloudStack の全体構成を**図 2** に示す。管理サーバは、ユーザインタフェースを提供し、仮想マシンインスタンス（以下、ユーザ VM と記す）のプロビジョニング、資源管理を行う。ユーザ VM は、計算ノード上で動作する。プライマリストレージは、ユーザ VM 用のストレージ領域、セカンダリストレージは、ユーザ VM のテンプレートイメージ、ISO イメージ、スナップショット用のストレージ領域である。ASGC では、プライマリストレージとして計算ノードのローカル SSD を用い、セカンダリストレージとして RADOS ストレージクラスタを利用する。詳細は 3.4 節にて後述するが、プロビジョニング時には、セカンダリストレージ上のテンプレートが、プライマリストレージに転送され、ユーザの公開鍵の埋め込みなどのインスタンス化が行われ、ユーザ VM が起動する。

ユーザ VM には、仮想 CPU を 1 つ有し、物理マシン上にコンソリデーション可能なスタンダード VM と、仮想マシンを占有する HPC VM の 2 種類が存在する。HPC VM は InfiniBand デバイスを PCI パススルー経由で利用できる。仮想クラスタは複数の HPC VM から構成される。

ユーザの要求によって作成されるユーザ VM の他に、ルータ、DHCP、ファイアウォールなどを提供する仮想ルータ、ユーザに仮想マシンのコンソールアクセスを提供するコンソールプロキシ VM、セカンダリストレージを提供するセカンダリストレージ VM と呼ばれるシステム VM が存在する。システム VM は、必要に応じて作成されるので、その数は動的に増減する。

ASGC の全物理マシンは、単一の L2 ネットワーク（CloudStack ではクラスタと呼ばれる）に所属している。また、仮想クラスタ内の仮想マシンは、他のクラスタと隔離されたゲストネットワークで接続されている。CloudStack では、ネットワーク設定として、基本ネットワークと拡張ネットワークが選択できるが、仮想クラスタ間の隔離を VLAN を用いて制御するために、拡張ネットワークを採用した。仮想マシンは、前述の仮想ルータを経由して、外部のネットワークと接続される。標準のファイアウォール設定では、仮想マシンは産総研のイントラネットワークからのアクセスのみを許すが、Web サーバ等、外部との通信が必要な場合は、ユーザが適時ファイアウォールの設定を変更することで対応できる。

我々は、CloudStack 4.3.0 をベースに、PCI パススルーおよび SR-IOV に対応するための機能拡張を施している他、HPC VM のように、CPU コア数が多い仮想マシンを作成する場合の CPU コアトポロジに関する不具合など、軽微なバグに対する修正を行ったり、運用において緊急度の高いパッチを開発バージョンからバックポートするなど、安定運用に向けた取り組みを行っている。また、産総研所内の事情に合わせた課金計算・レポートシステムも実装した。これは CloudStack の利用状況測定サーバ（Usage server）から得られる情報を利用し、ユーザや予算毎に課金情報を取りまとめるものである。

3.3.2 仮想クラスタ構築ツール

仮想クラスタは、フロントエンドノードと計算ノードから構成される Beowulf 型クラスタ構成を採用した。フロントエンドノードは、NFS サーバ、NIS、バッチシステムを提供しており、ユーザはフロントエンドノードに SSH でログインし、ジョブを投入できる。また、ユーザは、仮想クラスタに対して、動的に計算ノードを追加、削除することを可能とする。仮想クラスタのテンプレートは、フロントエンドノードと計算ノード用の 2 組で取り扱う。ユーザは自身で仮想クラスタ環境をカスタマイズした後、それぞれの仮想マシンをテンプレート化することで、望むときにそのテンプレートから仮想クラスタを再構築できるようになる。ASGC が標準で提供する HPC VM テンプレートでは、バッチシステムとして TORQUE、MPI ライブラリとして Intel MPI および Open MPI がインストールされている。

フロントエンドノードと計算ノードは、ゴシッププロトコルベースのメッセージングシステム Serf [11] を用いてメンバ管理されており、仮想クラスタの構成が変更されると、ホストファイルを更新したり、バッチシステムに対する計算ノードの登録、削除といった操作が自動的に実行される。テンプレートから仮想マシンが初めて起動される場合、OS 初期化スクリプト経由で Cloud-init が実行され、ホスト名や SSH 鍵、Serf などの設定が行われる。Serf の動作は次のとおりである。計算ノードが仮想クラスタに参加する際、計算ノードからフロントエンドノードへメッセージが配信される。そして、フロントエンドノード経由で全計算ノードに情報が伝播される。計算ノードはフロントエンドノードのアドレスを知る。仮想クラスタ構築ツールでは、まず、フロントエンドノードをプロビジョニングし、その IP アドレスを取得する。続いて、計算ノードをプロビジョニングする際、先に得た IP アドレスを Serf の設定として埋め込む。

ユーザによる仮想クラスタの構築は、CloudStack 標準の Web UI からではなく、ログインノードにおいてコマンドラインから sgc-tools コマンドを用いて実行する。sgc-tools は、仮想マシン・仮想クラスタを作成、操作するためのユーティリティスクリプト群である。表 2 にその一覧を示す。主なツールは単一の仮想マシンを操作する `sgc-vm` コマンドと、仮想クラスタを操作する `sgc-cluster` コマンドの 2 つであり、残りはその補助コマンドである。

仮想クラスタを作成する場合は、`sgc-cluster create mycluster.txt` のように構成情報を定義ファイルとして渡して実行する。次に定義ファイルの例を示す。1〜4 行目がフロントエンドノード固有情報、6〜10 行目が計算ノード固有情報、12〜16 行目が両者に共通の情報である。`sgc-cluster` コマンドの実行結果、フロントエンドノード 1 台と計算ノード 16 台から構成される仮想クラスタ myCluster が生成される。なお、HPC コンピュートオファリングは、PCI パススルーを利用するために必要なオファリングであり、CloudStack は利用可能な計算ノードから、InfiniBand デバイスが利用可能なノードを探し、仮想マシンをプロビジョニングする。InfiniBand デバイスは各計算ノードに 1 基なので、その計算ノードには 1 つの仮想マシンしかプロビジョニングされず、占有が可能となる。

```
1  [frontend]
2  template = cluster_frontend-20140701
3  compute_offering = HPC
4  disk_offering = Large
5
6  [cmpnode]
7  number = 16
8  template = cluster_compute-20140701
9  compute_offering = HPC
10 disk_offering = Large
11
12 [common]
13 name = myCluster
14 zone = zone01
15 network = myNetwork
16 sshkey = myKey
```

16 行目で指定した SSH 公開鍵は、プロビジョニング時に Cloud-init によってフロントエンドノードの仮想マシンイメージに書き込まれる。その結果、ユーザはデフォルトユーザとしてログイン可能になる。仮想クラスタの起動は `sgc-cluster start` コマンドで、停止は `sgc-cluster stop` コマンドを用いる。これらは物理クラスタにおける電源の入断に対応し、プロビジョニングされた仮想マシンイメージはプライマリストレージに残ったままである。さらに、`sgc-cluster destroy` コマンドで仮想クラスタを破棄する。破棄されると、仮想マシンイメージも完全に解放される。

3.3.3 Zabbix モニタリングシステム

Zabbix[12] は、サーバ、ネットワーク機器、アプリケーションなど、計算機システムを構成するコンポーネントを一元的にモニタリングできる統合監視ソフトウェアである。標準状態で多くの機器に対応している上、監視対象機器の種類によらず、Web ブラウザから統一された手順で監視設定を操作できる。ASGC では、物理マシンおよび仮想マシンの資源利用状況をモニタリングするために Zabbix を用いており、対象とする資源は、CPU 負荷、メモリ使用量、ネットワークインタフェースのスループット、およびディスク使用量である。

物理マシンのモニタリングでは、Zabbix エージェントを用いて、物理マシンの統計情報を取得する。モニタリング情報の授受においては、モニタリング情報を Zabbix エージェント側から送信する構成とし、Zabbix サーバの負荷低減を図っている。一方、仮想マシンのモニタリングでは、すべての仮想マシンで Zabbix エージェントが動作するこ

表 2　仮想クラスタ構築ツール

コマンド名	サブコマンド名	機能説明
sgc-init	-	sgc-tools に必要な設定ファイルの初期化
sgc-vm	list|create|destroy|start|stop|show	仮想マシンの作成、一覧表示、破棄などの操作
sgc-cluster	list|create|destroy|start|stop|show|add-node|delete-node	仮想クラスタの作成、起動、停止、破棄などの操作
sgc-network	list|create|acquire	ゲストネットワークの作成、一覧表示などの操作
sgc-sshkey	list|register|delete	公開鍵の登録、一覧表示などの操作

とを強制できないので、ホスト OS の cgroups および KVM が提供するインタフェースを用いて、間接的に測定を行う。ここで、個々の仮想マシンを、ハイパーバイザ経由で監視するのではなく、直接 Zabbix の監視対象として登録することで、仮想マシンが別の物理マシンに移動しても、特段の設定変更なしに、継続してモニタリングが可能になる。

3.4 RADOS ストレージクラスタ

仮想クラスタで使われる仮想ディスクイメージを管理するため、ASGC では次のようなストレージ環境を構成する。各計算ノードで動作する仮想マシンのディスクを格納するプライマリストレージとしては計算ノードに搭載された SSD を利用し、OS 用のルートボリュームとユーザが自由に使えるデータボリュームを提供する。いずれのボリュームも仮想マシンの作成時に用意され、仮想マシンの破棄後には削除される一時ストレージととしてのみ利用可能である。一方、仮想マシンのテンプレートや仮想ディスクのスナップショットを格納するセカンダリストレージとしては、データ転送用の 10 ギガビットイーサネットを介して計算ノードからアクセス可能なオブジェクトストレージを提供する。これは RADOS (Reliable Autonomic Distributed Object Store) [13], [14] と RADOS Gateway (RGW) [15] を用いて構築されたオブジェクトストレージである。RADOS は 10 台のストレージサーバからなり、合計で 160 TB の物理容量を持つが、3 コピーのレプリカ設定を採用しており、サービス向けの実効容量は約 48 TB である。RGW は RADOS のフロントエンドとして、Amazon S3 の互換インタフェースを介して RADOS へのアクセスを可能にするサービスである。現在は 2 台の RGW サーバにより Active-Stanby の構成をとっているが、負荷次第で Active-Active への移行やサーバ数の増強を検討する予定である。

現在の CloudStack において、S3 のセカンダリストレージを用いる場合、NFS によるステージング・キャッシュが必須となる。つまり、RADOS ストレージクラスタとステージング・キャッシュと間では、RGW を介して、S3 API でのファイルの授受が行われる。セカンダリストレージ VM は、ステージング・キャッシュ上のファイルを NFS 経由で授受する。ASGC では、2 台の NFS サーバを Active-Stanby 構成で運用し、6 台の SSD を RAID0 の構成にしてキャッシュ領域を作成してサービスを提供している。

仮想クラスタ上で動作するアプリケーションが直接使うストレージとしては、上記のデータボリュームを利用できるほか、専用のストレージを ASGC のデータ用ネットワークに接続可能である。ストレージの用意はユーザ側の負担になるが、アプリケーションの I/O 特性に適したストレージが選択可能であるとともに、仮想マシンのライフサイクルに関わらず、永続的にデータを格納しておける利点がある。

なお、将来的には、RADOS をセカンダリストレージとしてだけでなく、プライマリストレージとしても利用する予定である。これにより仮想マシンのライブマイグレーションが可能となる。RADOS をプライマリストレージとして利用した場合の性能や安定性の検証、計算ノードのローカル SSD とのハイブリッド利用の研究を進めている。

4. 予備評価

ASGC の運用に向けて、3 つの予備評価を行った。1 つ目は仮想クラスタ構築ツール sgc-tools の動作確認、2 つ目は仮想クラスタの実行性能、3 つ目は Amazon EC2 への仮想クラスタの可搬性の確認である。

4.1 仮想クラスタの構築

`sgc-cluster` コマンドを用いて 128 ノードまでの仮想クラスタを構築できることを確認した。なお、原理的には 155 ノードまで起動は可能であるが、CloudStack のシステム VM とノード割当てが衝突するので、安定した性能を得られる保証がないので、そのような利用は現実的ではないと考える。仮想マシン 1 台のプロビジョニングにはおおよそ 2 分かかる。その大部分はセカンダリストレージの NFS サーバからプライマリーストレージへの転送にかかる時間である。RADOS クラスタストレージからの転送は、2 回目以降はテンプレートがステージング・キャッシュになるので、現時点では大きな問題になってはいない。一方、大規模な仮想クラスタ構築時など、複数の仮想マシンを並列にプロビジョニングする際は、ステージング・キャッシュの読み出し性能が性能律速になる。この点に関しては、運用状況を見ながら、`sgc-cluster` コマンドにおける仮想マシンの並列起動数の最適化や、ハードウェアの増強を検討する予定である。

4.2 仮想クラスタの計算性能

仮想クラスタの性能評価として、HPC Challenge Benchmark version 1.4.3 [16] に含まれる High Performance Linpack を用いて評価を行った。MPI は Open MPI version 1.6.5、コンパイラとして Intel compiler version SP1 1.1.106 を用いた。仮想クラスタの計算ノード数は 16 台とした。物理クラスタの性能は 6.77 TFLOPS と実行効率は 94.4%であった。一方、仮想クラスタの性能は 6.40 TFLOPS であり、物理クラスタに対して約 5.4%の性能低下が見られた。より大規模かつ詳細な性能評価については、改めて報告する予定であるが、仮想クラスタでも十分な性能が得られたと考える。

4.3 仮想クラスタの可搬性

ASGC で用いる CloudStack のテンプレートから、Amazon EC2 へ仮想クラスタをプロビジョニングするツールのプロトタイプ実装を開発し、動作を確認した。動作の流れは以下のとおり、テンプレートのアップロードと、仮想クラスタの起動の 2 段階に分けられる。まず、テンプレートのアップロードでは、ASGC で利用している qcow2 形式のテンプレートを raw 形式に変換し、`ec2-import-instance` コマンドを用いて、Amazon S3 に格納する。続いて、EC2 から起動するために、AMI (Amazon Machine Image) 形式に変換する。仮想クラスタのプロビジョニングでは、ASGC 同様に、フロントエンドノードと計算ノードの 2 つの AMI を用いて仮想クラスタをプロビジョニングできる。動的な計算ノードの増減も確認した。

上記とは逆向きに、EC2 から ASGC へとテンプレートを転送できるかは未確認であるが、`ec2-create-instance-export-task` コマンドを使えば、実現可能と考える。また、テンプレートの管理方法やユースケースなど、運用に向けて解決すべき課題はまだ残っている。

5. まとめと今後の予定

本論文では、高性能かつスケールアウト可能な HPC クラウドを提案し、その実装であるプライベートクラウド ASGC の設計と予備評価について述べた。ASGC は約 70 TFLOPS の計算性能を有し、2014 年 7 月から運用を開始した。そのシステムソフトウェアは、Apache CloudStack を基に仮想クラスタ構築や、PCI パススルーおよび SR-IOV 対応など、HPC クラウドの実現に資する機能拡張を施している。ユーザは、物理クラスタに匹敵する性能を有する仮想クラウドをオンデマンドで作成可能であり、さらに負荷に応じてノード数を動的に追加、削除できる。我々の知る限り、プライベートクラウドと商用クラウドに対して可搬性のある仮想クラスタを提供する HPC クラウドサービスは存在しない。

今後の予定として、まず、Amazon EC2 への仮想クラスタのプロビジョニングについて、定量的な評価を実施すると共に、実運用に耐えうるための頑健化を図る。次に、ASGC の全システムを用いた HPC クラウドの性能評価を行う。さらに、高速ライブマイグレーション Yabusame を用いた VM パッキングによる省電力運用など、これまで我々が研究開発してきた成果を積極的に展開する。そして、ASGC を中心に、運用で得た知見を研究開発に活かす DevOps サイクルを確立したいと考えている。

謝辞 ASGC の環境構築から運用に尽力されている、ASGC サポートチームに感謝します。なお、本研究の一部は、JSPS 科研費 (24700040) の成果を活用している。

参考文献

[1] XSEDE: Extreme Science and Engineering Discovery Environment: `https://www.xsede.org`.
[2] PRACE: Partnership for Advanced Computing in Europe: `http://www.prace-ri.eu/`.
[3] HPCI: High Performance Computing Infrastructure Consortium: `http://hpci-c.jp`.
[4] Tanaka, Y., Yamamoto, N., Takano, R., Ota, A., Papadopoulos, P., Williams, N., Zheng, C., Huang, W., Pan, Y.-L., Wu, C.-H., Yu, H.-E., Shiao, J. S., Ichikawa, K., Tada, T., Date, S. and Shimojo, S.: *Building Secure and Transparent Inter-Cloud Infrastructure for Scientific Application*, Advances in Parallel Computing, Vol. 23: Cloud Computing and Big Data, IOS Press (2013).
[5] 高野了成，池上 努，広渕崇宏，田中良夫：HPC クラウドの実現に向けた仮想化クラスタの性能評価，情報処理学会論文誌コンピューティングシステム（ACS），Vol. 5, No. 2, pp. 111–120 (2012).
[6] Pornkitprasan, P., Visoottiviseth, V. and Takano, R.: Engaging Hardware-Virtualized Network Devices in Cloud Data Centers, *Proceedings of ICT International Student Project Conference (ICT-ISPC2014)*, pp. 1–4 (2014).
[7] GEO Grid: `http://www.geogrid.org/`.
[8] Songle: `http://songle.jp/`.
[9] Songrium: `http://songrium.jp/`.
[10] Apache CloudStack: `http://cloudstack.apache.org/`.
[11] Serf: `http://www.serfdom.io/`.
[12] Zabbix: `http://www.zabbix.com/`.
[13] Weil, S. A., Leung, A. W., Brandt, S. A. and Maltzahn, C.: RADOS: A Scalable, Reliable Storage Service for Petabyte-scale Storage Clusters, *Proc. of the 2nd International Workshop on Petascale Data Storage (PDSW2007): Held in Conjunction with Supercomputing '07*, pp. 35–44 (2007).
[14] RADOS: Reliable Automatic Distributed Object Storage: `https://ceph.com/docs/master/rados/`.
[15] RADOS Gateway: `https://ceph.com/docs/master/radosgw/`.
[16] HPC Challenge: `http://icl.cs.utk.edu/hpcc/`.