情報処理学会論文誌　コンピューティングシステム　Vol.5　No.5　76–89 (Oct. 2012)

オーバレイネットワークを用いたマルチサイト仮想クラスタ構築システム

多田 大輝[1,a)]　市川 昊平[2]　伊達 進[3]　阿部 洋丈[3]　下條 真司[3]

受付日 2012年3月28日, 採録日 2012年7月9日

概要：近年，広域分散計算への期待が高まっている．とりわけ，科学研究においては，複数の計算機から仮想計算機という形で計算リソースを切り出し，それらを用いて計算クラスタを構築し，ユーザ専用の計算クラスタを提供できる仮想クラスタが注目されている．しかし，現状では複数サイトの計算機を利用した実用的な仮想クラスタ構築システムは提案されていない．これは，複数サイトの仮想計算機間で相互に通信可能なネットワークを構築すること，および単一のクラスタシステムとしてシステムソフトウェアなどのソフトウェアスタックをインストール・設定し，セットアップすることが難しい点に起因している．本論文では，クラスタ管理ツールである Rocks とオーバレイネットワーク技術を統合することで柔軟に計算リソースの割当てが可能なマルチサイト仮想クラスタ構築システムを提案し，プロトタイプ実装を行った．実装したシステムの有用性を測るため，エミュレートした WAN 環境上で構築されたマルチサイト仮想クラスタ上でのアプリケーション実行性能を評価した．その結果，構築した仮想クラスタがネットワーク通信量の少ない分散計算型のアプリケーションに対して実用上問題ない性能を示すことを確認した．

キーワード：広域分散計算，グリッド，オーバレイネットワーク，仮想クラスタ

A Multi-site Virtual Cluster Deployment System Using Overlay Network

Taiki Tada[1,a)]　Kohei Ichikawa[2]　Susumu Date[3]
Hirotake Abe[3]　Shinji Shimojo[3]

Received: March 28, 2012, Accepted: July 9, 2012

Abstract: Recently, wide-area distributed computing has been increasingly concerned. Particularly in the scientific research, virtual cluster, which aggregates virtual machines from multiple physical computers to provide a user-dedicated computing environment, has attracted scientists and researchers expectation. In reality, however, no practically available virtual cluster solution has been realized so far. This fact is explained from the difficulty of establishing a network where virtual machines composing a virtual cluster directly communicate and that of installing and configuring software stack as a single cluster system on them. In this paper, we have proposed and prototyped a multi-site virtual cluster deployment system that builds a virtual cluster aggregating virtual machines, each of which is located on a different site. To evaluate the practicality, we have measured the computational performance of a virtual cluster system deployed on a WAN-emulated environment. The result indicates that the virtual cluster provides computational performance enough to perform a parameter-study problem with less data transfer.

Keywords: wide-area distributed computing, Grid, overlay network, virtual cluster

1. はじめに

複数のサイトから柔軟に計算リソースを利用する広域分

[1] 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University, Suita, Osaka 565–0871, Japan
[2] 大阪大学情報基盤本部
Central Office for Information Infrastructure, Osaka University, Ibaraki, Osaka 567–0047, Japan
[3] 大阪大学サイバーメディアセンター
Cybermedia Center, Osaka University, Ibaraki, Osaka 567–0047, Japan
[a)] tada.taiki@ais.cmc.osaka-u.ac.jp

散計算環境への期待と関心が高まっている．広域分散計算技術は，地理的に分散した複数の計算機を単一の計算環境として統合し，大規模な計算要求を持つアプリケーションを高速に処理できる環境を構築する．今日の科学研究で取り扱うデータ量の急速な増大にともない，様々な科学分野の研究者らの広域分散計算技術への期待と関心がますます増大しつつある．

このような研究者らの要求に対し，Globus [1], Legion [2], Ninf [3] などのグリッドミドルウェアを用いて，複数サイトに位置する計算機を集約し単一の計算環境として利用する広域分散計算環境を実現する試みがさかんに行われてきた．グリッドミドルウェアで構築される計算環境では，個々のアプリケーションのジョブがミドルウェアの用意するスケジューラへ投入されることで，計算リソースがユーザ間で共有され，高速な分散計算が可能となる．

しかし，グリッドミドルウェアで構築される広域分散計算環境を情報技術に詳しくない研究者らが最大限に駆使・利用することは，いまだに容易ではないのが現状である．これは，広域分散計算環境を構築する計算機間において，OS，コンパイラをはじめ導入されているソフトウェアなどのソフトウェアスタックが異なる点，および，計算リソースが複数のユーザによって共有されることをこれまでのグリッドミドルウェアが前提としている点に原因がある．そのため，これまでのグリッドミドルウェアで実現される広域分散計算環境では，個々のユーザが自身のアプリケーションの実行要求にあわせた計算環境を準備する際に，その計算環境を構成する計算リソースを有する管理者との調整に多大な負担が生じるのが現状である．

近年では，このような背景から，複数の計算機から仮想計算機という形で計算リソースを切り出し，それらを集約して構築する仮想クラスタの研究がある．従来の異なるソフトウェア環境で構成される計算機を集約する場合とは異なり，個々の仮想計算機のソフトウェア環境を統一することが可能となる．そのため，個々のユーザに対して，独立した専用の計算環境を提供できると期待される．

しかし，現状では，以下の理由から依然として複数サイトの計算機を利用した実用的な仮想クラスタ構築システムは実現されていない．1 点目は，各サイトに位置する仮想計算機は NAT やファイアウォールの背後に位置することが多く，互いに直接的に通信できるネットワークの配備が困難な点があげられる．複数のサイト間で仮想クラスタごとに独立したネットワークを構築するためには，各サイトの管理者間で VPN の設定やスイッチに設定する VLAN ID の決定といった事前調整が必要であり，その調整に大きな負担がかかる．2 点目は，個々の仮想計算機のソフトウェア環境を均質にインストールし，またそれら仮想計算機を統一的に制御することが困難な点である．これまでに提案されている仮想クラスタ技術の多くは，各サイトの仮想計算機を集約して利用することに焦点がある [4], [5], [6] ものの，それらの計算機間でネットワークファイルシステム，スケジューラなどのセットアップはユーザにまかせられている．

本研究では，ただ仮想計算機を複数台接続するだけではなく，ユーザの要求に応じた仮想計算機のソフトウェア環境までもセットアップした，マルチサイト仮想クラスタとして構築するシステムを提案する．すなわち，提案するシステムでは，複数のサイトに位置する仮想計算機の起動，仮想計算機間でのネットワークの配備，およびそれらの仮想計算機を単一のクラスタシステムとして利用するためのシステムソフトウェアのセットアップに必要なプロセスを自動化する．これにより，ユーザらが占有して利用できる仮想クラスタを提供する．

以下，2 章では，マルチサイト仮想クラスタ構築システムの設計指針と実装へのアプローチについて述べる．3 章では，システムの概要と詳細実装についてまとめる．4 章では，プロトタイプ実装したシステムで実際に構築した仮想クラスタの実用性に関して議論するとともに，マルチサイト仮想クラスタ上でのアプリケーション実行性能の評価を行い，システムの有用性を示す．5 章で関連研究を述べ，6 章で本論文をまとめ，最後に 7 章で今後の課題を示す．

2. 設計指針と実装へのアプローチ

2.1 設計指針

図 1 は本研究で目指す複数サイトの仮想計算機を用いたマルチサイト仮想クラスタ構築システムの概念を示したものである．マルチサイト仮想クラスタ構築システムの実現には，前章で記したように，仮想計算機間で直接的に通信可能なネットワークの配備および，仮想クラスタを構成するすべての仮想計算機に対して，OS やコンパイラなどのインストール，ネットワーク，システムソフトウェアの設定を行うための統一的な管理・制御可能な仕組みが必要不可欠である．

まず，前章の 2 点目の問題としてあげた個々の仮想計算機へのシステムソフトウェアのセットアップが困難な点に着目する．この問題に対しては，本研究では，現在，クラスタシステムのセットアップで広く利用されているクラス

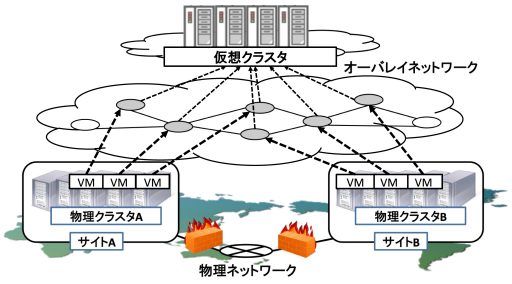

図 1　マルチサイト仮想クラスタの概念図
Fig. 1　Concept of multi-site virtual cluster.

タ構築ツールによる解決を考える．クラスタ構築ツールの一例としては，Rocks [7]，Lucie [8]，OSCAR [9] などがあげられる．これらのクラスタ構築ツールは，構築ツールのイメージをインストールしたフロントエンドノードを起点とし，ローカルネットワーク内の複数の計算機に対して，ネットワークを経由して，OS，コンパイラなどのソフトウェアに加え，ジョブスケジューラやネットワークファイルシステムのインストールを自動的に行う．クラスタ構築ツールが有するクラスタセットアップの自動化の仕組み，および多くの計算クラスタの構築において利用されている点を考慮すると，既存のクラスタ構築ツールの機能を継承しつつ，複数サイトの仮想クラスタを構築できることが望ましい．

次に前章の 1 点目の問題に着眼する．この問題に対しては，クラスタ構築ツールの機能にソフトウェアによる論理的なネットワークを構築するオーバレイネットワーク技術を応用することを考える．オーバレイネットワーク技術は，物理的なネットワーク構成や NAT，ファイアウォールなどによる直接的な通信の障害を越えて相互に通信可能な仮想ネットワークの構築を可能にする．また，オーバレイネットワーク技術の機能を適切に利用・拡張することで，仮想クラスタごとに独立した通信が可能な仮想ネットワークの構築が可能になると考える．

既存のクラスタ構築ツールは，ローカルネットワークに接続された物理的な計算機を用いたクラスタ構築を前提としており，複数のサイトの仮想計算機を用いたクラスタ構築システムで利用することは想定していない．そこで，本研究では個々の仮想計算機がローカルネットワークに接続されているかのように見せるため，仮想クラスタを構成する仮想計算機間を仮想ネットワークで接続する．その後，構築された仮想ネットワーク上で，複数サイトの仮想計算機を既存のクラスタ構築ツールを用いて単一の仮想クラスタとしてセットアップすることが可能になると考える．

2.2　実装へのアプローチ

本研究では，前節で述べたように，マルチサイト仮想クラスタを実現するために，仮想クラスタを構成する仮想計算機間にオーバレイネットワーク技術による仮想ネットワークを構築し，その上で既存のクラスタ構築ツールを応用することを試みる．

クラスタ構築ツール Rocks は，ネットワーク設定などの基本設定を施し，ある一定の用途のソフトウェア群を管理する Roll をユーザの要望にあわせて選択するのみで，ほぼ全自動でクラスタシステムが必要とするユーザアカウントの同期，ジョブスケジューラの設定，Home ディレクトリの共有などの設定を行う．そのため，ユーザは迅速にクラスタシステムを構築，利用することができる．この高い利便性から Rocks は世界中の研究機関や大学で広く利用されている．また，バージョン 5.0 以降の Rocks は Xen [10] をハイパーバイザとした仮想クラスタを構築する機能も提供しており，単一サイトの物理クラスタシステム上で，ユーザごとに異なる VLAN で区切られた仮想ネットワークを利用した仮想クラスタを構築することも可能である．今日では数多くのクラスタ構築ツールがあるが，本研究では，上述のように Rocks が単一サイト内での仮想クラスタ構築をサポートしていることに加え，その安定性，利用実績を考慮し，マルチサイト仮想クラスタ構築システムを実現するための基盤技術として，Rocks を選定することとした．

次に，仮想計算機間で構築する仮想ネットワークに，オーバレイネットワーク技術を用いた VPN を実現するソフトウェアである N2N [11] を用いることとした．N2N は P2P によるオーバレイネットワーク技術を利用したレイヤ 2 レベルの仮想プライベートネットワークを提供するソフトウェアである．レイヤ 2 レベルの仮想ネットワーク環境を構築するため，既存の多くのアプリケーションやツールに対して複数サイトにまたがる透過的な単一のネットワーク空間を提供することができる．そのため，複数サイトの統合環境において，既存のソフトウェア資産を最大限に有効利用することを可能としている．

オーバレイネットワーク技術としては，IPOP [12]，VIOLIN [13]，ViNe [14] など数多くの技術が提案，実装されている．本研究では，クラスタ構築ツール Rocks が PXE ブートを利用したクラスタシステムのセットアップ・インストールを行う点，および個々の仮想クラスタごとに独立した通信を実現する機能を有している点に着目し，マルチサイト仮想クラスタ構築システムを実現するための基盤技術として，クラスタごとに独立したレイヤ 2 レベルの仮想ネットワークを構築できる N2N を選定した．

本研究では，Rocks の仮想クラスタ構築を，複数サイトの仮想計算機間を接続する N2N によるオーバレイネットワーク上で行うことにより，マルチサイト仮想クラスタを構築できるように拡張する．N2N が提供する透過的な仮想ネットワークを構築する機能を活用し，Rocks の仮想クラスタ構築機能を複数サイト統合環境の上で透過的に機能させる．これにより，複数サイトの計算機から余剰計算資源を仮想計算機という形で切り出し，それらを仮想クラスタとして利用できるようにセットアップを行う．

3.　マルチサイト仮想クラスタ構築システム

本章では，マルチサイト仮想クラスタ構築システムを実現するために設計，実装した MVC コントローラについて概説する．その後，システムの基盤技術である Rocks クラスタによる仮想クラスタの構築手法と N2N のアーキテクチャについて説明する．最後に，MVC コントローラの実装の詳細について，基盤技術である Rocks と N2N がどのように統合，拡張されたかという視点から説明する．

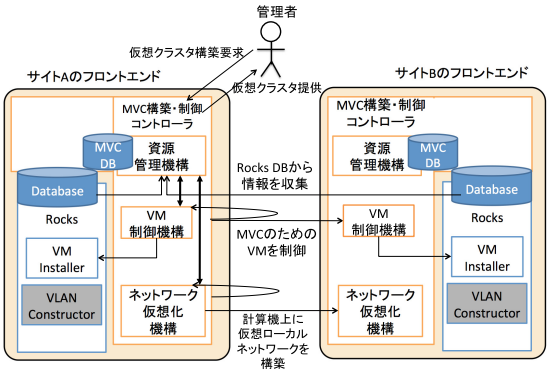

図 2　マルチサイト仮想クラスタシステム概要
Fig. 2　Outline of multi-site virtual cluster system.

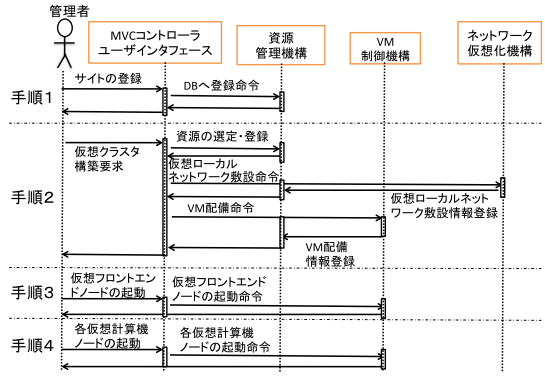

図 3　MVC コントローラの動作手順
Fig. 3　Operational flow on MVC controller.

3.1　システムの概要

本研究では，マルチサイト仮想クラスタ構築システムを実現するために，MVC（Multi-site Virtual Cluster）コントローラを設計した．MVC コントローラは，既存の Rocks が有する単一サイト内仮想クラスタ構築の機能に影響を与えることなく，マルチサイト仮想クラスタを構築する機能を提供する．

MVC コントローラの概要を**図 2** に示す．MVC コントローラは，1) 仮想クラスタを構成する各サイトの仮想計算機の情報および利用可能な計算リソース情報を管理する資源管理機構，2) 複数サイトにまたがる仮想ネットワークを構築・維持するネットワーク仮想化機構，3) 仮想計算機の起動および破棄処理を担当する VM 制御機構の 3 点の機構から構成される．

この MVC コントローラを用いて仮想クラスタを構築する手順を**図 3** に示す．まず，ユーザは仮想クラスタを構築する際に利用する各サイトの Rocks クラスタを指定する（手順 1）．これにより，資源管理機構は複数サイトに位置する Rocks クラスタが所有する拡張したデータベースから，各クラスタが所有している計算リソースやその使用状況などを含んだ情報を取得する．なお，本研究で拡張したデータベースについては，以降で詳説する．次に，ユーザは手順 1 で資源管理機構が取得した各サイトにおける利用可能な計算リソース情報に基づき，仮想クラスタに使用する計算リソースを選択する（手順 2）．その後，ネットワーク仮想化機構が，選択されたリソース間で N2N による仮想ネットワークを構築する．その後，VM 制御機構が，仮想クラスタ構築の起点となるフロントエンドのインストールを行う（手順 3）．最後に，VM 制御機構が，N2N の構築した仮想ネットワークを各仮想計算機に接続し，それらを起動しインストールを行う（手順 4）．以上により，複数サイトの仮想計算機からなる仮想クラスタが構築される．

このような分散した複数のコンポーネントが相互連携するシステムでは，各コンポーネント間の認証がセキュリティ側面から重要となる．本論文執筆時点では，MVC コントローラは各サイトの root の ssh 公開鍵を事前に Rocks クラスタのフロントエンドノードに登録しておくことで，各機構，データベースへのアクセスを行っている．しかし，このような方法はセキュリティ側面からは現実的ではなく，より現実的かつ実用的なセキュリティ機構が必要であると考えている．

そこで，本研究では，Rocks が実装している，root 権限が必要な仮想クラスタの制御を一般ユーザでも利用可能にするサービス AirBoss [15] を拡張，あるいは同様の仕組みを MVC コントローラ内に実装し，マルチサイト仮想クラスタを構築するうえで必要な仮想計算機，仮想ネットワークデバイス，データベースの制御を一般ユーザ権限でリクエスト可能とするセキュリティ機構を今後設計・実装する予定である．このようなセキュリティ機構により，各サイトのクラスタ管理者はユーザの公開鍵を，そのユーザが利用する Rocks クラスタにあらかじめ登録しておくだけで，マルチサイト仮想クラスタの制御を簡便に許可できると考えている．

3.2　Rocks による仮想クラスタ構築手法

Rocks はクラスタ構築を自動化するために，PXE ブートによるネットワーク経由でのインストール方式を採用している．ユーザはフロントエンドのみ手動でインストールし，計算機ノード群は PXE ブートにより自動的にフロントエンドノードを探し出し，インストールを開始する．PXE ブートによる自動インストールの仕組みは次のとおり行われる．1) 計算機ノードは起動時にブロードキャスト通信を行って，フロントエンドノードの発見を行う．2) フロントエンドノードを発見すると DHCP により IP アドレスの自動割当てを受ける．3) その後，フロントエンドより OS やその他システムソフトウェアのイメージをネットワーク経由で取得し，自動的にインストールを始める．

Rocks がサポートする単一クラスタ内での仮想クラスタ構築においても，この PXE ブートによる仮想計算機のインストールが行われる．ただし，仮想クラスタ構築の場合は 1 つの物理クラスタ上に複数の仮想クラスタが混在す

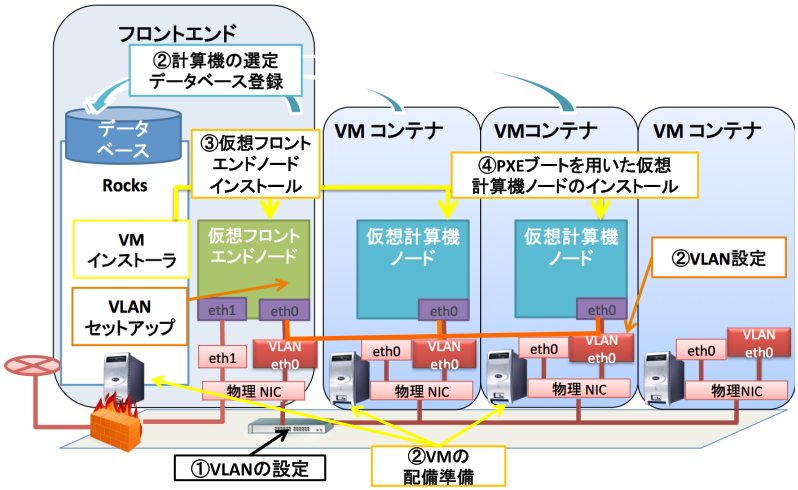

図 4　従来の Rocks による仮想クラスタの構築
Fig. 4　Constructing virtual cluster using original Rocks.

ることになるので，各仮想クラスタごとに独立したネットワークが構築される．特に PXE ブートは仕組み上，レイヤ 2 レベルのブロードキャスト通信が必須であるため，レイヤ 2 レベルでの独立性の確保が必要となる．

Rocks は各仮想クラスタ間のネットワークの分離に，タグ VLAN を用いる．仮想クラスタを構築する際には，ユニークな VLAN ID を個々の仮想クラスタに割り当て，物理計算機上ではタグ VLAN に対応するネットワークデバイスが作成される．たとえば，物理クラスタを構成する物理計算機間のデータの通信経路となっているネットワークデバイス eth0 に対して，VLAN ID が 2 のポートを作成すると新たに eth0.2 という仮想的なネットワークデバイスが生成される．Rocks はこの VLAN ID に対応する仮想的なネットワークデバイスと仮想クラスタを構成する仮想計算機のネットワークデバイスを，Xen が提供するブリッジデバイスに接続する．その結果，仮想計算機が送信したパケットには，物理計算機のネットワークデバイスを経由するときに，その仮想計算機が属している仮想クラスタに割り当てられた VLAN ID が取り付けられる．送信先の物理計算機から仮想計算機へ受け渡されるときにはこの VLAN ID が取り外され，送信先の仮想計算機に転送される．このように，各仮想クラスタを構成する仮想計算機どうしは，物理計算機に設定されている VLAN の設定を意識することなくクラスタごとに分離した通信が可能となっている．

Rocks が仮想クラスタを構築する手順を図 4 に示す．このとき，クラスタに接続されたスイッチの各ポートが任意の VLAN タグ付きパケットを通過できるように，クラスタ管理者はすべてのポートをトランクポートとして設定し，VLAN ID が設定されたパケットを通過できるように管理者は設定しておく必要がある（①）．次に，Rocks が提供する「rocks add cluster」というコマンドを実行することによって，各物理計算機から仮想計算機を割り当てる計算機

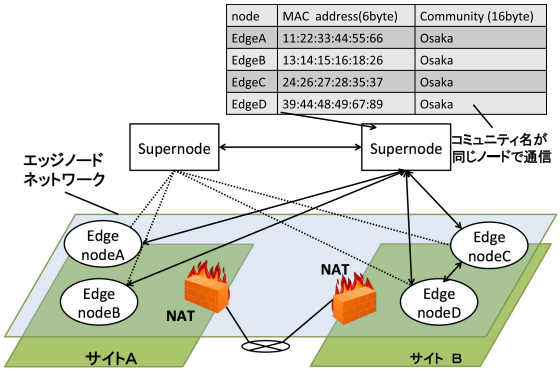

図 5　N2N のアーキテクチャ
Fig. 5　Architecture of N2N.

の選定およびその情報がフロントエンドの保持するデータベースへ登録される．これと同時に，VLAN ID も新たに発行され，VLAN ID が割り当てられたネットワークデバイスが生成され，ブリッジデバイスを介して仮想計算機に接続する準備がされる（②）．次に，「rocks start host vm ⟨フロントエンドのホスト名⟩」というコマンドの実行により，仮想クラスタのフロントエンドノードのインストールを開始する（③）．フロントエンドのインストールが終わった後，「rocks start host vm ⟨仮想計算機のホスト名⟩」というコマンドを実行し，各仮想計算機ノードのインストールを順次開始する（④）．

3.3　N2N のアーキテクチャ

N2N は P2P によるオーバレイネットワーク技術をベースとしたレイヤ 2 レベルの暗号化した仮想プライベートネットワークを提供する．N2N は図 5 に示すように supernode と edge から構成される．edge が各末端の計算機上に配置されるプログラムであり，Linux の tap ドライバをベースとしたレイヤ 2 レベルの仮想ネットワークデバイスを提供する．この仮想ネットワークデバイスを通じて送受信さ

れるデータは N2N の supernode と edge が構成する P2P ネットワーク内で交換され，目的の edge まで運ばれる．supernode はグローバル IP の割り当てられている計算機上で動作するプログラムで，各 edge の MAC アドレスとネットワーク上のルーティングを管理する．さらに，グローバル IP の割当てのない edge の通信の中継を行う．それゆえ，仮に edge が NAT の背後にある場合でも，edge 間で supernode を介して通信を行うことで，edge が互いに通信することを可能にする．グローバル IP アドレスの割当てのある edge どうしは supernode の管理テーブルに従って，互いに直接通信を行う．

また，N2N は，各 edge に割り当てられた MAC アドレスおよびあらかじめ決定した固有のコミュニティ名を用いて，オーバレイネットワーク上に存在する特定の edge 間で独立したレイヤ 2 仮想ネットワークを構築することが可能である．コミュニティ名は edge を起動する際のコマンドラインオプションで与えることができ，supernode は共通のコミュニティ名を持つ edge 間でのみ通信を可能とすることで，独立したネットワークを構成する．

3.4　MVC コントローラの実装

3.4.1　MVC コントローラのインタフェース

ユーザがマルチサイト仮想クラスタの構築を行うために，Rocks が提供しているコマンドラインインタフェースを拡張し，以下に示す MVC コントローラ用のユーザとのインタフェースとなるコマンドを新たに追加した．

```
$ rocks add site \
  fqdn="ip address or host name" \
  user="user name" ssh-key="public key"
```

「rocks add site」コマンドは，図 3 の手順 1 に対応し，仮想クラスタの構築に利用する各サイトの Rocks クラスタシステムの情報をデータベースに登録する．指定する内容は，サイトの Rocks クラスタのフロントエンドノードが持つ IP アドレスあるいはホスト名と，クラスタにログインするユーザ名，ログイン時に利用する SSH の公開鍵のディレクトリのパスである．このコマンドを実行することで，資源管理機構用に設計したデータベースにサイト情報を登録する．

これらサイトの情報は，複数サイトに位置する Rocks クラスタを用いたマルチサイト仮想クラスタの構築において，計算機の選定や仮想ネットワークの敷設，仮想計算機ノードの配備，および起動時に他サイトへログインする際に利用する．

```
$ rocks add mvc site="siteA:siteB:…" \
  num-computes="4:5:…."
```

「rocks add mvc」コマンドは，図 3 の手順 2 にあたる，仮想クラスタの構築を MVC コントローラに要求するコマンドである．引数として，利用するサイト名，各サイトで割り当てる仮想計算機ノードの数を指定する．実際のマルチサイト仮想クラスタのリソース確保は各サイトにおけるローカル仮想クラスタ配備の機能を利用することで実現している．たとえば，サイト B から 4 ノードを確保する場合は，サイト B 上で 4 ノードからなるローカル仮想クラスタを配備するコマンドを実行する．こうすることで，4 ノード仮想マシンがサイト B で利用中であることが従来の Rocks のデータベースにも登録され，既存実装のリソース割当てメカニズムと競合することなくマルチサイト仮想クラスタに使用するリソースの確保を可能としている．

```
$ rocks start host vm overlay "hostname"
```

```
$ rocks start rhost vm overlay "hostname" \
  site="siteB"
```

「rocks start host vm overlay」，「rocks start rhost vm overlay」コマンドは，各 VM を起動するために利用する．「rocks start host vm overlay」コマンドは，図 3 の手順 3 に相当し，VM 制御機構に対して仮想フロントエンドノードや仮想計算機ノードを N2N が構築する仮想ネットワークに接続する形で起動するよう命令する．また，「rocks start rhost vm overlay」コマンドは，図 3 の手順 4 にあたり，他サイトにある仮想計算機に対して VM 制御機構を用いた VM の起動を命令する．

3.4.2　資源管理機構

資源管理機構は，複数サイトに位置する Rocks クラスタの計算リソースの情報，オーバレイネットワークに関する情報およびマルチサイト仮想クラスタを構成する仮想計算機に関する情報を一元的に管理する．Rocks では，クラスタを構成する計算機の情報と各仮想計算機に割り当てる計算リソースの情報や VLAN ID などの仮想ネットワークに関する情報を，フロントエンド上の MySQL データベースで管理している．資源管理機構では，マルチサイト仮想クラスタの起点となる Rocks クラスタのデータベースを拡張し，複数サイトで分散的に管理されている Rocks クラスタの有する情報を管理する．拡張したデータベーステーブルを**図 6** に示す．

マルチサイト仮想クラスタを構築，管理するためには，複数サイトの Rocks クラスタから利用可能な計算機資源を見つけ出す必要がある．つまり，マルチサイト仮想クラスタを構成する仮想計算機の割当て先である，各 Rocks クラスタを構成する物理計算機の利用状況を把握しなければならない．そこで，各サイトの Rocks クラスタの情報を管理する Site テーブル，各サイトの Rocks クラスタを構成する物理計算機の情報を管理する phy_nodes for MVC テーブル，各物理計算機上の仮想計算機と物理計算機の対応を管

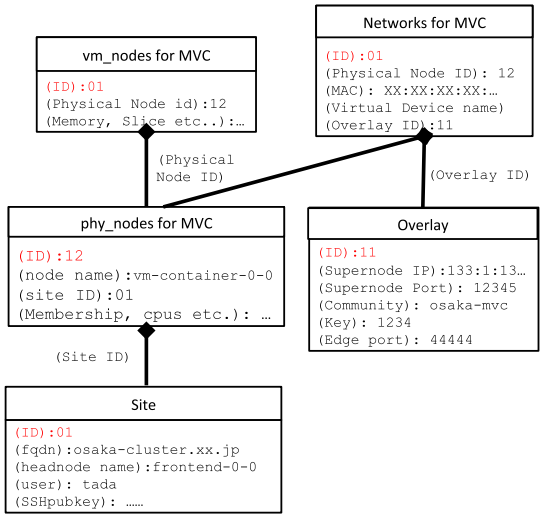

図 6　追加したデータベーステーブル
Fig. 6　RDB tables added to MVC controller.

理する vm_nodes for MVC テーブルを追加した．Site テーブルでは，各サイトの ID とサイト名，Rocks のフロントエンドノード名，ログインを許可するユーザ名，ログインに利用する SSH の鍵を管理する．phy_nodes for MVC テーブルでは，VM コンテナの ID とそのホスト名，所属するサイトの ID，CPU などの計算機の情報を管理し，vm_nodes for MVC テーブルでは，仮想計算機ノードと VM コンテナの対応づけるために個々の ID，VM に割り当てられるメモリ量をそれぞれ仮想計算機ノードごとに管理する．

また，仮想ネットワークが敷設される VM コンテナ上では，オーバレイネットワークを構成する N2N の tap デバイスを仮想クラスタごとに設置する．そのため，仮想クラスタごとのオーバレイネットワークの区別をするとともに，各仮想計算機ノードとそれに対応した N2N の tap デバイスをそれぞれ管理する必要がある．このような観点から，本研究では仮想ネットワーク自体を管理する Overlay テーブル，N2N の tap デバイスの配置情報など仮想ローカルネットワークの構成に関連する情報を管理する Networks for MVC テーブルを作成する．

Overlay テーブルでは，N2N オーバレイネットワークごとの ID のほか，N2N のオーバレイネットワークを構築するために必要な supernode が存在するフロントエンドノードの IP アドレスとポートの情報やコミュニティ名，共通鍵，edge が利用するポートの情報を管理する．Networks for MVC テーブルでは，ネットワークに関連する情報を管理する．具体的には，フロントエンドノードと VM コンテナ群の ID，N2N の tap デバイスの名前と割り当てられる MAC アドレス，接続するオーバレイネットワークの ID を管理する．

本研究で実装したプロトタイプシステムでは，資源管理機構において，マルチサイト仮想クラスタを構成する各仮想計算機の MAC アドレスは，Rocks がランダムに割り当てるものを利用している．しかし，実際には複数のサイト間において，仮想計算機間の MAC アドレスの重複を確認し，再割当てできる仕組み，たとえば，データベースに登録するときにチェックを行い，重複がある場合は再度 MAC アドレスを発行するといった仕組みを実装する必要がある．

3.4.3　ネットワーク仮想化機構

ネットワーク仮想化機構は，複数サイトに位置する仮想計算機間でレイヤ 2 ネットワークを構築するために，Rocks が仮想クラスタを構築する際に用いる VLAN を N2N によるオーバレイネットワークで行うようにする．

ネットワーク仮想化機構は，「rocks add mvc site」コマンドにより，仮想クラスタの起点となる Rocks クラスタのフロントエンドで supernode を起動する．次に，「rocks start host vm overlay」，「rocks start rhost vm overlay」コマンドにより，拡張したデータベースの情報を基に，資源管理機構が選択した仮想計算機ごとに edge を起動する．このとき，supernode で作られる経路表が VM に割り当てられた MAC アドレスをベースに作成される必要があるため，supernode から見える edge の MAC アドレスを，VM に割り当てられる MAC アドレスと一致させておく必要がある．したがって，ネットワーク仮想化機構は，資源管理機構により渡された各仮想計算機の MAC アドレスを指定して edge を起動し，**図 7** に示すように，Xen のブリッジデバイスに接続する．

3.4.4　VM 制御機構

VM 制御機構は，「rocks start host vm overlay」，「rocks start rhost vm overlay」コマンドにより，資源管理機構が選択した計算リソースにおいて，仮想計算機を起動する．ネットワーク仮想化機構は，仮想クラスタにネットワークを提供する際に，上記のネットワーク仮想化機構が作成した N2N の仮想ネットワークデバイスと仮想計算機のネットワークインタフェースをブリッジデバイスを経由して仮想計算機のネットワークデバイスに接続する．これにより，起動される仮想計算機に対して，透過的に N2N が構築したオーバレイネットワークが割り当てられる．

4.　評価

本章では，最初に我々の構築したマルチサイト仮想クラスタ構築システムを用いて，実際に複数サイトにまたがった仮想クラスタが構築できることを確認する．次に，構築した仮想クラスタが実際の WAN 環境でどの程度のアプリケーション実行性能の影響を受けるか確認する．

4.1　マルチサイト仮想クラスタの実用性に関して

我々の構築したマルチサイト仮想クラスタ構築システムの実用性を確認するために，**図 8** に示す環境下でマルチサイト仮想クラスタの構築を行った．まず，複数サイトに位置する複数の Rocks クラスタを想定し，1 台のフロントエ

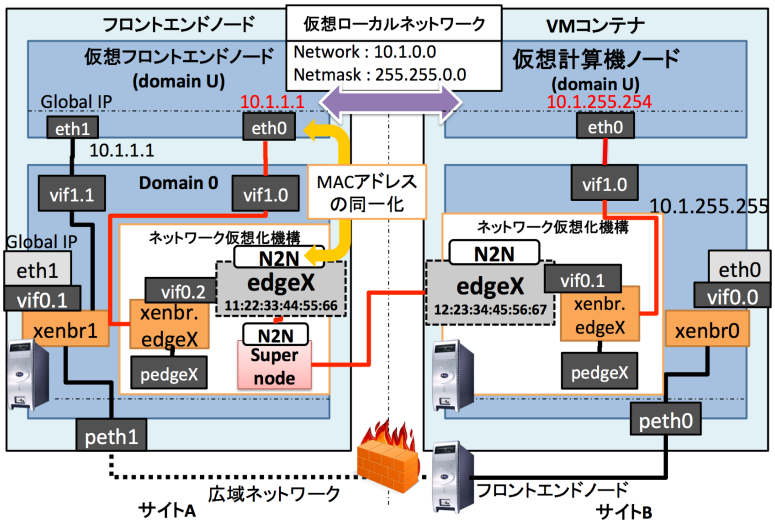

図 7　ネットワーク仮想化機構

Fig. 7　Configuration of network virtualization.

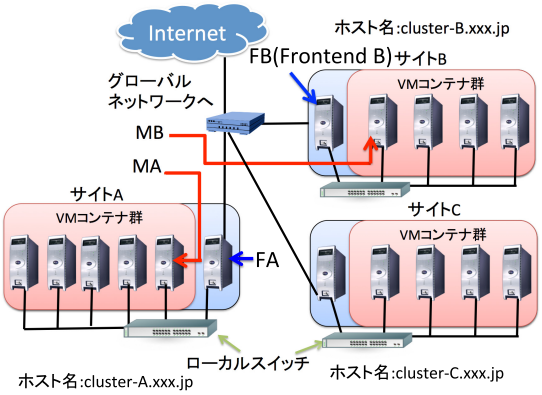

図 8　評価環境

Fig. 8　Evaluation Environment.

表 1　各ノードに関する情報

Table 1　Specification and configuration of node.

OS	CentOS 5.5 (Final)
Hypervisor	Xen 3.0
CPU	Intel Xeon(R) E5520 @ 2.27 GHz x2
Memory	12 GB
Cluster Management Software	Rocks 5.4

ンドノードと 5 台の VM コンテナで構築される Rocks クラスタ A と，1 台のフロントエンドと 4 台の VM コンテナで構築される Rocks クラスタ B，C をセットアップした．個々の Rocks クラスタを構成する計算機にインストールされている OS，CPU などの情報を**表 1** に示す．それぞれの Rocks クラスタはそれぞれ 1 Gbps のローカルネットワークスイッチで単一セグメントをなすクラスタを構成しており，それらは 1 Gbps の単一のルータで集約されている．

これら 3 基の Rocks クラスタを用いて，Rocks クラスタ A のフロントエンド上に起動した仮想フロントエンドを起点とした仮想クラスタを構築した．構築した仮想クラスタは，1 台の仮想フロントエンドと各 Rocks クラスタを構成する VM コンテナにそれぞれ 1 台ずつ起動された 13 台の仮想計算機から構成される．各仮想計算機は，N2N の仮想ネットワークを介して接続されており，supernode は，Rocks クラスタ A の物理的なフロントエンド上で起動している．

実際に構築した仮想クラスタ上では，仮想計算機全体で物理的なクラスタと同様に利用できるクラスタ環境が構築されていることを確認した．また，構築された仮想クラスタ上でセットアップされたジョブスケジューラなどを用いて，MPI などのノード間通信が発生するアプリケーションの実行が可能なことも確認できた．

なお，上記のマルチサイト仮想クラスタ環境の構築手順は次のとおりとなる．

1) 利用する各リモートサイトの情報を登録

```
$ rocks add site \
  fqdn="cluster-A.xxx.jp" \
  user="tada" ssh-key="/path/to/key_dir"
```

2) マルチサイト仮想クラスタの構成を決定

```
$ rocks add mvc \
  site="cluster-A.xxx.jp:cluster-B.xxx.jp: \
  cluster-C.xxx.jp" \
  num-computes="5:4:4"
```

3) マルチサイト仮想クラスタの仮想フロントエンドノードの起動，インストール

```
$ rocks start host vm \
  overlay frontend-0-0-0
```

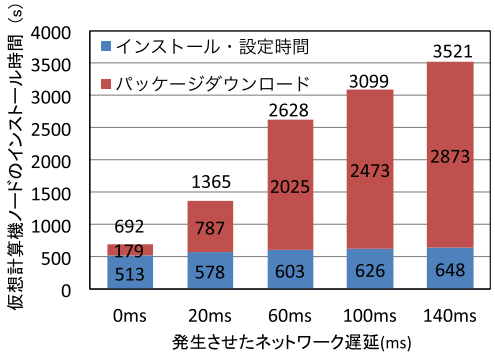

図 9　仮想クラスタ構築時間
Fig. 9　Building time of virtual cluster.

4) サイト A の仮想計算機ノードおよびサイト B，C 上の仮想計算機ノードの起動およびインストール

```
$ rocks start host vm \
  overlay hosted-vm-0-0-0

$ rocks start rhost vm \
  overlay hosted-vm-0-0-5 \
  site="cluster-B.xxx.jp"
```

以下，上記と同様のコマンドにより，サイト A，B，C の仮想計算機 11 台のインストールを行う．

4.1.1　マルチサイト仮想クラスタ構築時間および構築可能性に関して

前節では，ローカルネットワーク環境において，13 台の仮想計算機からなるマルチサイト仮想クラスタの構築の確認を行った．本項では，ネットワーク遅延の大きい広域環境下で，提案手法によるマルチサイト仮想クラスタが構築可能であるか検証を行う．前節と同様の環境で，サイト B とサイト C を用い，ネットワーク遅延を発生させた擬似的な広域環境を構築し，8 台の仮想計算機からなるマルチサイト仮想クラスタを構築した．この際，インストールする Roll のパッケージ群として，bio，ganglia，os，web-server，base，condor，hpc，kernel，sge の 9 種類の Roll を選択，インストールした．ネットワーク遅延は，サイト A のフロントエンド上のネットワークデバイスに対して，20 ms，60 ms，100 ms，140 ms と増加させつつ，マルチサイト仮想クラスタの構築にかかる時間を測定した．結果を**図 9** に示す．なお，遅延時間に関しては，複数サイトにまたがる仮想クラスタを想定し，大阪と筑波間の往復遅延（Round-trip Time; RTT）が 40 ミリ秒（ms），大阪とアメリカのカリフォルニア州間の RTT が 120 ms であることを考慮して，設定した．

図に示すとおり，すべての場合においてマルチサイト仮想クラスタの構築は可能であったが，構築時間は遅延の増加に合わせて増加傾向にある．ネットワークの遅延が 20 ms のときは，22 分 45 秒，60 ms のときは 43 分 48 秒，100 ms の場合，51 分 39 秒の構築時間を要した．

これらの結果から考えれば，国内の大学や研究機関などの余剰計算資源を用いて，2 サイト 16 ノード規模のマルチサイト仮想クラスタであれば，20 分以内で構築が可能であるといえる．計算資源を有していない研究機関の研究者が余剰計算資源を有効利用して，この程度の構築時間でマルチサイト仮想クラスタを構築できることのメリットを考えれば，この構築時間は許容範囲であるとも考えられる．しかし，遅延の大きい広域ネットワークにおいては，2 サイト 16 ノード規模のマルチサイト仮想クラスタであっても，構築時間を小さくすることは今後の課題となる．

4.2　構築したマルチサイト仮想クラスタの性能評価

構築したマルチサイト仮想クラスタを用いて性能評価を行った．最初に，ネットワークに遅延を加えることで生じる仮想ネットワークの帯域へ影響を計測した．次に，マルチサイト仮想クラスタを構成する仮想ネットワークの通信性能の劣化の要因を明確にするために，N2N の通信性能に関して詳細評価を行った．次に，N2N を介して接続されたマルチサイト仮想クラスタが，実際の WAN 環境におけるサイト間のネットワーク遅延によって，アプリケーションにどの程度影響を与えるかをアプリケーションの実行時間から性能評価した．

4.2.1　通信性能の評価

ここでは，提案システムにより構築したマルチサイト仮想クラスタを構成するノード間での通信性能を評価する．本評価のために，図 8 に示した環境上で以下の 4 項目を，ルータにかかる通信遅延を変更しながら計測し，本提案手法のネットワーク通信性能について議論する．

(1) 物理ネットワーク：サイト A，サイト B のフロントエンド間（FA，FB）の通信性能
(2) VLAN：サイト A とサイト B の物理計算機を VLAN で接続してオリジナルの Rocks で仮想クラスタを構築した場合の通信性能
(3) VLAN+VPN：サイト A，サイト B のフロントエンド間（FA，FB）を VPN で接続し，サイト内では VLAN を用いてオリジナルの Rocks でマルチサイト仮想クラスタを構築した場合の MA，MB 上に配備された仮想計算機間の通信性能
(4) 提案手法：サイト A とサイト B の計算資源を N2N で接続し，マルチサイト仮想クラスタを構築した際の物理計算機 MA，MB 上に配備された仮想計算機間の通信性能

この際，サイト間の VPN には，広く一般的に利用されている OpenVPN [16] を用いた．また，通信性能は，Netperf [17] を用いて測定した．**図 10** に結果を示す．

この結果より，追加するネットワーク遅延を増加させると，いずれの場合においても通信性能が低下することが確

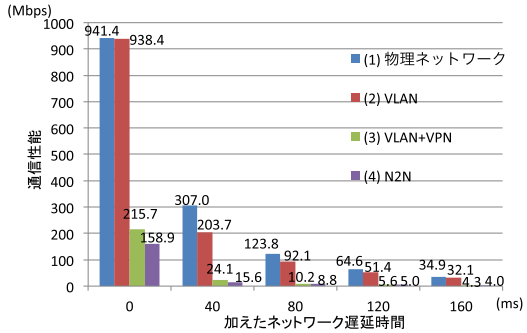

図 10　仮想ネットワークの通信性能
Fig. 10　Throughput of virtual network.

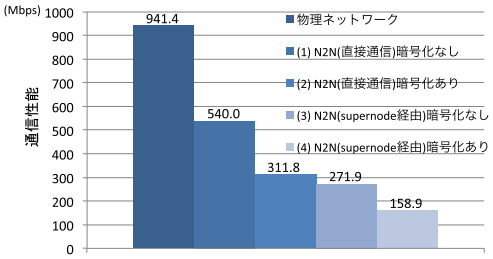

図 11　N2N の性能評価
Fig. 11　Performance of N2N.

認できる．また，追加するネットワーク遅延時間にかかわらず，物理ネットワーク，VLAN，VLAN+VPN，N2N の順に通信性能が高いことも確認できる．

さらに，(2) VLAN で構築した仮想クラスタの通信性能は，(3) VLAN+VPN，(4) N2N による方法に比較して非常に良いことが分かる．しかし，複数サイトの計算資源間で VLAN を形成するためには，サイト間をまたいで VLAN タグのついたパケットを送受信できるように通信経路上のすべてのスイッチに適切な設定を施す必要があり，各サイトの管理者間の調整作業も大きく現実的ではない．(3) VLAN+VPN による方法は，VPN を使用する区間は両端のスイッチのみを設定すればよいため，(2) VLAN による手法ほど調整負荷は高くないものの，依然として管理者間の調整負荷は存在する．(4) の提案手法では，仮想クラスタを構成する計算ノードが複数サイトに配置されたとしても，VLAN や VPN などの設定や管理者間の調整負荷は増加しない．通信性能は最も低くなるが，(3) の VLAN+VPN と比較した場合の性能低下はわずかである．

4.2.2　仮想ネットワーク N2N の詳細評価

本項では，前項で測定した本システムのマルチサイト仮想クラスタを構成する仮想ネットワークの通信性能の低下要因を明らかにするため，N2N の詳細な評価を行う．本評価では，前項と同様に図 8 の環境を用い，下記に示すように N2N の暗号化処理および supernode を経由することでの性能低下を計測し，それぞれがどの程度，N2N の通信性能に影響を与えるかを以下の 4 項目を測定することから明らかにする．

(1) N2N（直接通信）暗号化なし：edge が supernode を介することなく，かつ暗号化処理せずに互いに直接通信する仮想ネットワークを介したサイト A のフロントエンドと物理計算機（FA，MA）上の仮想計算機間の通信性能
(2) N2N（直接通信）暗号化あり：edge が supernode を介することなく暗号化処理したデータを互いに直接通信する仮想ネットワークを介したサイト A，サイト B のフロントエンド間（FA，FB）上の仮想計算機間の通信性能
(3) N2N（supernode 経由）暗号化なし：edge が supernode を介して暗号化処理せずにデータを互いに直接通信する場合のサイト A，サイト B の物理計算機間（MA，MB）上の仮想計算機間の通信性能
(4) N2N（supernode 経由）暗号化あり：edge が supernode を介して暗号化処理したデータを互いに直接通信する場合の MA，MB 上の仮想計算機間の通信性能

通信性能の測定結果を，**図 11** に示す．物理ネットワークで 941.4 Mbps の通信性能が得られる場合，N2N を通すと，(1) の結果より，supernode を経由せず，かつ暗号化処理を行わない場合は，540.0 Mbps の通信性能が利用可能である．つまり，約 43%性能が低下していることが分かる．(1)，(3) の性能差，(2)，(4) の性能差から，supernode を経由した場合はさらに約 50%の通信性能になることが分かる．これは edge の送受信するパケットが supernode を中継する際に，カーネル空間からユーザ空間の間を 1 往復する処理が余計にかかることから生じる．(1)，(2) の性能差，(3)，(4) の性能差から，パケットの暗号化処理により，さらに約 42%程度通信性能が低下していることが分かる．したがって，supernode を経由し，かつ暗号化処理を行った場合は，N2N の通信は物理ネットワークから比べて通信性能は約 83%低下することが分かる．

4.2.3　アプリケーションに適用した際の性能評価

次に，実際に分散計算アプリケーションを構築した仮想クラスタ上でサイト間のネットワーク遅延時間を増加させながら実行し，計算リソース間のネットワーク遅延時間がどの程度アプリケーションに影響を与えるか，評価した．分散計算アプリケーションとしては，ジョブの実行中に各仮想計算機間のネットワーク通信量が少ない，パラメータスタディ型のアプリケーション DOCK [18] を用いた．DOCK は，薬物候補となる化合物を探索する分散計算アプリケーションであり，MPI を用いて一連の化合物ごとに複数の計算機へ分散して処理できるように実装されている．評価では，DOCK を用いてあるターゲットとなるタンパク質に対して，400 個の化合物の Docking シュミレーションを行ったときに要した処理時間を計測した．

図 12 に示すように，DOCK のようにノード間で通信

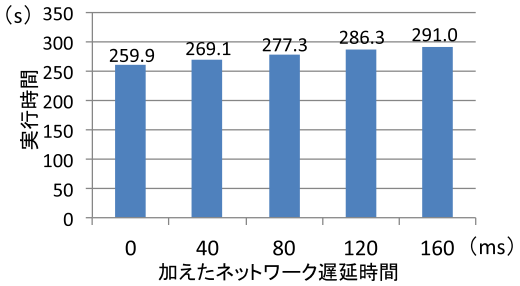

図 12　DOCK の実行時間
Fig. 12　Execution time of DOCK.

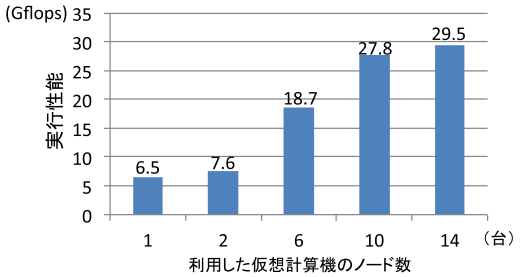

図 13　HPL の実行性能
Fig. 13　Computing Performance of HPL.

があまり発生しない分散計算アプリケーションの場合は，たとえノード間の通信が 40 ms 増加したとしても，アプリケーションの実行性能にはたかだか 2，3%ほどしか影響を与えない．したがって，我々の提案したマルチサイト仮想クラスタ構築システムが構築する仮想クラスタが実際の WAN 環境において構築されても，アプリケーションの実行性能には大きく影響することなく，実行できることが示された．

一方で，各計算機間で通信が発生するような計算の評価を HPL（High-Performance Linpack）[19] を用いて実施した．**図 13** は挿入遅延が 0 ms の場合で，利用する計算ノード数を変化させながら，そのときの実行性能を計測した結果である．この結果から，ネットワークを利用しない 1 ノードで計測を行った場合の性能は約 6.5 Gflops であるが，利用するノード数を増やしていっても性能はスケールせず，すぐに頭打ちになっていることが分かる．

5.　関連研究

現在，複数サイトの計算リソースを集約し，単一の計算環境を構築する研究はいくつか進められつつある [6], [20]．WOW [21] では，P2P ベースのミドルウェア IPOP を用いて複数サイトの仮想計算機を集約することで，広域での分散計算環境を実現している．IPOP は，ユーザの利用するアプリケーションに対して，仮想ネットワークデバイスである tap を提供する．この tap を介して，Condor [22] などのジョブ管理ソフトウェアが用意するバッチキューにジョブを投入することで，IPOP の構築する仮想ネットワークを介して各計算リソース上でジョブを実行する．この論文では，IPOP による仮想ネットワーク構築方法が提案されているのみであり，ジョブ管理ソフトなどを含めたアプリケーションの実行環境の構築までを行うツールを提供しておらず，ユーザビリティは低い．また，tap デバイスがユーザの目に触れる形で提供されており，透過性に関しても低いため，ユーザは仮想ネットワークを意識して利用する必要がある．

一方で，我々と同様に，ユーザビリティを考慮してクラスタ構築ツールである Rocks をベースにしたマルチサイト仮想クラスタ構築システムも提案されている [23]．このシステムでは，各サイトに設置されたゲートウェイノード間に VPN を配備し，サイト内では VLAN による仮想ネットワークを構築し，サイトをまたぐ独立した単一のクラスタ構築を実現している．構築された仮想ネットワークを介した PXE ブートサーバ機能によるインストールも実現し，自動的な仮想クラスタ構築が可能にしている．しかし，このシステムでは，個々の仮想クラスタ配備のために異なる組織間で VPN を構築したり，VLAN の設定を調整したりする必要がある．したがって，オンデマンドかつ柔軟な仮想クラスタを提供することはできない．

我々のシステムは，ユーザの必要な計算リソースに応じて，各サイトに位置する計算機間で動的にオーバレイネットワークを配備することで，オンデマンドな仮想クラスタの提供を実現する．また，既存の仮想クラスタ構築システムを拡張する形でシステムを構築することで，研究者の仮想クラスタ利用にかかる負担を最小限にとどめている．

6.　まとめ

本論文では，オーバレイネットワークを既存のクラスタ構築システムを統合し，複数サイトのリソースを柔軟に割り当て，単一のクラスタを構築するマルチサイト仮想クラスタ構築システムを提案した．システムを実現するために，Rocks の仮想クラスタ構築機能を変更することなく，複数サイトの仮想計算機からなる仮想クラスタを構築可能な MVC コントローラを設計，実装した．MVC コントローラにより，研究者らは通常，Rocks 上で仮想クラスタの構築を行う方法と同様のコマンドラインインタフェースを用いて，複数サイトのリソースを用いた仮想クラスタを構築することができる．

構築したマルチサイト仮想クラスタ構築システムの評価を行うために，実際にバイオサイエンス分野で用いられている創薬シミュレーションを行う分散計算アプリケーションを実行し，その計算性能を測定した．その結果，ネットワーク通信量の少ないパラメータスタディ型の分散計算アプリケーションに対して，計算性能にほとんど影響を与えず，実行が可能であることが確認された．

7. 今後の課題

今後の課題としては，3.1 節で述べた提案システムのセキュリティ機構および 3.4.2 項で述べた MAC アドレスの重複の問題に加えて，提案したシステムのスケーラビリティ，運用上および利用上の問題点，適用可能なアプリケーションに関する 3 点について議論が必要である．

7.1 スケーラビリティに関して

まず，本システムのスケーラビリティを確保するためには N2N のアーキテクチャを改善する必要がある．本研究が用いた N2N は NAT の背後にある edge どうしが通信を行う際には必ず supernode を介して通信を行うため，NAT の背後の edge 台数が増えると supernode が通信のボトルネックとなることが予想される．そのため，NAT 背後の edge の通信を必ず supernode を経由させるのではなく，グローバル IP を保持した近隣の edge を経由して通信をさせるなど，通信を分散させる仕組みが必要と考えられる．また，評価結果から分かるとおり，N2N が採用する tap デバイスや N2N 自身のアーキテクチャに起因するオーバヘッドが非常に大きい．近年では独自のカーネルモジュールとして実装することでオーバヘッドを最小化した仮想ネットワークデバイスなどの研究もされており [24], [25]，それらを活用したり，N2N のアーキテクチャを改善したりすることによる高遅延環境下でのパフォーマンス向上への取り組みをする必要があると考えられる．

7.2 運用上，利用上の問題点

マルチサイト仮想クラスタを構成する仮想計算機の数が数百台から数千台になった場合においては，サイト間の遅延時間の違いやノード数による通信量の増加により，N2N によるブロードキャスト通信が機能しないといった運用上の問題が発生すると考えられる．この問題に関しては，ブロードキャスト通信を高遅延環境下で機能するように，P2P ネットワーク上でサイトを越えたブロードキャスト通信を行わないように独自の実装を行う必要があると考えられる．

一方で，提案したマルチサイト仮想クラスタを構成する仮想計算機間のネットワーク遅延および通信帯域が異なるネットワークが，ユーザにとって論理的に単一のネットワークとして見えるため，利用上問題になると考えられる．この問題に対しては，N2N のオーバレイネットワークから物理トポロジに関する情報を取得し，アプリケーションの実行に適用するための API をユーザに提供する必要があると考えられる．

7.3 適用可能なアプリケーションに関して

本研究では，アプリケーションへの適用評価として通信をあまりともなわない DOCK のみを用いたが，既存の他のアプリケーションへの適用検証を進める必要があると考えられる．本研究が実現したマルチサイト仮想クラスタは論理的には単一のクラスタに見えるが，そのネットワークは高遅延・低帯域である．このような環境下でも，実用的に動作するアプリケーションや，動作しないアプリケーションの例や，実用的に動作させるためのチューニング方法を分析する必要があると考えられる．

謝辞　本研究の一部は財団法人国際コミュニケーション基金（現 KDDI 財団）の助成を受けた．本研究の一部は科研費（No.22700052，23700058）の助成を受けた，ここに記して謝意を示す．

参考文献

[1] Foster, I. and Kesselman, C.: Globus: a Metacomputing Infrastructure Toolkit, *The International Journal of Supercomputer Applications and High Performance Computing*, Vol.11, No.2, pp.115–128 (1997).

[2] Natrajan, A., Nguyen-Tuong, A., Humphrey, M.A., Herrick, M., Clarke, B.P. and Grimshaw, A.S.: The Legion Grid Portal, *Concurrency and Computation: Practice and Experience*, Vol.14, No.13-15, pp.1365–1394 (2002).

[3] Tanaka, Y., Nakada, H., Sekiguchi, S., Suzumura, T. and Matsuoka, S.: Ninf-g: A reference implementation of rpc-based programming middleware for grid computing, *Journal of Grid Computing*, Vol.1, No.1, pp.41–51 (2003).

[4] Ruth, P., McGachey, P. and Xu, D.: Viocluster: Virtualization for Dynamic Computational Domains, *IEEE International Conference on Cluster Computing*, pp.1–10 (Sep. 2005).

[5] Marshall, P., Keahey, K. and Freeman, T.: Elastic site: Using clouds to elastically extend site resources, *The 10th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pp.43–52, IEEE Computer Society (May 2010).

[6] Foster, I., Freeman, T., Keahy, K., Scheftner, D., Sotomayer, B. and Zhang, X.: Virtual Clusters for Grid Communities, *IEEE International Symposium on Cluster Computing and the Grid*, pp.513–520, IEEE (May 2006).

[7] Papadopoulos, P.M., Katz, M.J. and Bruno, G.: NPACI Rocks: Tools and techniques for easily deploying manageable Linux clusters, *Concurrency and Computation: Practice and Experience*, Vol.15, No.7-8, pp.707–725 (2003).

[8] Takamiya, Y., Sakae, Y., Yamagata, I. and Matsuoka, S.: A flexible configuration and packaging method for cluster installers, IEIC Technical Report (Institute of Electronics, Information and Communication Engineers), Vol.105, No.225, pp.19–24 (2005).

[9] Mugler, J., Naughton, T., Scott, S.L., Barrett, B., Lumsdaine, A., Squyres, J.M., des Ligneris, B., Giraldeau, F. and Leangsuksun, C.: OSCAR Clusters, *Proc. Ottawa Linux Symposium* (*OLS'03*), Ottawa, Canada (July 2003).

[10] Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I. and Warfield, A.: Xen and the art of virtualization, *ACM SIGOPS Operating Systems Review*, Vol.37, No.5, pp.164–177 (2003).

[11] Deri, L. and Andrews, R.: N2N: A Layer Two Peer-to-Peer VPN, *2nd International Conference on Autonomous Infrastructure, Management and Security: Resilient Networks and Services*, pp.53–64, Springer (July 2008).
[12] Ganguly, A., Agrawal, A., Boykin, P.O. and Figueiredo, R.: IP over P2P: Enabling Self-Configuring Virtual IP Networks for Grid Computing, *The 20th IEEE International Parallel and distributed processing Symposium*, pp.46–49, IEEE Computer Society (Apr. 2006).
[13] Jiang, X. and Xu, D.: VIOLIN: Virtual Internetworking on Overlay Infrastructure, *Parallel and Distributed Processing and Applications*, 3358, pp.937–946 (2005).
[14] Tsugawa, M. and Fortes, J.A.B.: A virtual network (ViNe) architecture for grid computing, *The 20th IEEE International Parallel and Distributed Processing Symposium*, p.10, IEEE (2006).
[15] Xen roll users guide, available from ⟨http://www.rocksclusters.org/roll-documentation/xen/5.4/⟩.
[16] Yonan, J.: OpenVPN - an open source SSL VPN solution, available from ⟨http://www.openvpn.net/⟩.
[17] Jones, R.: The netperf homepage, available from ⟨http://www.netperf.org/⟩.
[18] Moustakas, D.T., Lang, P.T., Pegg, S., Pettersen, E., Kuntz, I.D., Brooijmans, N. and Rizzo, R.C.: Development and validation of a modular, extensible docking program: Dock 5, *Journal of Computer-Aided Molecular Design*, Vol.20, No.10-11, pp.601–619 (2006).
[19] Petitet, A., Whaley, R.C., Dongarra, J. and Cleary, A.: Hpl-a portable implementation of the high-performance linpack benchmark for distributed-memory computers, available from ⟨http://www.netlib.org/benchmark/hpl/⟩.
[20] Rezmerita, A., Morlier, T., Néri, V. and Cappello, F.: Private Virtual Cluster: Infrastructure and Protocol for Instant Grids, *The 12th International Conference on Parallel Computing, Euro-Par*, pp.393–404, Springer (Aug./Sep. 2006).
[21] Ganguly, A., Agrawal, A., Boykin, P.O. and Figueiredo, R.: WOW: Self-Organizing Wide Area Overlay Networks of Virtual Workstations, *The 15th IEEE International Symposium on High Performance Distributed Computing*, pp.30–42, IEEE (June 2006).
[22] Litzkow, M.J., Livny, M. and Mutka, M.W.: Condor-a hunter of idle workstations, *8th International Conference on Distributed Computing Systems*, pp.104–111, IEEE (June 1988).
[23] Hirofuchi, H.T., Yokoi, T., Ebara, T., Tanimura, Y., Ogawa, H. and Nakada, H.: Multi-Site Virtual Cluster: A User-Oriented, Distributed Deployment and Management Mechanism for Grid Computing Environments, *The 4th IEEE/IFIP International Workshop on End-to-end Virtualization and Grid Management*, pp.203–216 (Sep. 2008).
[24] Rizzo, L., Carbone, M. and Catalli, G.: Transparent acceleration of software packet forwarding using netmap, *The 31st Annual IEEE International Conference on Computer Communications*, pp.2471–2479 (Mar. 2012).
[25] Pfaff, B., Pettit, J., Koponen, T., Amidon, K., Casado, M. and Shenker, S.: Extending Networking into the Virtualization Layer, *8th ACM Workshop on Hot Topics in Networks* (Oct. 2009).

多田 大輝　（学生会員）

2011 年大阪大学工学部電子情報工学科卒業（学士）．大阪大学大学院情報科学研究科マルチメディア工学専攻へ入学．現在，博士前期課程学生．分散システムを支えるミドルウェアおよび仮想ネットワークに関する研究に興味を持ち，研究を行っている．IEEE 学生会員．

市川 昊平　（正会員）

2008 年大阪大学大学院情報科学研究科博士課程修了．博士（情報科学）．関西大学ソシオネットワーク戦略研究機構博士研究員，大阪大学情報基盤本部助教を経て，2012 年より奈良先端科学技術大学院大学情報科学研究科准教授．この間（2005 年 12 月から 2006 年 2 月まで）米国カリフォルニア大学サンディエゴ校客員研究員．分散システムに関する研究に取り組み，分散システムを支えるミドルウェアからアプリケーション応用までの幅広い研究開発を行っている．人工知能学会，IEEE 各会員．

伊達 進　（正会員）

2002 年大阪大学大学院工学研究科情報システム工学専攻博士後期課程修了．工学博士．2002 年大阪大学大学院情報科学研究科助手．この間（2005 年 2 月から 9 月まで），米国カリフォルニア大学サンディエゴ校客員研究員．2005 年大阪大学大学院情報科学研究科特任助教授．2007 年大阪大学大学院情報科学研究科特任准教授．2008 年大阪大学サイバーメディアセンター准教授，現在に至る．

阿部 洋丈　（正会員）

2004 年筑波大学大学院博士課程工学研究科電子・情報工学専攻修了．科学技術振興機構 CREST 研究員，豊橋技術科学大学情報工学系助教を経て，2010 年より大阪大学サイバーメディアセンター助教．システムソフトウェア，特に分散システムやコンピュータセキュリティ等の研究に従事．情報処理学会平成 16 年度山下記念研究賞，情報処理学会平成 16 年度論文賞受賞．博士（工学）．日本ソフトウェア科学会，IEEE，ACM 各会員．

下條 真司　（正会員）

1958 年生．1986 年大阪大学大学院基礎工学研究科物理系専攻博士後期課程修了．工学博士．1986 年大阪大学基礎工学部助手．1989 年大阪大学大型計算機センター講師．1991 年大阪大学大型計算機センター助教授．この間（1996 年 2 月から 9 月まで），米国カルフォルニア大学アーバイン校客員研究員．1998 年大阪大学大型計算機センター教授．2000 年大阪大学サイバーメディアセンター教授（副センター長），現在に至る．