Construction of Virtual Cluster System and Evaluation of Performance

Masahiro NAKAO* , Tomoyuki HIROYASU** , Mitsunori MIKI*** , Masato YOSHIMI***

(Received October 26, 2009)

Because of high-availability and running-cost reduction, a virtual cluster provided by virtualized calculation resources becomes popular to utilize. However, a setting cost of a virtual cluster is very large and a processing ability of a virtual cluster is lower than that of a cluster without virtualization (no-virtual cluster). In this paper, we introduce how to construct a virtual cluster simply by using virtualization software "Xen" and cluster auto-setup tool "DCAST". We also draw a comparison by using HPL benchmark between a virtual cluster and a no-virtual cluster. The result shows that the performance of a virtual cluster is about 1.1% on 1 process and about 11.9 % on 32 processes lower than that of no-virtual cluster. Moreover, we evaluate a network performance by using Netperf benchmark. The result shows that the performance of Request/Response ability of a virtual cluster is about twice as low as that of no-virtual cluster. In conclusion, a parallel application where there is no frequent network communication can be performed on a virtual cluster. On the other hand, a parallel application which needs frequent communication cannot be performed on a virtual cluster.

Key words ： virtualization, PC cluster system, benchmark, parallel computation

キーワード ： 仮想化，PC クラスタ，ベンチマーク，並列計算

仮想クラスタの構築と性能評価

中尾 昌広，廣安 知之，三木 光範，吉見 真聡

1. はじめに

1 台の PC に複数の OS を同時に稼働させることができるプラットフォーム仮想化（以下，仮想化）技術が注目されている．仮想化技術を用いることで，システムの導入・管理コストを削減できると期待されている．

さらに仮想化技術を PC クラスタやグリッドコンピューティング環境に利用し，計算資源を仮想化して提供する試みも行われている [1, 2]．このように，仮想化された計算資源を 1 つのクラスタとして用いるシステムのことを仮想クラスタと呼ぶ．仮想クラスタの利点を下記に示す．

- 可用性の向上
  メンテナンスなどのため一部のノードを停止させる場合，仮想化技術のライブマイグレーション機能を利用することで，仮想 OS を他のノードに無停止で移動することができる

* Graduate School of Engineering, Doshisha University, Kyoto
Telephone:+81-774-65-6130, Fax:+81-774-65-6019, E-mail:mnakao@mikilab.doshisha.ac.jp
** Department of Life and Medical Sciences, Doshisha University, Kyoto
Telephone:+81-774-65-6932, Fax:+81-774-65-6019, E-mail:tomo@is.doshisha.ac.jp, hyokouch@mail.doshisha.ac.jp
*** Faculty of Science and Engineering, Doshisha University, Kyoto
Telephone:+81-774-65-6930, Fax:+81-774-65-6796, E-mail:mmiki@mail.doshisha.ac.jp

- コスト削減
  ハードウェアを新規購入した際のイニシャルコストの削減，およびシステム移行の容易化が実現できる．また，負荷の低い時間帯は少数のノードで仮想 OS を稼働させ，他のノードは停止，もしくはスタンバイ状態にすることで消費電力を節約できる

仮想化技術を用いて計算資源を提供する場合，仮想化ソフトウェアが動作の仲介を行うため，仮想化を用いない場合と比較して計算の実行速度は遅くなるという問題点がある．また，仮想クラスタを構築する場合，複数台の PC に対して手動で仮想環境の設定を行うには作業者の負担が大きいため，仮想クラスタ構築の自動化が求められている．

そこで本稿では，オープンソースの仮想化ソフトである Xen[3] と，自動 PC クラスタ構築ツールである DCAST（Dynamic Cluster Auto Setup Tool）[4] を用いた少ない手順で仮想クラスタを構築する方法を紹介する．さらにベンチマークプログラムである HPL（High-Performance Linpack Benchmark）[5] を用いて，仮想クラスタの性能評価を行う．この性能評価により仮想クラスタの計算速度の低下率を測定し，またその特性について検証を行う．

2. プラットフォーム仮想化技術

仮想化とは，ハードウェア上で仮想計算機をソフトウェアによって提供することを指す．ユーザは提供された仮想計算機上に通常の OS をインストールすることで，仮想 OS を通常の OS と同様に扱うことができる．

仮想化技術は，ホスト OS 型と仮想マシンモニタ型に大きく分けることができる．ホスト OS 型とは，通常の OS の上で仮想 OS が動作する形態を指す．通常の OS も利用できるという利点はあるが，仮想 OS の処理のオーバーヘッドは大きい．ホスト OS 型の実装としては，VMware Server，QEMU などがある．

仮想マシンモニタ型（ハイパーバイザ型と呼ばれることもある）とは，仮想 OS がハードウェア上で直接動作する形態を指す．通常の OS を通さないため，ホスト OS 型と比較してパフォーマンスが高く，またハードウェアの管理を柔軟に行うことができる．仮想マシンモニタ型の実装としては，VMware ESX Server，Xen などがある．

仮想マシンモニタ型は，仮想 OS に対する修正の有無により完全仮想化と準仮想化に分けることができる．完全仮想化とは，仮想 OS のすべての命令を解析し，仮想ソフトウェア上で完全な計算機環境を再現する技術である．完全仮想化を用いるためには，CPU が完全仮想化に対応している必要がある．

準仮想化とは，仮想 OS に修正を加えることで，仮想 OS のすべての命令を解析することなく，仮想 OS を動作させる技術である．準仮想化の利点は，仮想 OS のオーバーヘッドを最小限に抑えることができるため，完全仮想化と比較して性能が高い点である．しかしながら，準仮想化で用いることが可能な OS は仮想化ソフトウェアがサポートしている OS に限定される．

3. 仮想クラスタの構築

3.1 構築概要

仮想クラスタ構築に用いるハードウェアには，同志社大学知的システムデザイン研究室が所有する Supernova クラスタを用いた．Supernova クラスタは 256 台のノードで構成され，2003 年の TOP500[6] において 93 位となった大規模計算システムである．今回は，256 台中の 16 台のノードを用いた．Table 1 に Supernova クラスタのハードウェアとソフトウェアの仕様を，Fig. 1 に Supernova クラスタの外観を示す．

仮想化ソフトウェアには Xen を用いる．その理由は，Xen はオープンソースソフトウェアであるため導入費用が不要であり，また処理のオーバーヘッドが比較的少ない仮想マシンモニタ型であるからである．Xen は完全仮想化，準仮想化のどちらにも対応しているが，今回は仮想化された環境を計算資源として利用したいため，準仮想化を用いる．

Table 1. Specifications of each host of SuperNova.

CPU	AMD Opteron 1.8GHz × 2
Memory	PC2700 Registered ECC 2GB
NIC	Gigabit Ethernet
NIC driver	Broadcom Tigon3
Switch	NETGEAR GS524T
Cable	Category 5e
OS	Debian GNU/Linux 4.0 amd64
Kernel	Linux 2.6.16 + Xen patch
Xen	3.0.3
DCAST	3.0.5
HPL	1.0a
Compiler	gcc 4.1.2, gfortran 4.1.2
通信ライブラリ	mpich 1.2.7
行列演算ライブラリ	GotoBLAS2 1.04

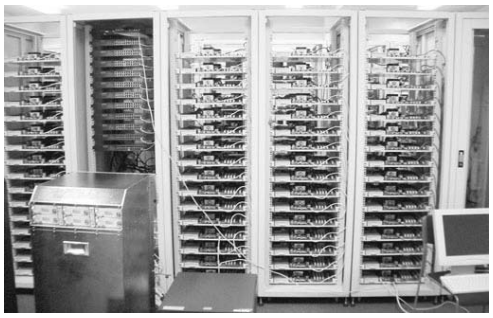

Fig. 1. Supernova Cluster System.

Supernova クラスタの各ノードはデュアルプロセッサであるため，各ノードで 2 つの仮想 OS を稼働させる．すなわち計 32 個の仮想 OS を稼働させる．各ノードは 2GB のメモリを搭載しており，2 つの仮想 OS とその仮想 OS を管理するための OS が動作する．各 OS に割り当てるメモリ量は，各仮想 OS は 768MB，仮想 OS を管理するための OS は 512MB に設定した．

また，仮想 OS 用のイメージファイルは比較的大きい（今回は 1 つの OS あたり 4GB）ため，すべての仮想イメージをファイルサーバに保存する．計算ノードはその仮想イメージをネットワークを通して参照することで，仮想 OS を稼働する．

3.2 DCAST の設定と実行

本節以降は，DCAST を用いた仮想クラスタの構築手順を紹介する．事前準備として，仮想イメージを保存するファイルサーバの構築，ファイルサーバと Supernova の各ノードのネットワークケーブルを用いた接続は完了しているとする．

まず，Supernova のノードの 1 台に OS である Debian GNU/Linux と DCAST をインストールする．このノードをマスタノードと呼ぶ．次に，PC クラスタとして動作するために必要なソフトウェア（コンパイラや並列計算用のライブラリなど），および Xen をインストールし，設定を行う．

DCAST はマスタノードに設定した OS とソフトウェア群を，計算ノードにインストール，および設定を自動で行うことができる．そのため，マスタノードのみに Xen 環境の設定を行うことで，計算ノードに対して仮想環境の設定を行うことができる．

DCAST はネットワークブートを利用したインストーラであるので，構築するノードを識別するために，DCAST の設定ファイルに各ノードの MAC アドレスを記入する必要がある．DCAST はネットワークに接続されたノードの MAC アドレスを収集する機能を提供している．下記にコマンドの例を示す．

```
# ./dcast-collect-mac --eth=（インタフェース名） --number=（構築するノードの数）
```

コマンド終了後に，マスタノード上で DCAST を実行することで，計算ノードに OS と Xen を含むソフトウェア群を自動インストールできる．下記にコマンドを示す．

```
# ./dcast
```

仮想クラスタの構成図を Fig. 2 に示す．Fig. 2 では，一般的な PC クラスタでは存在するサービスである”ルータ”と”他のサービスサーバ”も記述している．今回は性能評価が目的であるため，これらは必要ではないが，実際に運用する際は必要である場合が多い．

3.3 仮想 OS の作成

Debian GNU/Linux では，debootstrap と xen-tools というソフトウェアを利用することで，簡易に仮想 OS

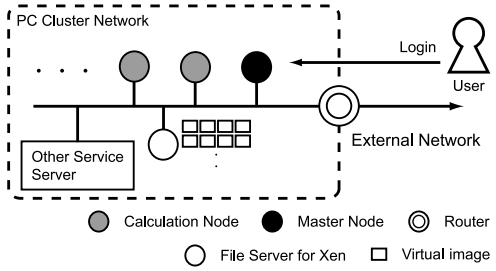

Fig. 2. Composition of Virtual PC Cluster System.

のイメージを作成することができる．下記にコマンドの例を示す．

```
# xen-create-image --ip （IP アドレス）
--hostname （ホスト名）
```

このコマンドで作成した仮想 OS のイメージは，必要最小限のソフトウェアのみインストールされている．そのため，作成された仮想 OS のイメージに対して，PC クラスタとして動作するために必要なソフトウェアをインストール，設定を行う必要がある．しかし，すべての仮想 OS に対してソフトウェアの設定作業を行うことは作業者の負担が大きい．

そこで，1 つの仮想 OS に対してのみソフトウェアの設定を行い，他の仮想 OS には設定した仮想 OS のイメージを利用することを考える．各仮想 OS の固有の設定はホスト名と IP アドレスのみである．そのため，コピーされた仮想イメージに対してホスト名と IP アドレスのみを変更する簡易なスクリプトを作成することで，他の仮想 OS の設定作業を行った．なお，今回の環境で，そのスクリプトを用いて 1 つの仮想 OS イメージを作成する時間は約 180 秒であった．

4. 性能評価

4.1 概要

仮想 OS を用いた場合と仮想化を用いない場合において HPL を用いた性能比較を行う．実験に使用するノードはデュアルプロセッサであるため，各ノードに２つずつ HPL のプロセスを割り当てる．今回は全 16 ノードで実験を行うため，1，2，4，8，16，32 プロセスの 6 通りの計測を行う．計測は各 5 試行行い，その平均値で性能評価を行う．

HPL で用いる行列演算ライブラリである GotoBLAS[7] には，自動的にコア毎（CPU 毎）にスレッドに分けて動作する機能を持つ．しかし，今回は 1 スレッドで HPL を実行させる必要があるため，スレッド機能を無効にした GotoBLAS を用いる．

HPL ではシステムの特性に応じたパラメータの設定を行うことが可能である．今回計測に用いた HPL のパラメータは文献 [8] を参考にし，Table 2 のように設定する．

仮想化を用いない場合の Linux カーネルには Xen のパッチを適用していない Linux 2.6.16 を用いる．

4.2 結果

HPL の結果を Fig. 3 に示す．

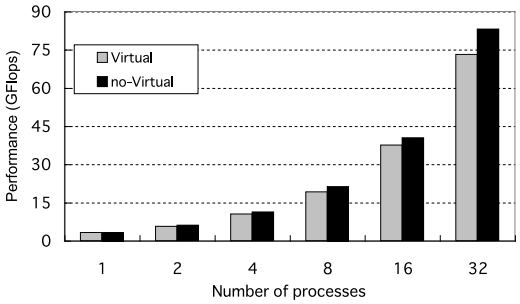

Fig. 3. HPL Result.

Fig. 3 の結果から，1 プロセスの場合は仮想 OS の方が評価値は約 1.1%低く，32 プロセスの場合は約 11.9%低いことがわかった．

5. 考察

前章の結果から，プロセス数が増えるほど，すなわち計算に用いるノード数が増えるほど，評価値の差は大きくなる傾向にあることがわかる．この理由は、仮想 OS のネットワーク性能が仮想化を用いない場合と比較して低いからであると考えられる．

そこで，仮想 OS のネットワーク性能を測定する．並列計算を行う際は，小さいサイズのデータをノード間で交換する場合が多い．そのため，2 ノード間でデータのリクエスト/レスポンス（単位時間あたりに送受

Table 2. The main parameters of HPL.

Number of Processes	1	2	4	8	16	32
N	8973	12690	17947	25381	35895	50763
NB	240					
P，Q	1, 1	1, 2	2, 2	2, 4	4, 4	4, 8
BCAST	1ring Modified					

信されるデータ数）の計測を行う．ネットワークの性能測定用ソフトウェアには Netperf[9] を用いた．データサイズは 1bit とした．測定環境は，異なるノード上の仮想 OS 間，および仮想化を用いない OS 間である．結果を Table 3 に示す．結果は 5 試行の平均値である．

Table 3. Netperf Result.

	Virtual	no-Virtual
Request/Response (Trans./s)	6403	11399

Table 3 の結果より，リクエスト/レスポンス性能は，仮想 OS 間のネットワーク性能は仮想化を用いない場合と比較して，約 44% も低いことがわかった．このことから，仮想クラスタはノード間でデータの交換を多く行うような並列計算には適していないことがわかる．

6. まとめ

本稿では，Xen と DCAST を用いた仮想クラスタの構築方法について述べ，HPL を用いて仮想クラスタの性能評価を行った．性能評価の結果，1 プロセスの計算であれば，仮想 OS と仮想化を用いない OS との性能は同程度であることがわかった．このことから，逐次アプリケーション（例えば，パラメータスイープ等の用途）やネットワークをあまり利用しない並列アプリケーションを仮想クラスタで用いる場合であれば，性能低下はほとんど起こらないと考えられる．

しかしながら，仮想 OS 間のネットワークのリクエスト/レスポンスの性能は仮想化を用いない OS 間と比較して低いことがわかった．そのため，多くのノードを用いた並列アプリケーションを行う場合は，仮想化を用いない場合と比較して性能は低いと考えられる．

参 考 文 献

1) 西村 豪生, 中田 秀基, 松岡 聡, ” 仮想計算機と仮想ネットワークを用いた仮想クラスタの構築”, 情報処理学会研究報告．ハイパフォーマンスコンピューティング, Vol. 1, No 1, pp. 73–78, 2006.

2) 高宮 安仁, 山形 育平, 青木 孝文, 中田 秀基, 松岡 聡, ”ORE Grid：仮想計算機を用いたグリッド実行環境の高速な配置ツール”, 先進的計算基盤システム SACSIS2006, Vol. 1, No. 1, pp. 541–550, 2006.

3) http://www.xen.org/.

4) 中尾 昌広, 廣安 知之, 三木 光範, ” ディスクレスノードとディスクフルノードが混在するクラスタシステム環境をセットアップできるツールの開発”, Transactions of JSCES, pp. 1–9, 2008.

5) HPL A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers, http://www.netlib.org/benchmark/hpl/index.html.

6) TOP500 Supercomputer Sites, http://www.top500.org/.

7) Texas Advanced Computing Center, http://www.tacc.utexas.edu/.

8) 廣安 知之, 渡辺 崇文, 中尾 昌広, 大塚 智宏, 鯉渕 道紘, ”PC クラスタにおける VLAN イーサネットのトポロジの評価”, 情報処理学会論文誌, Vol. 2, No. 3, pp. 1–11, 2009.

9) http://www.netperf.org/netperf/.