クラウドリソースを使用したデータ処理負荷分散のためのミドルウェア開発

豊島詩織[†1]　山口実靖[†2]　小口正人[†1]

高度 IT 社会の進展に伴いデータの管理や IT コストの問題が深刻になっている．より効率的なデータ処理システムが望まれる中，データインテンシブアプリケーションにおいて，手元のクラスタ使用状況を観察し，リソースが不足している場合は外部のクラウドリソースを動的に使用する，クラスタのスケーラブルな運用のためのミドルウェア構築を目指す．クラスタを仮想マシン PC クラスタとし，そのネットワークストレージに SAN(iSCSI) を導入することで，サーバとストレージ間の広域環境における通信を低コストで実現できる他、動作中のアプリケーションの状態を維持したまま仮想マシンを別のノード上へマイグレーションしたり，サーバとストレージの位置を分離したリソースの柔軟な調達が可能となる．本稿においてクラウドには商用のクラウドサービスである Amazon EC2 を用い，クラスタの負荷状況に応じてジョブを振分けるミドルウェアを構築した

Development of middleware for data processing load distribution using cloud computing resource

Shiori TOYOSHIMA,[†1] Saneyasu YAMAGUCHI[†2] and Masato OGUCHI[†1]

In recent years, Data management and IT cost has become serious problem as development of advanced IT society. In the situation that more efficient data processing system is required, we aim to construct a middleware for scalable operation of cluster. Scalable resource management is achieved by monitoring resource usage of own local cluster and insufficient resources are acquired dynamically from cloud computing. We have proposed a virtual machine PC cluster in which virtualization is applied to a PC cluster that uses a general-purpose personal computer for each node. In addition, we have used IP-SAN for storage access which can realize long-distance connection at low cost over a high-latency network. In this paper, we have constructed middleware that sorting job based on the result of monitoring in local cluster. using commercial cloud service Amazon EC.

1. はじめに

高度 IT 社会の進展によりコンピュータシステムにおいて利用可能なデータの量が増大している近年，よりスケーラブルなリソース管理の実現が望まれている．そこで期待が高まっているのがクラウドコンピューティングである．クラウドコンピューティングにおいてユーザはリソースを利用するだけであるため，システムの運用コストが大幅に削減できる．そして必要なときに必要な分だけリソースを利用することが可能となる．全てのシステムをクラウドで構築することも考えられるが元々自前のシステムを所有していることが考えられる他，運用面やコストの面でのリスクが懸念される．そのためクラウドコンピューティングのメリットを活かし，使用しているクラスタのシステム状況をモニタリングし，急激に大量のキャパシティが必要となる場合に外部のクラウドリソースへ負荷分散するミドルウェアの構築を目指す．

ローカル環境における負荷が高い場合にネットワーク越しの遠隔リソースへ負荷を分散すること自体は，グリッドコンピューティングなどの枠組みで実現できるため，新しい考え方ではない．しかし遠隔リソースとしてクラウドを利用した場合，以下のような点で従来とは異なる特徴がある．まずユーザのニーズに応じてリソースを大幅に増減できることが期待される．またセキュリティポリシにより社外にデータを置けないユーザでも、データは社内に保存したまま、計算能力だけクラウドから借りることが可能になる．

処理すべき情報量が増えると単なる計算処理だけではなくデータインテンシブアプリケーションの動作が重要となる．そのためシステム評価にはデータベースベンチマークの pgbench を使用した．科学技術計算など計算処理が中心となるアプリケーションの場合は，各ノードの CPU 負荷により判断して適切な負荷分散を行うことができるが，データインテンシブアプリケーションの場合，CPU は I/O 待ちとなっていることが多く，CPU 負荷では適切な判断が行えない．そこで本研究では負荷の指標として，ディスクアクセス量を用いた．このようにリソースとして伸縮性の高いクラウドを用いている点，そしてデータインテンシブアプリケーションを対象負荷として用いている点が，本研究の特徴である．

†1 お茶の水女子大学
Ochanomizu University
†2 工学院大学
Kogakuin University

2. 研究背景

2.1 仮想マシン PC クラスタ

IT リソースを効率的に活用するために仮想化が用いられる．自前のクラスタを効率よく運用するためクラスタシステムとしてワーカノードに仮想マシンを配置した PC クラスタを使用した．

仮想化ソフトには Xen[1] を使用した．Xen は図 1 に示すように複数の OS を動かす為の基盤となるプラットフォームのみを提供することで仮想マシンのオーバヘッドを少なくし，物理マシンに近い性能が発揮されるよう工夫されている．オープンソースとしては非常に高性能で，現在ビジネスユースにおいても用いられるようになっている．

仮想マシンモニタが仮想化のための土台となり，その上で Domain と呼ばれる仮想マシンが動作する．ホスト OS として動いているものが Domain0，ゲスト OS として動いているものが DomainU で，Domain0 は実ハードウェアへのアクセスやその他のドメインを管理する特権を持つ．

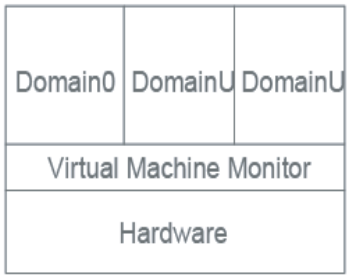

図 1 Xen の構造

2.2 IP-SAN

ストレージネットワークには SAN を使用した．近年情報システムにおいて処理されるデータの量が膨大になってきたことから，ネットワークストレージ技術が発展し，PC クラスタのストレージに SAN を用いることが多くなっている．SAN は，分散したストレージをネットワークで統合し，ストレージの集中管理とディスク資源の効率的な活用を可能にする．特に IP-SAN は Ethernet インタフェースと TCP/IP 対応ネットワークさえあれば導入でき，また専用網も含め広範囲に IP ネットワークのインフラが整備されているため長距離接続が可能で，クラウドコンピューティングなどの枠組を用い，計算機リソースのアウトソーシングに利用されることが期待される．

IP-SAN のプロトコルとしては iSCSI (Internet Small Computer System Interface)[2] を使用した．iSCSI の構造を図 2 に示す.iSCSI は SCSI コマンドを TCP/IP パケットの中にカプセル化することでブロックレベルのデータ転送を行う.Gigabit Ethernet/10Gigabit Ethernet が広く普及していくであろうことを考慮すると，IP-SAN をバックエンドに持つ PC クラスタ多くが使用されるようになると考えられる．

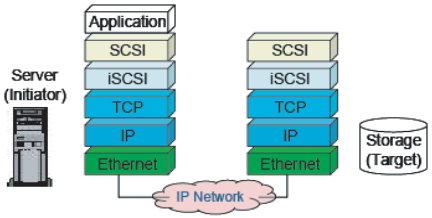

図 2 iscsi の階層構造

文献[3] は IP-SAN のプロトコルである iSCSI を使用し，クライアント-計算ノード間のフロントエンドと計算ノード-ストレージ間のバックエンドネットワークを同一の IP ネットワークに統合した IP-SAN 統合型 PC クラスタを構築している．

IP ネットワークを使用していることや，フロントエンドとバックエンドに同じネットワークを使用することから構築および管理コストの削減が期待されるが，ノード間通信とストレージアクセスで同じネットワークリソースを使用するため互いに衝突し，性能が低下する可能性が懸念された．このシステム上で並列データマイニングの HPA (Hash Partitioned Apriori[4] と PFP (Parallelized FP-growth)，科学技術計算の mpiBLAST[5] を動作させ，IP-SAN 統合型 PC クラスタの詳しい振舞を明らかにしている．評価を行なった範囲では iSCSI のネットワークを統合してもネットワークバウンドにはならないということが分かっている．

2.3 リモートサイトのリソース利用時のデータの配置

本研究では自身のクラスタの負荷を一定間隔のモニタリングにより観察し、負荷が大きい場合は外部のクラウドコンピューティングリソースへ負荷分散を行うミドルウェアを構築し

システム評価を行う．負荷分散のためのクラウドコンピューティングとしては商用のクラウドサービスである Amazon EC 2[6] を用いる．

ただし本研究では主にデータインテンシブアプリケーションの実行を想定しているため，リソースとしては，計算処理を行うサーバだけでなく，データおよびストレージをどのように取り扱うか考えなければならない．

リモートサイトのサーバに計算処理の一部をマイグレートする場合，処理を行うデータの配置については，大きく分けて以下の 3 つのケースが考えられる．まず 1 つめは (a) 処理に必要なデータについても処理のマイグレート時にオンデマンドでリモートサイトにコピーする場合、2 つめは (b) ローカルクラスタでの処理中に遠隔バックアップが行われており、あらかじめリモートにデータが存在する場合、3 つめは (c) セキュリティポリシや、データが巨大すぎるなどの理由でリモートにデータを置くことができない場合である．

1 つめ (a) については一般にリモートへのデータ転送はスループットが低いため，この方式はデータ量が多い場合には，性能低下を招く可能性がある．ただしコピーが終わったらリモートサイトにおいてデータへ高速アクセスが可能となるため，データ量が少ない場合やアプリケーション全体の処理時間が長い場合には有効であると考えられる．

2 つめ (b) の場合にはデータアクセスに関する制約が無くなるため，積極的にリモートサイトのリソースを利用した方が性能面では有利になると考えられる．

3 つめ (c) の場合には，計算処理のみリモートサイトのリソースを利用しながら，データはローカルに置きリモートサイトからアプリケーション実行時にアクセスする事が考えられる．例えば Google 社が提供するクラウドサービスである Google Apps においては Secure Data Connector[7] という仕組みが提供されており，これを利用するとクラウドとローカルサイトの間にセキュアトンネルが構築され，クラウドからローカルサイトのデータに対し，安全にアクセスを行う事ができるようになる．このケースの場合には，リモートサイトから計算処理サーバだけ借りれば良く，容易に負荷分散のマイグレーションが実現できる．ただしリモートサイトとローカルクラスタの間の通信性能が全体の実行性能に大きな影響を与えるため，ネットワークの帯域幅が小さい場合やデータアクセス頻度が高いアプリケーションの場合には，性能の大幅な低下が予想される．また，リモートサイトからローカルのストレージへのアクセスには制限がある場合もあり，これらの問題をクリアしなければならない．

このようにリモートサイトのリソースを利用して負荷分散を行う事を考える際，データインテンシブアプリケーションの場合には，データをどのように扱いどこに置いて実行するか考える事が重要である．さらに上記の 1 つ目と 2 つ目のケースのように，データをリモートサイトに配置して計算処理を実行する場合には，リモートサイトにおけるストレージについても，何台用いデータをどのように配置すべきかについて検討する必要がある．

3. Amazon EC2 の性能測定

本研究ではまず Amazon EC2 の基本性能の測定を行い、ローカルクラスタのマシンとどの程度性能差があるのかを調べた．Amazon EC2 はインターネット上のサーバレンタルサービスである．バックエンドではサーバ仮想化技術が使われているため，自分で作った環境の OS イメージをまるごとバックアップしたり，さらにその環境のイメージを複製して同一の環境を持つサーバを複数稼動させたりすることが可能となる．また，急に負荷が増えた際なども，イメージファイルさえ用意しておけば，数分で新しいマシンを起動できるためシステムに合わせて柔軟な運用が可能である．

EC2 では数種類のスペックの仮想マシンが提供されており，全部で 5 種類のインスタンス・タイプの中から選択することができる．本実験においては 32bit コンピュータである「スタンダードプラン」の small と,「High CPU プラン」の High-CPU Medium という 2 種類のスペックの性能を測定した．それぞれのスペックは表 1 の通りとなっている．ECU(EC2 Compute Unit) は Amazon が定義した CPU リソース単位量であり，1ECU は 1.0-1.2GHz の 2007opteron または 2007Xeon プロセッサ相当と定義されている[6]．ローカルクラスタのマシンは仮想マシンに対するメモリ割当てが 1 台あたり 1GByte のマシンを使用した．CPU は IntelXeon3.6GHz である．

表 1 本研究で使用する EC2 のスペック

Instance	CPU	memory	storage	platform
Small	1ECU × 1	1.7GByte	160GByte	32bit
High-CPUMedium	2.5ECU × 2	1.7GByte	350Gbyte	32bit

3.1 Network Throughput

まずローカルクラスタの仮想マシンの DomainU 間と，EC2 の 2 種類のインスタンス間のネットワークスループットをそれぞれ測定した (図 3)．DomainU のメモリが 1GByte, EC2 におけるメモリは 1.7GByte であることを考えても，EC2 間のスループットはローカルクラスタのマシンに比べ低いことが分かる．

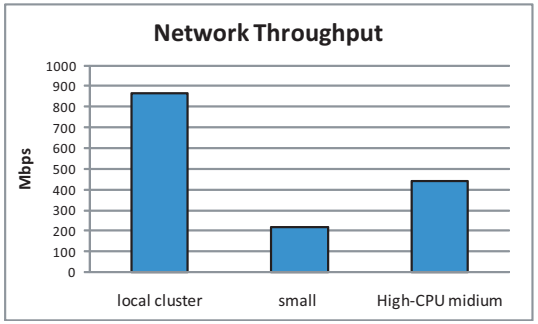

図 3 Network Throughput

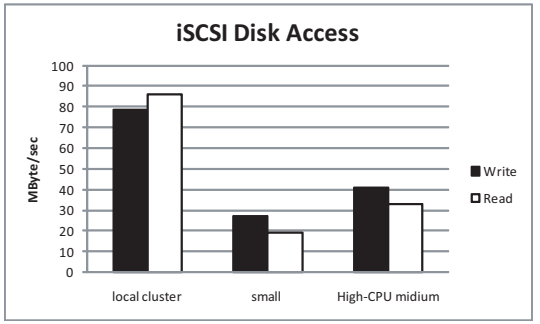

図 5 iSCSI disk

3.2 Disk Access

データインテンシブアプリケーションの性能にはストレージアクセス性能の評価が不可欠となるため，ローカルクラスタの DomainU，EC2 において local disk，iscsi disk のアクセス性能を測定した (図 4，5).

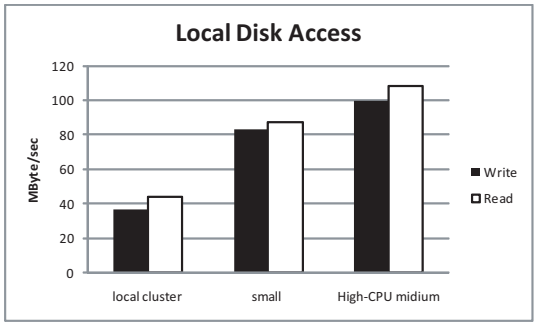

図 4 local disk

local ディスクアクセスについては実クラスタより EC2 インスタンスの性能がよいことが分かる．iSCSI ディスクアクセスについてはかなりの性能差がみられた．これは図 3 におけるネットワーク性能の差によるものと思われる．

3.3 クラスタとの通信

次に EC2(High-CPU Medium) とクラスタ間の Network Throughput を測定した (図 6). この結果から性能がよいと考えられるクラスタが送信側の場合の性能が高いことが分かる．ただしいずれの場合もクラスタ内の通信などと比較すると，スループットの値は極めて低い．

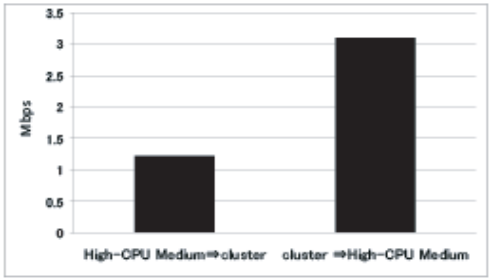

図 6 EC2 とクラスタ間の Network Throughput

4. 実　　験

本研究では自身のクラスタの負荷を一定間隔のモニタリングにより観察し、負荷が大きい場合は外部のクラウドコンピューティングリソースへ負荷分散を行うミドルウェアを構築し

システム評価を行う．Amazon EC2 インスタンスには High-CPU Medium のスペックのマシンを用意した．

4.1 実験環境

表 2　Experimental setup：PCs

OS	Linux 2.6.18-128.el5(CentOS5.3)
CPU	Initiator：Intel (R) Xeon(TM) 3.6GHz Target：Intel (R) Xeon(TM) 2.66GHz
Main Memory	Initiator(DomainU)：1GB Target：8GB
iSCSI	Initiator：iscsi-initiator-utils Target：iSCSI-Enterprise-Target
Monitoring Tool	dstat

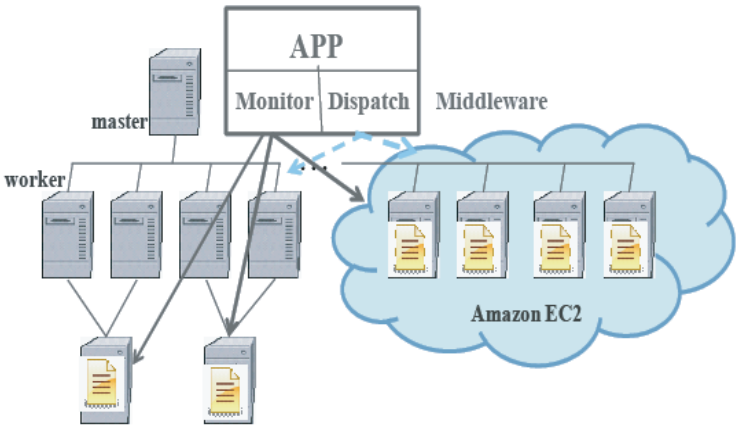

図 7　実験環境図

クラスタの各ノードのスペックを表 2 に、実験環境図を 7 に示す. クラスタの各計算ノードには DomainU(virtual machine) を一つずつ配置した．Amazon EC2 のマシンスペックは High-CPU Medium インスタンスを使用した．

ストレージについては、ローカル環境では iSCSI ストレージを使用し、EC2 ではローカルストレージを使用する．またデータについては上記の (b)「遠隔バックアップなどによりあらかじめリモートにデータが存在する」場合を考える．

負荷分散のためのリソースとして使用するクラウドコンピューティングの一つの大きな特徴として、必要なときに必要な分だけリソースを利用可能であることがあげられる．しかしクラウドは従量制のコストがかかるということも特徴であるため、実行時間とコストを考慮してリソースを配分する必要がある．今後はクラウドリソースが無限に使えるという条件において、クラウドでかかるコストを考慮したミドルウェアを構築したいと考えているが、本実験ではひとまず EC2 インスタンスを 4 台使用できるという条件において実験を行った．

本研究ではデータインテンシブアプリケーションの動作を想定している．そのため評価アプリケーションとしては PostgreSQL のベンチマークである pgbench を用いた．データベースの大きさは 7.5GByte，サーバで実行するユーザ数 1 に対するトランザクション数を 1000 とした．

ミドルウェアの評価のため，5 秒ごとに 4 ユーザずつ合計 10 回投入し 8，1 回毎にローカルクラスタと EC2 のどちらに投げるかを決定する．またシステムに負荷をかけ，より分かりやすくミドルウェアの評価を行うため，1 回の実行につき 6 回 pgbench を繰り返している．そして 1 回の平均実行時間を測定し，ローカルクラスタと Amazon EC2 へ理想的な振分けをした場合と、本研究で作成したミドルウェアを使用し振分けた場合の実行時間を比較する．また予備の実験により，pgbench 実行時 CPU やネットワークにはまだ余裕があり，ボトルネックはディスクアクセスであるということを確認している．

4.2 ミドルウェア概要

まず理想的な振分けを決定するため pgbench においてクライアント数毎の実行時間 9 を測定した．

この結果を元に，10 回ジョブを投入する際最も実行時間が速くなる一番理想的なジョブの振分けは

(local-EC2-local-local-EC2-local-local-EC2-local-local) であると決定される．この理想的な振分けは全体の実行時間はどんな振分けよりも最短であるが，個々の実行時間は必ずしも最短になるとは限らない．

ミドルウェアはモニタリング部とアプリケーション実行部に分けられる．モニタリング部では最もボトルネックになるであろうと考えられるディスクアクセス状況を，dstat コマンドにより 5 秒毎にモニタリングする．

アプリケーション部ではモニタリング部の結果を受けて，5 秒毎に 4 ユーザずつ投入され

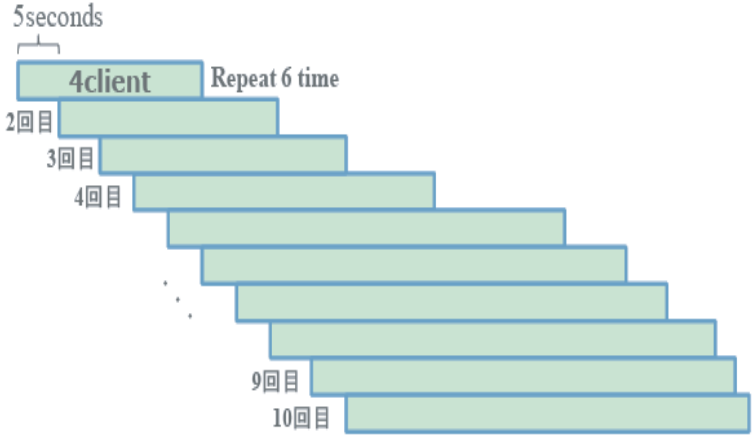

図 8 ジョブ投入イメージ

図 9 pgbench 実行時間

るジョブをローカルまたは EC2 に振分ける．図 10 は pgbench のそれぞれのクライアント数毎おけるディスク負荷である．

この図の Disk-Read よりクライアント数 1〜5 において負荷が右肩上がりになっていることが確認された．ただしクライアント数が 6 以降の場合，処理が飽和状態となり，ジョブが増えてもモニタリングには現れていない．そのためクライアント数が 6 相当以上の負荷がかかっている場合には乱数を発生させ，ローカルのマシンと EC2 の性能差により 2/3 の確率でローカルに，1/3 の確率で EC2 に投げ分けるよう設定する．

4.3 結　　果

図 11 は理想的な振分けとミドルウェアを使用した場合における 10 回の実行時間，表 3 はその平均である．一部理想的な振り分けをした場合の実行時間が長くなる場合が見受けられるが，偶発的な要因やデータベースの汚れ具合によるものである．

表 3 評価結果-平均

	Execution time
Ideal Pattern	256.65 (seconds)
Using Middleware	270.5 (seconds)

上記の結果よりモニタリングの値がディスクアクセス状況だけにも関わらず，ミドルウェアを使った場合に，理想に近い実行性能が出ていることが分かる．

振分けについても一番理想的な場合の振分けは
(local-EC2-local-local-EC2-local-local-EC2-local-local) であるが，ミドルウェアを使用した場合は
(local-EC2-local-local-local-local-EC2-local-local-local) となる場合や
(local-EC2-local-local-local-local-local-EC2-local-local) というように理想と異なる場合も多かったが，
(local-EC2-local-local-EC2-local-local-EC2-local-local) と理想と同一になる場合も見受けられた．

5. まとめと今後の課題

よりスケーラブルなデータ処理システムを実現するため，データインテンシブアプリケーション実行時，手元のクラスタ負荷をモニタリングし，負荷が大きい場合は外部のクラウドコンピューティングリソースへ負荷分散を実現するミドルウェアを構築した．データインテンシブアプリケーションにはデータベースベンチマークの pgbench を使用した．実行のボトルネックになるディスクアクセスの結果を元に，ローカルクラスタと EC2 のジョブの振分けを行ったところ，実行時間では理想に近い値が確認された．またローカルと EC2 への

User	Disk Read (kByte/sec)	Disk Write (kByte/sec)
1	3273	29000
2	4969	33666
3	5746	32000
4	6189	35000
5	6758	38000
6	6213	38333
7	6993	33000
8	6381	28666
9	6743	42500
10	7093	42000

図 10 ディスクアクセス負荷

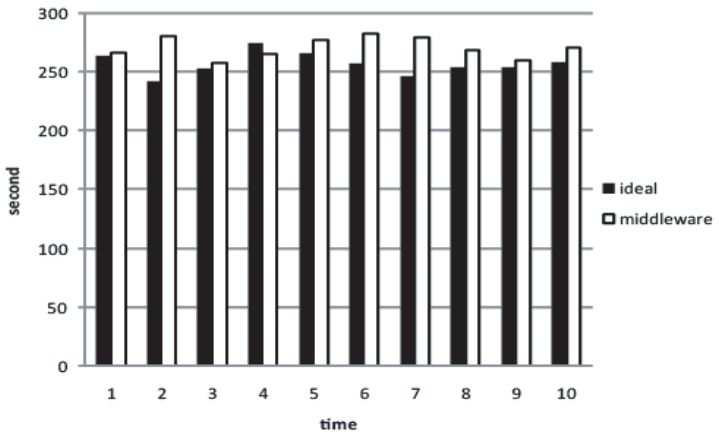

図 11 評価結果

振分けについても理想に近い振分けができていることが分かった．

現在は使用できる EC2 インスタンスの数を限定しているが，今後は際限なくクラウドリソースを使用できる環境を想定したミドルウェアに改良していく．ただクラウドには従量制のコストが発生するため，全体の実行時間と EC2 にかかるコストの 2 点を考慮する．

謝　　辞

本研究は一部，文部科学省科学研究費特定領域研究課題番号 18049013 によるものである．

参 考 文 献

1) Xen : http://www.xen.org/
2) iSCSI RFC: http://www.ietf.org/rfc/rfc3722.txt
3) 原明日香、神坂紀久子、山口 実靖、小口正人: ”並列データマイニング実行時の IP-SAN 統合型 PC クラスタのネットワーク特性解析”, DEIM2009,D3-5,2009 年 3 月
4) 小口正人、喜連川優:”ATM 結合 PC クラスタにおける動的リモートメモリ利用方式を用いた並列データマイニングの実行”, 電子情報通信学会論文誌,Vol.J84-D-I, No.9, pp.1336-1349, 2001 年 9 月
5) mpiBLAST:http://www.mpiblast.org/
6) Amazon Elastic Compute Cloud:http://aws.amazon.com/ec2/
7) Secure Data Connector:http://code.google.com/intl/ja/securedataconnector/ http://code.google.com/intl/ja/securedataconnector/
8) Shiori Toyoshima, Saneyasu Yamaguchi, and Masato Oguchi:”Storage Access Optimization with Virtual Machine Migration and Basic Performance Analysis of Amazon EC2,” In Proc. the Fourth International Workshop on Telecommunication Networking, Applications and Systems (TeNAS2010) in conjunction with the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA2010), pp.905-910, Perth, Australia, April 2010.
9) Ganglia Monitoring System:http://www.ganglia.info/
10) Aravind Menon,Alan L.Cox,Willy Zwaenepoel: ”Optimizing Network Virtualization in Xen”, USENIX Annual Technical Conference,2006 年
11) Jose Renato Santos,Yoshio Turner,G.(John)Janakiraman,Ian Pratt:”Bridging the Gap between Software and Hardware Techniques for I/O Virtualization”, USENIX Annual Technical Conference,2008 年
12) 谷村勇輔、小川宏高、中田秀基、田中良夫、関口智嗣: ”仮想クラスタに対する IP ストレージの提供方法の比較”, 「ハイパフォーマンスコンピューティングとアーキテクチャの評価」に関する北海道ワークショップ (HOKKE),2007 年