社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

VMライブマイグレーションにおける大規模I/O処理挙動に関する一考察

石田　涉† 横山　大作† 中野美由紀† 豊田　正史† 喜連川　優†

† 東京大学

あらまし 近年のクラウドコンピューティングでは, 計算資源の管理に仮想化技術を利用している. 仮想化技術の要素技術の一つである VM ライブマイグレーション等を利用することで, クラウドプロバイダではサービスを停止することなく計算資源の集約化や負荷分散が可能となる. 現在, ウェブコンテンツ, センシングデータなどのデジタルデータが急増するなか, クラウドコンピューティングにおける大容量のデータを処理するアプリケーションの効率化が急務である. しかしながら, 大規模 I/O 処理を行う VM ライブマイグレーションの挙動は十分に解析されていない. 本報告では, VM ライブマイグレーション時の入出力挙動を実機を用いて詳細に解析し, 大規模データ処理における VM ライブマイグレーションにおける課題について明らかにする.

キーワード クラウドコンピューティング, ライブマイグレーション, 伸縮性

Consideration of large-scale I/O processing during VM live migration

Wataru ISHIDA†, Daisaku YOKOYAMA†, Miyuki NAKANO†, Masashi TOYODA†, and Masaru KITSUREGAWA†

† University of Tokyo

Abstract Virtualization is used as a key technology for Cloud Computing to manage computational resources. VM live migration, which is one of the element technology of virtualization, enables cloud providers to consolidate computational resources or distribute loads. As digital data such as web contents or sensing data have been rapidly increasing, it becomes imperative to process massive data efficiently in clients' applications. However the behavior of VM live migration in which massive I/O operations are required haven't precisely analyzed yet. In this report, we analyze the behavior of VM live migration in which I/O intensive applications runs with large I/O access in detail using real machines, and then classify the importance of VM live migration performance with heavy I/O applications.

Key words Cloud Computing, Live Migration, Elasticity

1. はじめに

近年 Amazon EC2 [8], Salasforce [11], Microsoft Azure [9], Google AppEngine [10] といったクラウドサービスが広く利用されている. クラウドコンピューティングはクライアントが必要とする計算資源を必要なときに必要なだけ提供することができる新しい IT 基盤として認識されており, さらに大きな利点の一つとして, アプリケーションが意識することなく割り当てる計算資源を動的に増減できるエラスティシティ(伸縮性) の存在が挙げられる.

クラウドコンピューティングでは, e コマースやオンラインゲームなど, データベースがサービスの根幹となるようなアプリケーションの利用が多い. これらのアプリケーションでは利用者数の変動, アクセスされるデータ量の変動が大きく, 大規模な I/O 処理を必要とするものが増えており, エラスティックな計算資源を提供することがクラウドプロバイダにとって急務となっている. クラウドコンピューティングにおいてクライアントアプリケーションは仮想化技術を利用して VM(Virtual Machine) として提供されることが一般的である. 仮想化技術を利用した場合, エラスティシティは VM ライブマイグレーションを利用することで実現可能であるが, クライアントアプリケーションが大規模な I/O 処理を伴う際の挙動については十分な解析がされていない. 図 1 に本論文が対象とするクライアントアプリケーションの構成を示す. 対象とするアプリケーションではアプリケーションロジックが実装されるアプリケーションサーバが存在するアプリケーション層とデータ管理を行

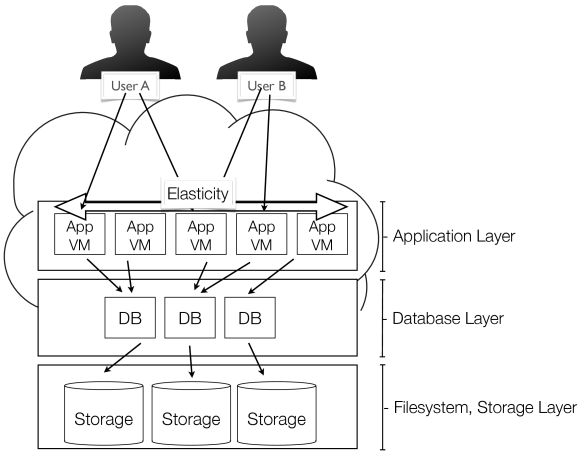

図 1　対象とするクライアントアプリケーションの構成
Fig. 1　client application

うデータベースサーバが存在するデータベース層が分けて実装され, アプリケーションサーバは VM として実装されているとする.

本論文では大規模な I/O 処理を伴うアプリケーションサーバの VM ライブマイグレーションについてデータベースのキャッシュまで考慮した VM ライブマイグレーションの課題を検討し, 単純な VM ライブマイグレーションのままでは性能低下を招く可能性があることを示す. またアプリケーションとデータベースが協調し, データベースのキャッシュの情報を共有することでキャッシュミスを減らす仕組みを検討する. さらに大規模な I/O 処理を伴う VM ライブマイグレーション時の入出力挙動を実機を用いて詳細に解析し, 大規模データ処理における VM ライブマイグレーションにおける課題について明らかにする.

本論文の構成は以下のようになっている. 2. で現在クラウドコンピューティングで広く利用されている仮想化技術を概説し, 仮想化技術の一要素技術である VM ライブマイグレーションの仕組みを説明する. 3. では本論文が対象とするクライアントアプリケーションのエラスティシティを実現するための課題を挙げ, それに対する解決手法を提案し, 4. で 3. で考察した問題を検証するための計測方法と計測結果を述べる. 次に 5. で関連する研究を紹介し, 最後に 6. で結論を述べる.

2. 仮想化技術と VM ライブマイグレーション

クラウドコンピューティングの実装には仮想化技術が広く利用されている. 仮想化技術は物理マシン上で稼働する HV(Hypervisor) がエミュレートした仮想的な物理環境上で VM を稼働させるもので, VM は自らが仮想化されていることを意識せず, HV を物理マシンと認識し稼働する. クラウドプロバイダは仮想化技術を利用しクライアントに提供する計算資源を VM として扱うことで, 物理的な設定などを必要とせず, クライアントに迅速に計算資源を提供できる. またクライアントはクラウドプロバイダが持つ物理インフラを意識せずに自分専用の計算資源として VM を利用することができる. 代表的な HV の実装として KVM [12] と Xen [13] がある.

VM ライブマイグレーションとは HV 上で稼働している VM を停止させることなく, 異なる HV 上に移す技術で Clark らによって提案された [6]. VM のメモリと CPU ステート, ハードウェアのステートを VM を止めずに段階的にマイグレート先の HV に転送することで行う. VM の動作中はメモリの書き換えが生じるため多くの場合メモリの再転送が必要となり, VM ライブマイグレーションにおいてはこのメモリ転送量の削減がマイグレーションのオーバーヘッドを減らす要となっている. VM ライブマイグレーションにおけるメモリの転送量の削減に関しては多くの研究がなされており, 代表的なものに [3], [4], [5] がある. VM のディスクイメージはマイグレーション元と先の両 HV がアクセスできる仮想ディスクに保管することでマイグレーション後の VM のディスクアクセスを可能にするのが一般的であるが [7] のようにディスクイメージの転送も行うマイグレーションも提案されている. VM ライブマイグレーションにより, クライアントの VM の HV への集約, 分散が可能になり VM が必要とする計算資源と HV が提供できる計算資源のバランスを考慮し VM をライブマイグレーションにより再配置することでエラスティックな計算資源をクライアントに提供することができる.

3. 大規模 I/O 処理を伴うクライアントアプリケーションのエラスティシティ

本節では図 1 に示したようなアプリケーションサーバの下に DB サーバがあるクライアントアプリケーションでのエラスティシティを実現するための課題を考察する.

図 2 に示すように, クライアントアプリケーションのアプリケーションサーバを $APPVM1$, $APPVM2$, $APPVM3$, ハイパバイザーを $HV1$, $HV2$, DB サーバを $DB1$, $DB2$ とする. $DB1$ と $DB2$ は両者で同じマスターノードを参照しているスレーブノードとし, アプリケーションサーバはどちらの DB サーバにも接続できるとする. はじめすべてのアプリケーションサーバは $HV1$ 上に集約されて稼働し, $DB1$ に接続しているとする.

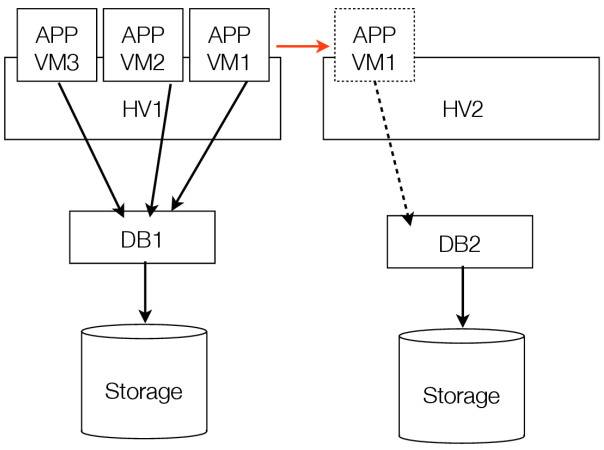

図 2　クラウドアプリケーションでのクライアントアプリケーションの実装
Fig. 2　cloud application implementation

ここで $APPVM1$ の負荷が上昇したと想定する．その結果 $HV1$ が十分な計算資源を提供できなくなった場合, $APPVM1$ を余力のある $HV2$ にライブマイグレートすることによって $APPVM1$ に十分な計算資源を割り当てることができる (図 2 赤線部)．しかし $APPVM1$ の計算スループットが上昇することで $DB1$ への I/O レートも増加するため, 今度は $DB1$ の計算資源が足りなくなりクライアントアプリケーション全体としては負荷の上昇に応えられなくなることが想定される．

そこで $APPVM1$ のマイグレーションと同時に接続する DB サーバを $DB1$ から余力のある $DB2$ に切り替えることで DB サーバの負荷も分散することを考える (図 2 破線部)．しかしこの場合でも $DB2$ は十分な I/O 性能を $APPVM1$ に対して出せない可能性がある．それは $DB1$ が $APPVM1$ との I/O で利用していた DB キャッシュを $DB2$ は持っていないためである． アクセスするデータが DB キャッシュにある場合, DB サーバからのデータの読み出しはメモリアクセスで済むのに対し, DB キャッシュにない場合はディスクアクセスを行わなければいけない．そのため本論文が想定するような大規模な I/O 処理を行うクライアントアプリケーションの場合, VM ライブマイグレーションの後に接続する DB サーバを変更するとキャッシュが温まるまでディスクアクセスが頻発することがクライアントアプリケーションにとって大きなオーバーヘッドになってしまうと考えられる．

以上がクラウドコンピューティングにおいて大規模 I/O 処理を行うクライアントアプリケーションのエラスティシティがアプリケーションサーバの VM ライブマイグレーションだけでは達成できない理由である．そこで事前に DB 層がアプリケーションサーバがアクセスするテーブルの情報を共有し, DB キャッシュ上に当該データをプレロードすることで VM ライブマイグレーション後に接続する DB サーバを変更した際のディスクアクセスを低減できる可能性がある．これにより VM ライブマイグレーション後のアプリケーションサーバのパフォーマンス低下が抑えられ, より積極的に VM ライブマイグレーションを行えるようになり, 大規模 I/O 処理を行うクライアントアプリケーションにおいて高いスケーラビリティを実現できる．つまりアプリケーションサーバと接続している各 DB サーバはアプリケーションサーバがそれぞれ参照するテーブルの情報を持っていることを利用し, マイグレーション後の接続先となる DB サーバにテーブルを先読みさせ, アプリケーションサーバの VM ライブマイグレーションが起こった時には新しい接続先の DB サーバに参照するテーブルのキャッシュが用意されている状況を実現する．

クラウドコンピューティングにおけるデータベースのエラスティシティに関する研究としては Aaron らによる Albatross [2] や Sudipto らによる Zephyr [1] があるが, 両者は DB サーバ自身の VM ライブマイグレーションを実現するものでアプリケーションサーバとの協調は考えられていない．Albatross や Zephyr のアプローチはアプリケーション層とデータベース層とのアイソレーションを保ったままエラスティシティを実現しようとするものなのに対し, 本提案手法のアプローチはデータベース層がエラスティシティに関してイニシアチブを持つことを想定している．

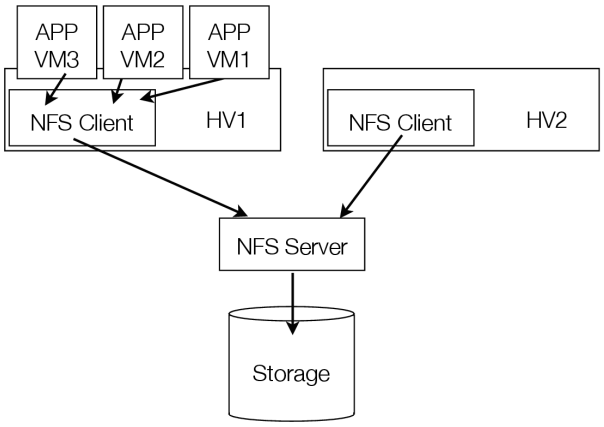

図 3 計測環境
Fig. 3 experiment environment

4. VM ライブマイグレーションにおける大規模 I/O 処理挙動の解析

前節でアプリケーションサーバの VM ライブマイグレーションと同時に DB サーバを変更した場合, マイグレーション前に利用していた DB キャッシュが利用できないことがクライアントアプリケーションにとって大きなオーバーヘッドになると述べた．本節では実機を用い実際に接続する DB の変更がクライアントアプリケーションにとってオーバーヘッドになるのかを検証した．

4.1 計測環境

具体的なクライアントアプリケーションの実装は難しい．そこで NFS を用いた簡単なモデルを構築した．図 3 に計測モデルを示す．

VM のディスクイメージはマイグレーション元とマイグレーション先の 2 台の HV で NFS を用いてマウントされた NFS ボリューム内に配置し, VM 内でローカルファイルに対して単純なシーケンシャルリードを行いながら VM ライブマイグレーションを行い VM の読み込みスループットと NFS サーバのディスク読み込みスループットを測定した．このとき計測モデルに存在するキャッシュは NFS サーバのキャッシュ, NFS クライアント (HV) のキャッシュ, VM のキャッシュである．VM 上で読み込みを行う際, 利用される可能性のあるキャッシュは VM のキャッシュ, そのとき VM が稼働している HV のキャッシュ, NFS サーバのキャッシュであり, すべてのキャッシュにヒットしなかった場合, NFS サーバでディスクアクセスが行われる．計測モデル (図 3) と想定する実環境 (図 2) では $APPVM$ と $APPVM$, NFS クライアントと NFS サーバの両者を合わせたものと DB サーバがそれぞれ対応している．計測は全てのキャッシュがクリアされている場合 (図 4(a)), 全てのキャッシュを事前に温めた状態にしておいた場合 (図 4(b)), マイグレーション元の HV のキャッシュのみを温めた場合 (図 4(c)) の 3 つのパターンで行った．各図で濃く塗りつぶされた矩形は温めた状態

のキャッシュ, 薄く塗りつぶされた矩形はクリアされたキャッシュを表している. NFS のキャッシュはページキャッシュとして実装されているため, NFS クライアントと NFS サーバのキャッシュクリアはページキャッシュのクリアによって行うことができる. VM 内のキャッシュクリアもファイルの読み込みに対するキャッシュクリアのため, ページキャッシュのクリアで行うことができる. キャッシュの温めは実際に VM 内でシーケンシャルリードを事前に行うことで再現した. 但しモデル内の全てのキャッシュを事前に温める場合は, 片方の HV 上でシーケンシャルリードを行った後, もう片方の HV へ VM ライブマイグレーションを行い, 再度シーケンシャルリードを行わなければいけない.

次に各パターンにおいてどのような I/O 挙動が起こることが想定されるかを説明する. 全てのキャッシュがクリアされている場合 (図 4(a)) は VM 内で行われるシーケンシャルリード全てに対しキャッシュミスが生じ, NFS サーバのディスク読み込みが行われると予想される. マイグレーション前後でキャッシュに関する状況に変化はないため, VM の読み込みスループットもマイグレーション前後で変化しないと考えられる. 全てのキャッシュを事前に温めた状態にしておいた場合 (図 4(b)) は VM 内で行われるシーケンシャルリード全てに対しキャッシュヒットすることで NFS サーバでのディスク読み込みは発生しないと予想される. マイグレーション前後でキャッシュに関する状況に変化はないため, VM の読み込みスループットもマイグレーション前後で変化しないと考えられる. マイグレーション元の HV のキャッシュのみを温めた場合 (図 4(c)) はマイグレーション元の HV のキャッシュは温まっているため, キャッシュヒットすることで NFS サーバでのディスク読み込みは発生しないと予想されるが, マイグレーション後, マイグレーション先の HV のキャッシュ, NFS サーバのキャッシュはクリアされているために NFS サーバでのディスク読み込みが発生すると考えられる. またマイグレーション前に比べマイグレーション後ではディスクアクセスのオーバーヘッドがある分 VM の読み込みスループットは低下すると考えられる.

さらに複数のクライアントが物理インフラを共有して利用するクラウドコンピューティングの実環境を想定し, VM の台数を 1 台だけでなく 6 台同時に VM ライブマイグレーションを行った時の挙動も計測した. 計算機環境は表 1 に示す.

4.2 計測結果

4.2.1 VM1 台の場合

まず VM1 台をライブマイグレートしたときの結果を説明する. 結果を図 5 に示す. VM 上で行ったシーケンシャルリードは VM のローカルファイルシステム上のファイルを 16KB ずつ約 100us の間隔で読み込むものである. 図 5(a) はすべてのキャッシュをクリアした場合, 図 5(b) はすべてのキャッシュを事前に温めた場合, 図 5(c) はマイグレーション元の HV のキャッシュのみを温めた場合の結果を示す. 各図の縦軸は VM の読み込みスループットと NFS サーバのディスク読み込みスループットを示し, 横軸は計測経過時間を示す. また各図中央付近の棒線は VM ライブマイグレーションを行なっている区間を

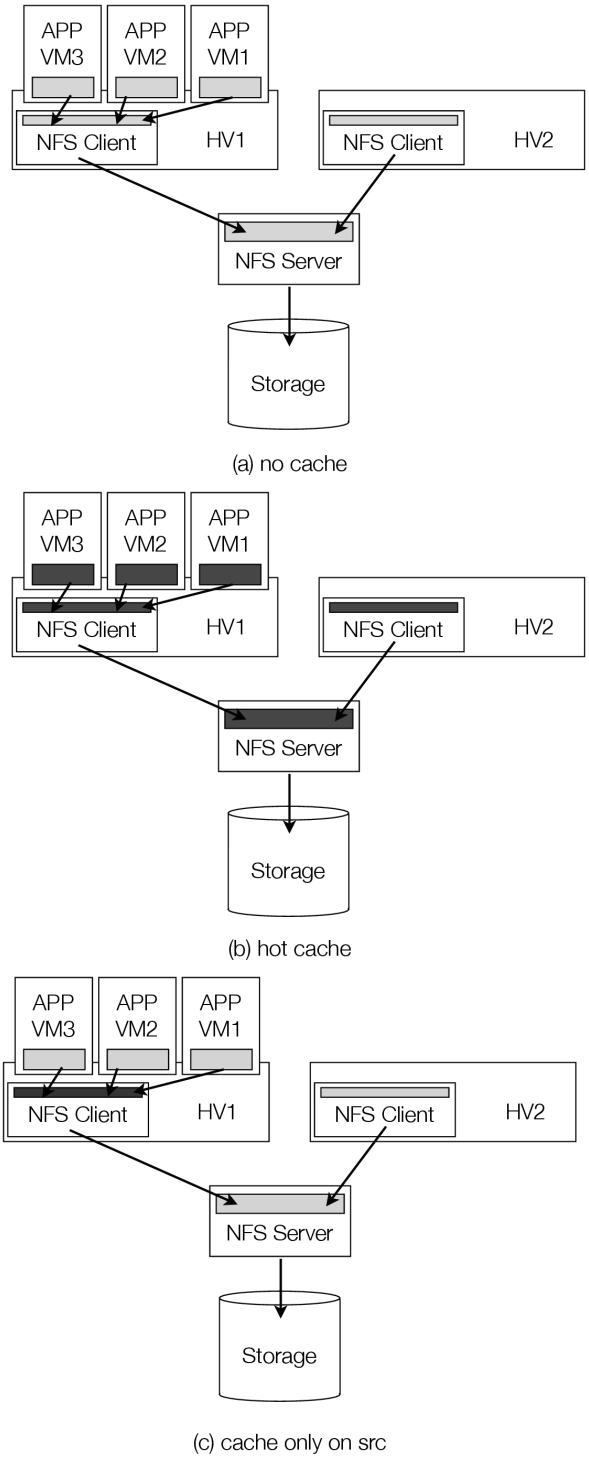

図 4　キャッシュの有無による 3 パターンの計測環境
Fig. 4　measurement patterns

表す. 図 5(a) と図 5(b) よりマイグレーションを行なっていない区間ではキャッシュがある場合は約 120MB/s の読み込みスループットがあるのに対し, キャッシュがヒットしない場合は約 100MB/s まで読み込みスループットが落ち込むことがわかる. これは図 5(a) で VM の読み込みと同じだけ NFS サーバでのディスク読み込みが生じていることからディスクアクセスのオーバーヘッドによるものである. また図 5(c) よりマイグレーション先にキャッシュがないためにマイグレーションの終了後

表 1　計算機環境

Table1　conputational environment

HV	CPU	AMD Opteron 6100 2. 2GHz
	memory	512GB
	OS	Debian 6. 0. 4 Squeeze
	kernel	Linux 2. 6. 32
	qemu	qemu-kvm-1. 0. 1
	NFS Client	NFS v3
	network	Intel 82599EB 10Gbit
VM	CPU	1 virtual CPU
	memory	1GB
	OS	Ubuntu 11. 04
	kernel	Linux 3. 0. 0
	filesystem	ext4
	blk driver	virtio-blk
	network	virtio-net
NFS	CPU	Intel Xeon E5530 2. 4Ghz
	memory	24GB
	OS	Debian 6. 0. 5 Squeeze
	kernel	Linux 2. 6. 32
	NFS Server	NFS v3
	HDD	SATA 500GB
	export FS	ext4
	network	Intel 82599EB 10Gbit

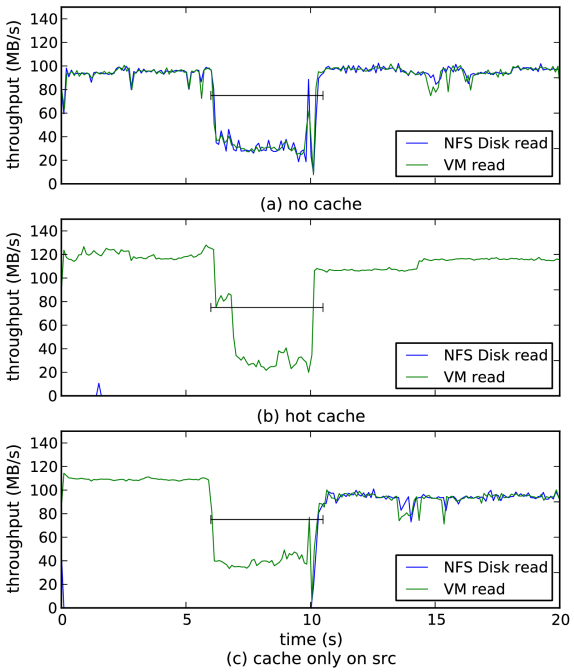

図 5　VM1 台でのライブマイグレーション

Fig. 5　live migration with 1VM

VM の読み込みスループットが図 5(a) のときと同程度落ち込んでしまうことがわかった.

4. 2. 2　VM6 台の場合

次に VM6 台をライブマイグレートしたときの結果を説明する. 結果は図 6 にまとめた. 各項目は VM1 台のときと同じであるが, VM の読み込みスループットに関しては 6 台の VM の読み込みスループットの平均を示している. また各 VM 上で行ったシーケンシャルリードは VM のローカルファイルシステム上のファイルを 8KB ずつ約 100us の間隔で読み込むものとした. VM が 6 台になるとディスクアクセスのオーバーヘッドの影響が顕著になる. これはディスクの I/O 帯域を 6 分割しなければいけないという理由だけでなく, VM6 台がそれぞれ別のファイルをシーケンシャルリードするためにディスクのシークが増えることにも起因すると考えられる. 図 6(a) と図 6(b) を比較すると, マイグレーションを行なっていない区間ではキャッシュがある場合は約 80MB/s の読み込みスループットがあるのに対し, ディスクアクセスが生じる場合は約 10MB/s まで読み込みスループットが落ち込んでしまう. また図 5(a) と図 6(a) を比較すると図 5(a) ではディスクの読み込みスループットが 100MB/s であったのに対して, 図 6(a) ではディスクの読み込みスループットが 50MB/s 前後まで低下しており, ディスクアクセスのオーバーヘッドの影響が大きくなっていることがわかる. また図 6(c) よりマイグレーション先にキャッシュがない場合, マイグレーション終了後に VM の読み込みスループットが大きく低下することがわかった.

図 5(c) と図 6(c) に関して, 本計測では VM 内で行う I/O 処理がシーケンシャルリードであるためにマイグレーション後にキャッシュが温まることによって VM の読み込みスループットが改善される様子が計測できなかった. しかし実際に本論文が想定するようなアプリケーションにおいてはマイグレーション後, マイグレーション先のキャッシュが徐々に温まることで VM の読み込みスループットはマイグレーション以前と同等かそれ以上になると考えられる.

以上の結果から 3. で考察した通り, 大規模 I/O 処理を行うクライアントアプリケーションではアプリケーションサーバの VM ライブマイグレーションだけでエラスティシティを得ることは難しく, 提案手法のようにアプリケーション層とデータベース層が協調してエラスティシティを実現しようとする仕組みが必要とされることがわかった.

5.　関 連 研 究

クラウドコンピューティングにおけるデータベースのエラスティシティに関する研究としては 3. でも紹介した通り Sudipto らによる Zephyr [1] と Aaron らによる Albatross [2] がある.

Zephyr は複数のクライアントに物理インフラを共有させるマルチテナントなデータベースで, エラスティックな負荷分散を行うためのデータベースのライブマイグレーション手法である. Zephyr ではマイグレーション元とマイグレーション先はデータベースのテーブルを含め何も共有していない状況を前提としており, ライブマイグレーションの前後を通してトランザクション処理の中断を生じさせないためにデータベースのライブマイグレーション中にトランザクション処理はマイグレーション先で行いつつ, 必要なデータがマイグレーション先にない場

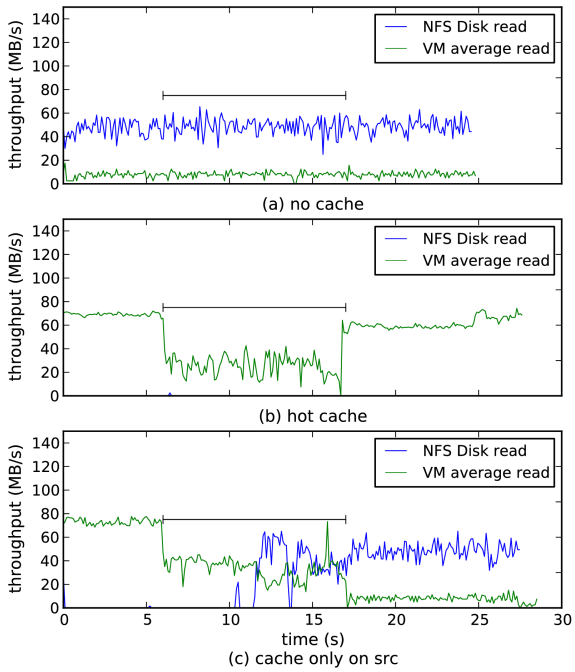

図 6　VM6 台でのライブマイグレーション
Fig. 6　live migration with 6VMs

合はオンデマンドに読み込む手法を提案している．Albatross も Zephyr と同様マルチテナントなデータベースのライブマイグレーションであるが, Zephyr とは異なり各物理ノードがストレージを共有しているシステムを前提としている．Albatross では VM ライブマイグレーションのメモリの転送と同様, トランザクションテーブルをイテレーティブに転送することでライブマイグレーションの前後を通してトランザクション処理の中断を避けデータベースにおけるエラスティシティの実現手法を提案している．

6. ま と め

本論文ではクラウドコンピューティングにおいて近年増加傾向にある大規模な I/O 処理を行うアプリケーションサーバと DB サーバの連携により成り立つクライアントアプリケーションのエラスティシティを実現するための課題を考察し, それに対する解決手法を提案した．また NFS を用いた簡単なモデルで大規模な I/O 処理を行う VM ライブマイグレーションしたときの挙動を実機により解析し, キャッシュの有無が VM ライブマイグレーション後の VM のパフォーマンスに大きな影響を与えることを明らかにした．今後は今回得られた知見を元に実際に DB を利用した際の挙動を調べ, 提案手法の具体的な実装方法を検討していきたい．

謝辞　本研究の一部は、総務省委託研究「広域災害対応型クラウド基盤構築に向けた研究開発（高信頼クラウドサービス制御基盤技術）」において実施された。

文　献

[1] Aaron J.Elmore, Sudipto Das, Divyakant Agrawal, Amr El Abbadi *Zephyr: Live Migration in Shared Nothing Databases for Elastic Cloud Platforms* SIGMOD, 2011.
[2] Sudipto Das, Shoji Nishimura, Divyakant Agrawal, Amr El Abbadi *Albatross: Lightweight Elasticity in Shared Storage Databases for the Cloud using Live Data Migration* VLDB, 2011.
[3] Jie Zheng, T.S.Eugene Ng and Kunwadee Sripanidkulchai. *Workload-Aware Live Storage Migration for Clouds.* VEE, March 2011.
[4] Samer Al-Kiswany, Dinesh Subhraveti, Prasenjit Sarkar and Matei Ripeanu. *VMFlock: Virtual Machine Co-Migration for the Cloud* HPDC, June 2011.
[5] Umesh Deshpande, Xiaoshuang Wang and Kartik Gopalan. *Live Gang Migration of Virtual Machines* HPDC, June 2011.
[6] C. Clark, K. Fraser, Steven Hand, et al. *Live Migration of Virtual Machines* NSDI, 2005.
[7] T. Hirofuchi, H. Ogawa, H. Nakada, et al. *A Live Storage Migration Mechanism over WAN for Relocatable Virtual Machine Services on Clouds* CCGrid, 2009.
[8] Amazon Elastic Compute Cloud(Amazon EC2). http://aws.amazon.com/ec2/.
[9] Microsoft azure. http://www.microsoft.com/azure
[10] Google App Engine. http://code.google.com/appengine
[11] Salesforce.com http://www.salesforce.com/
[12] KVM http://www.linux-kvm.org/page/Main_Page
[13] xen http://xen.org/