メモリ共通情報を用いた Live Migration の負荷軽減手法

61106719 情報工学 四年 上司 陽平

2015 年 1 月 13 日

目次

1 はじめに

1.1 背景

現在，仮想化技術が様々なサービスを提供するための計算資源管理などの基盤技術て用いられている．仮想化技術の一つである Live Migration は，クラウドの計算資理において有益である．Live Migration とはある物理マシン上で稼働する仮想マシ稼働したまま他の物理マシン上に移送する技術である．その他にも仮想化技術が利用ている一つの例としてクラウドコンピューティングの PaaS などがあげられる．クラコンピューティングとはネットワークを介して計算資源をユーザに提供するというサスである．クラウドコンピューティングの一つとして PaaS というサービスがあり計源の管理や利用に効果的である．PaaS とはアプリケーションを実行するためのプラフォームをネットワークを介してユーザに提供するというサービスである．PaaS でされるプラットフォームは実際には 1 つのプラットフォームに 1 台の実機があること く，仮想化技術により実機マシン 1 台上に複数の仮装マシンを起動させ，その仮装マ 1 台をプラットフォームとして提供する．よって PaaS では一つの物理マシン上に OS で同様のアプリケーションが動いている仮想マシンが多く稼働していて，その仮シンの 1 つがプラットフォームとしてユーザに提供される．(挿図)

上記の例は一部で他にも様々な仮想化技術が存在し，計算資源の管理などの場面で立っている．

1.2 問題点

既存の Live Migration では仮想マシンの全てのメモリを異なる物理マシン上に送るためメモリの送信総量が大きい．Live Migration に必要な時間の大半はメモリ送信るため，送信量が大きいと移送に時間がかかる．また送信するページが大きいほど多ネットワーク資源を使用する．また，Live Migration は移送時間が長いほど多くの C を使用する．他にも，PaaS のような環境では移送先で移送仮想マシンと同じ内容のリを持つ仮想マシンがある場合がある．その場合，移送する仮想マシンの移送先に同容のあるメモリページは再利用可能なので，そのメモリページは移送せず移送先で共再利用することが可能である．(挿図)

1.3 提案

PaaS 環境における Live Migration で移送仮想マシンのメモリと同様のメモリが先物理マシンにあるときはそのメモリは移送せず，移送先で共有し再利用する．移送リの総量を減らす．それによって移送時間を短くし，使用ネットワーク量も減らす．時間を短くすることで CPU の使用率も削減する．

1.4 実験

既存の Live Migration より提案手法の Live Migration の方が移送時間も短くネワークと CPU の使用量が少ないことを示した．

2　仮想化技術

現在仮想化技術は様々なサービス提供の基盤技術として使用されている．本章では化技術の全般の話からクラウドコンピューティングにおける仮想化，Live Migratio ついてなど詳細な内容を説明する．

2.1　仮想化

2.1.1　仮想化とは

現在 IT の複雑さが増しており，取り扱いも比例して複雑化している．そのようなを簡単で扱いやすいものにするのが仮想化の根本的な考え方である．つまり IT にお仮想化とは「ストレージ，メモリなどの計算資源の複雑な技術特性を隠蔽し，論理的用単位にし提供する技術」と考えることができる．その利用単位の粒度によって仮想様々な種類にわけられる．粒度の小さい例で言えばマルチプロセッサがあげられる．は CPU を仮想化する技術で，物理的には一つしかない CPU を時間単位などで使用配することであたかも複数の CPU が動作しているように振舞わせる技術である．ま理リソースの特性を隠蔽し，ユーザにサービスを提供する OS も仮想化といえる．大粒度で見ればサーバのクラスタリング技術などがあげられる．クラスタリング技術と数のコンピュータを繋げて，クライアントには 1 台のコンピュータだけが動作していのように振る舞わせる技術である．

2.1.2　サーバ仮想化

このように IT システムでは色々な仮想化技術で使用されている．そのような技術とつとしてサーバの仮想化という技術がある．これは物理サーバを論理的な構成単位割して利用する技術である．本来 OS は，CPU，メモリなどの物理資源を全て使用いるように振る舞う．そのため通常は，複数の OS で一つの物理資源を使用することきない．サーバ仮想化技術では実際の物理マシンの上で仮想マシンを作り出し，OS その仮想マシンを与えることであたかも実際に物理マシンを与えられている様に振るせることができる．(挿図)

サーバの仮想化の利点は多々ある．その利点について説明する．

OS が全ての物理資源を使用してその OS が使用しない物理資源が存在する方，仮想化を行うことで物理資源を複数の OS で効率的に分配し有効に利用すとができる．

- システムの柔軟性
  サーバを物理的な制約を考えることなく管理できる．サーバに与えられている仮想マシンで，多くの仮想マシンがが一台のマシンで稼働するので多大数を一管理することができる．またハードウェアのデバイスチェックなどの時間も省きるので立ち上げの高速化や Live Migration により仮想マシンを稼働させた他の物理マシンに移送することで物理マシンのメンテナンスも容易になる．
- 省コスト・省電力
  物理サーバの運用台数を削減できるので電力の消費，設置スペースの削減が行
- 障害性
  仮想マシンごとは完全に分離されているのでいずれかの仮想マシン上の OS ラッシュしても他のマシンには影響をあたえない．

このようにサーバ仮想化には様々な利点がある．

2.1.3 サーバ仮想化方式

サーバの仮想化には様々な方法が存在する．その方法は大きく二つに分類され，ホ OS 型とハイパーバイザ型に大別される．

1. ホスト OS 型 図（挿図）のように物理マシン上でひとつのホストとなる OS 働し，その OS 上で仮想マシン・モニタを起動する方式．仮想マシン・モニ は仮想マシンを OS に提供するソフトウェアの事を指す．ホスト OS 型は既 マシン環境で，他のアプリケーションを使用するようにインストールし，起動 ことができるので仮想化のために新たに環境を用意する必要がない．一例と Macintosh マシン上でアプリケーションの VirtualBox[1] などの仮想マシン ニタを用いて Windows を起動するなどということができる．一方，仮想マシ モニタに与えられる CPU 処理時間がホストマシンの OS のアプリケーション ケジューリング依存になるので，ホスト OS で多くのアプリケーションを起動 いる場合や多数のゲスト OS を起動している場合性能があまり出ないことがあ
2. ハイパーバイザ型 ホスト OS 型だと仮想マシンと物理マシンの間にはホスト

シンモニタのみが存在する．ハイパーバイザによってハードウェアは仮想化
OS に提供される．ホスト OS が存在せずハイパーバイザが直に物理資源を利
るためハイパーバイザがゲスト OS へ与えられる CPU のスケジューリング行
とができる．そうすることによりホスト OS 型と比べると，効率的にかつ安定
CPU をゲスト OS に割り当てることができる．vmware の vSphere[4]，citri
XenServer[5]，Microsoft の Hyper-V[6] など商用の仮想化ソフトウェアとし
イパーバイザ型の商品も多数存在する．オープンソースでの開発も行われてい
XenSever の元である Xen[7] や Linux カーネルにマージされている KVM[8
どがあげられる．

- 完全仮想化ページ数稼ぎたい場合は完全仮想化の話とかする

2.2 クラウドコンピューティングにおける仮想化技術の利用

2.2.1 クラウドコンピューティングとは

クラウドコンピューティングとは，インターネットを介して，必要に応じた計算資
ユーザに提供するサービスである．ユーザは実際の物理マシンを意識せず計算資源を
することができる．

従来ユーザがシステムを構築しようとしたとき，物理マシンを用意し，そのうえに
ソフトウェアをインストールし，必要な環境を用意することで初めて利用を開始し
た．また使用を開始した後も物理マシンのメンテナンスから，ソフトウェアの環境の
など様々な管理コストが掛かっていた．一方クラウドコンピューティングでは必要な
をユーザ登録などを済ませることですぐさま利用することができる．また利用するユ
は物理マシンのメンテナンスなどをサービス提供者に任せることができる．クラウド
ピューティングでは，下位層から上位層様々な層のサービスを提供する．具体的には
層では基幹ソフトウェアも何も用意していないようなマシンから上位層であればソ
ウェアのみのサービスなどである．ユーザは自分の提供された計算資源より上位層の
だけ管理すればよいので従来必要だった管理コストを他に割り当てることができる．
ばストレージのみをクラウドコンピューティングで利用する場合，保存したファイ
バックアップ作業はサービス提供元が行ってくれるのでバックアップの為に RAID
成するなどの管理作業からユーザは解放されることになる．

2.2.2 PaaS とは

クラウドコンピューティングの一つのサービス体系で，アプリケーションが稼働すめのハードウェアや OS などのプラットフォーム一式をインターネットを介して提るサービスである．

プラットフォームを提供されるユーザは開発，運用のためのハードウェアや OS，環境，ミドルウェアなどのプラットフォーム一式を自ら構築しなくてよい．ハードウのメンテナンスや障害対応もしなくてよくなる．またユーザは利用規模に応じたサーを受けることができるので利用規模に応じた柔軟な計算資源の利用をすることができ

PaaS 提供者は，プラットフォーム一式を大規模なデータセンタなどに用意して，企業へネットワークを通じてプラットフォーム一式を提供する．データセンタ内では の物理マシンサーバが用意され，プラットフォームの管理には仮想化技術が利用され る．各物理マシン上で仮想化技術により複数の仮想マシンが稼働しており，その仮想ン上に同様の構成のプラットフォーム一式が構築される．（挿図）ユーザにはその仮シン上のプラットフォーム一式が提供される．PaaS 提供者は各物理マシンの整備かドルウェア，開発環境などの管理をすることになる．

具体的な一例として Microsoft の Azure[10] をあげてみる．Azure のサービスは IaPaaS，ストレージなど多岐にわたる．そのうち， PaaS のサービスでは Azure Webというサービスがある．これは管理されたプラットフォームをユーザに提供し，ユープラットフォーム上で Python，Java，PHP，ASP.NET など様々な言語を使用しライムの Web アプリケーションを使用することができる．プラットフォームは WindSerever に手を加えたものが使われており，ミドルウェアなども含め多くの同じプロムを起動している仮想マシンが複数個データセンタの物理マシン上に存在している．

2.2.3 IaaS と SaaS 説明

ページ数足りなかったら

2.3 Live Migration

2.3.1 Live Migration とは

Live Migration とは，アプリケーションを稼働したまま仮想マシンを他の物理マ

パーバイザ型では VMware の vSphere[4] では「VMotion」，Citrix XenServer[5]「XenMotion」，Microsoft の Hyper-V[6] やオープンソースの KVM[8]．Xen[7] でのまま「Live Migration」という名前の機能として使われている．

2.3.2 Live Migration の使用

Live Migration が使用される場面を説明する．まず複数の仮想マシンの負荷総量理マシンの処理能力を上回った場合の利用についてあげる．元来，仮想化を導入するの一つにハードウェアの効率的利用があげられる．高性能のハードウェアの計算資源に全て利用することは難しいので，複数のサーバをその一台に集約することで計算資効率的に利用することができる．しかしそのような使い方をすると一台の物理マシン稼働する各仮想マシンが高負荷な処理を行ったとき，物理マシンの処理能力を複数のマシンの負荷が上回ってしまうことがある．上回った場合，各仮想マシンへの CPU は満足のいくものではなくなり各仮想マシン上で動いているサービスの性能が低下しまう．このような場合に Live Migration の使用が考えられる．負荷総量の上昇を検きた段階で一部の仮想マシンを他の物理マシンに移送すれば負荷総量が物理マシンの能力を超さないで済む．(図 1) 引用 OK か確認?どちらにしろ図が悪い?

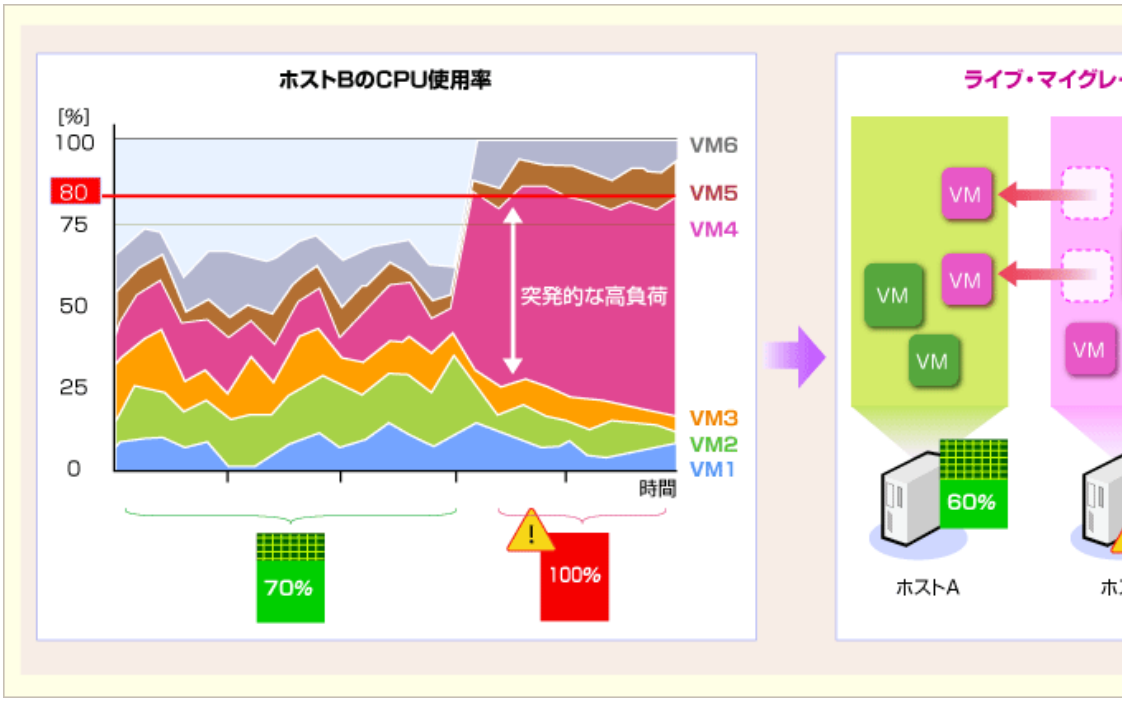

図 1 {

高負荷分散 (引用：[13])}

次にあげられるのが物理マシンのメンテナンスをする場合である．ホストマシン上働する仮想マシン・モニタはバグ修正，機能向上のために定期的にパッチがリリースる．そのため定期的にホストマシンを停止し，仮想マシン・モニタを更新する．そのもメモリの故障によるハードウェアの交換時などにもホストマシンを一旦停止するこなる．ホストマシンを停止する場合，ホストマシン上の仮想マシンもシャットダウンてしまう．しかし仮想マシンでサービスを動かし続けている場合は，サービスを中断わけにはいかない．このような場合仮想マシンのサービスを停止することなく他のホマシンに移送する Live Migration が用いられる．Live Migration によってメンテナをしたいホストマシン上の全ての仮想マシンを他のホストマシン上に待避させることきる．全ての仮想マシン待避後，ホストマシンを停止しハードウェアのメンテナン行ったり，再起動を伴う更新などを行うことができる．(図 2)

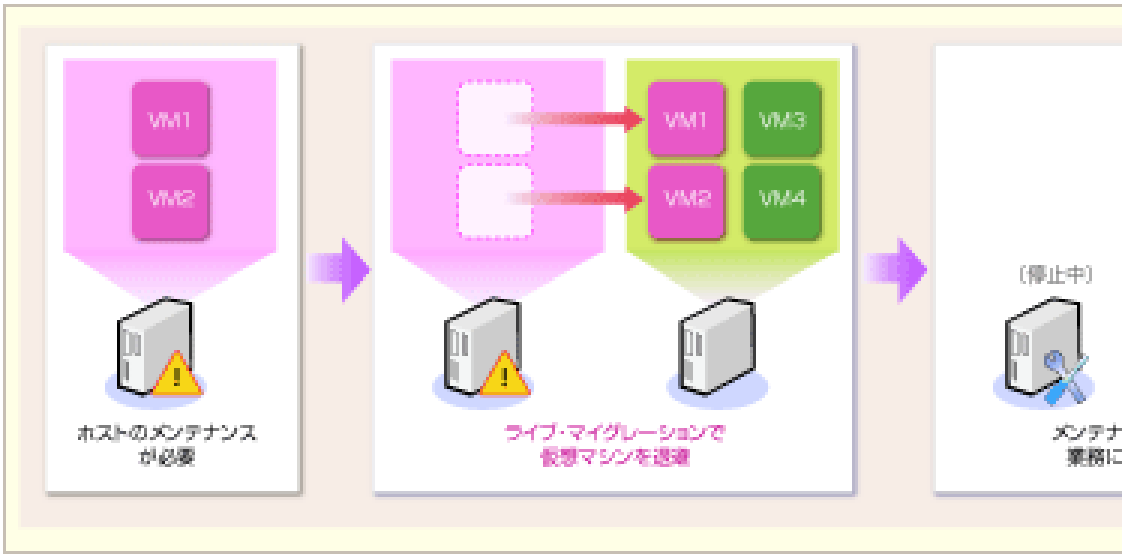

図 2 {

メンテナンス時 (引用：[13])}

2.3.3 移送する対象と方法

Live Migration を行うにあたっての移送する対象とその移送方法について説明す
まず仮想マシンを移送するということは，大きく二つにわけて，仮想マシンのディス
メモリを移送するということになる．この仮想マシンのディスクとメモリの移送方法
つにわかれる．一つはディスクとメモリ両方を移送先に移送する方法で，もう一つは
先とディスクを共有しておきメモリだけ移送する方法である．

- 両方とも移送先に移送する
  まず一つ目の移送方法は，仮想マシンのメモリもディスクもどちらも移送
  方法である．マシンの配置例は図 3 のようになる．ここではメモリとディ
  と両方を Live Migration する機能をもつ vSphere を例に説明する．この
  vSphere の移送元ホストマシンは移送先と移送元ホストマシン同士の vMo
  (Live Migration) 用の回線を使いまずストレージを移送先ホストマシンに移
  る．その後メモリを移送先に移送することで移送が完了する．詳しい移送の仕
  については後述する．ディスクを共有する方法では共有ディスクを用意しなく

圧迫したり，移送時間が長い．他にも citrix の XenServer の Storage XenMo
や Microsoft の Hyper-V の Shared Nothing Live Migration などがある．[1

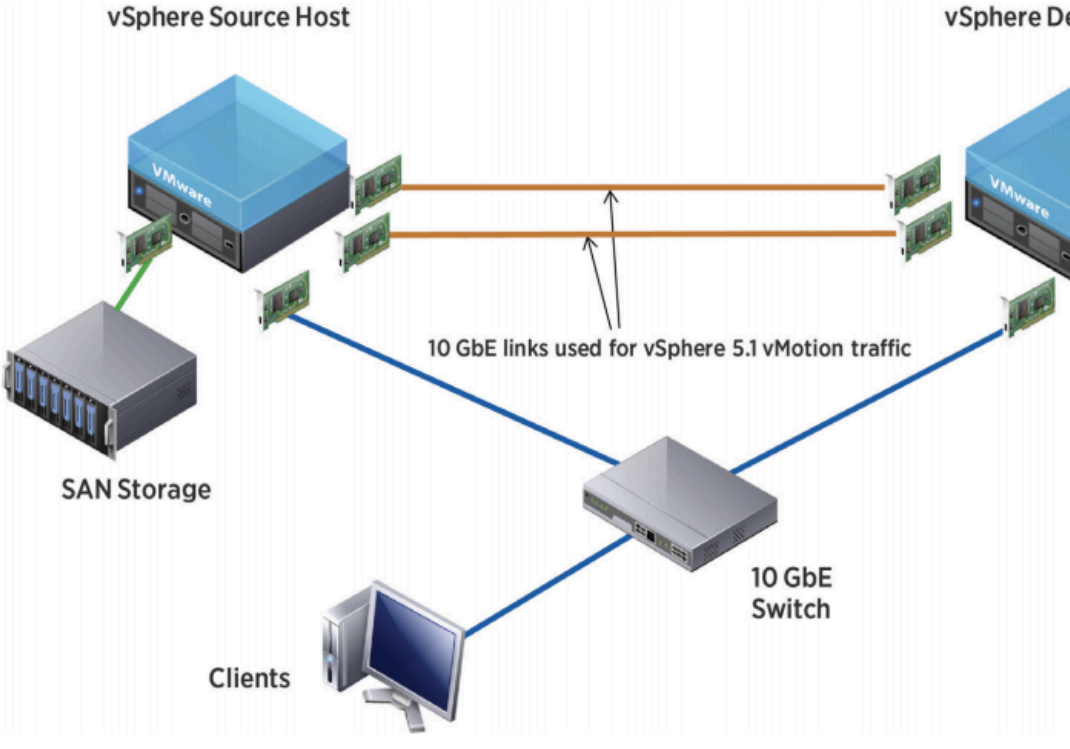

図 3 {

マシン配置例 1 (引用：[11])}

- ディスクを共有しメモリのみを移送する
  もう一つの方法は図**??**の様にストレージを共有し，メモリのみを移送先に移送
  方法である．移送の流れとしては (1) メモリを移送先に移送し，(2) 移送先の
  マシンを稼働させ，ストレージは共有ストレージを使用するようにするという
  になる．この方法では共有ストレージを用意しなくてはならないが移送するの
  モリだけなのでディスクとメモリ両方を移送する方法よりも大きく移送時間を
  すことができる．共有の方法には NAS(Network-attached storage) などがあ
  れる．本提案での Live Migration はこの方法をとる．

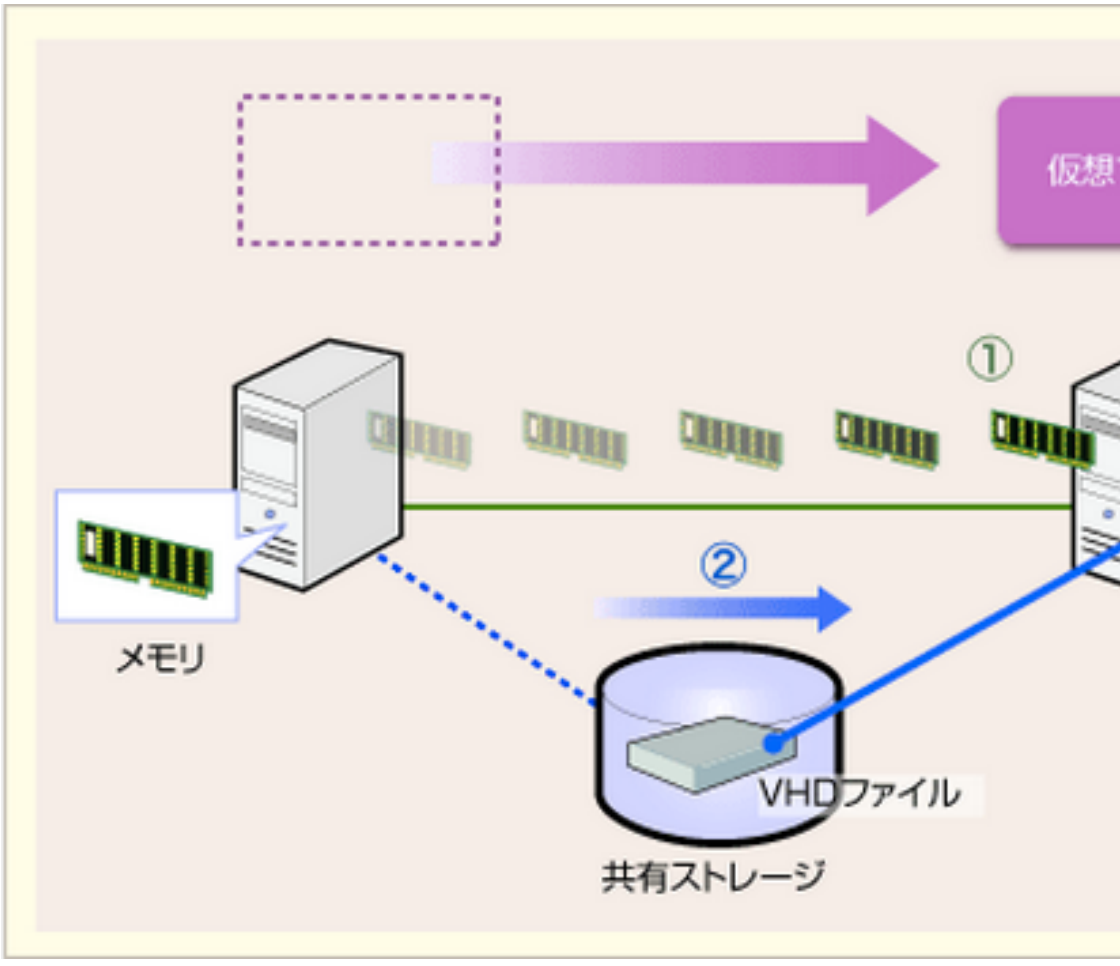

図 4　マシン配置例 2 この図は書き換える

2.3.4　移送の仕組み

- ストレージの移送

  本提案ではディスクを共有した，メモリのみの移送をする Live Migration を
  するので，ストレージの移送方法については vSphere の移送方法のみを参考と
  説明する．vSphere のディスク移送には並列して二つのプロセスが稼働する．
  5) 一つはストレージ全体を線形的に移送先にコピーする Bulk Copy プロセス
  もう一つは IO をミラーリングするプロセスである．IO ミラーリングのプロ
  は OS の IO 操作を監視して，IO 操作を移送元ホストの現在使用中のストレ

リングを行い非同期的に行うかというと，同期的に行うと，ネットワークのレンシによっては IO がとても混雑してしまい，VM の性能を下げてしまうことるからである．Bulk Copy プロセスが全てのストレージを移送先にコピーするBulk Copy プロセスは終了し，IO ミラーリングプロセスのみが稼働し続けたでメモリの移送を開始する．

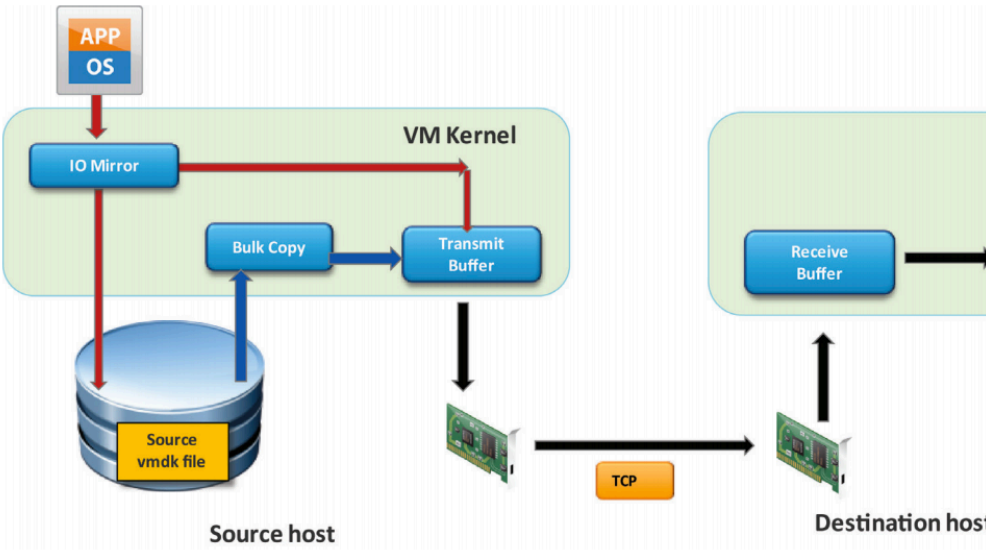

図 5　ストレージ vMotion

- メモリの移送
  メモリの移送は主に pre-copy が使用されている．[15] vSphere[4] の v tion(vSphere 特有の Live Migration の機能の名前)[11] や Hyper-V[6] の Migration [12]，オープンソースの Xen などで実際に pre-copy が使用される．ここでは pre-copy について説明する．pre-copy では六つのステージにて Live Migration を行う．動作の流れを図 6に示す．ステージ 0 では 移送し VM を稼働することができる程の計算資源を持った移送先ホストマシンを選る．ステージ 1 では移送先ホストに移送を開始することを通知する．また送信きる程の計算資源があるかどうか確認する．ステージ 2 ではメモリを繰り返し

ジを記録し，後続のイテレートとそのページを再送する．この移送中に更新さ
メモリを再送するという動作を，管理者が定めた一定の閾値に達するまで繰
す．閾値にはメモリの再送フェーズを行う回数の最大値や，再送するページ
の最小値などが用いられる．ステージ 3 では仮想マシンをサスペンドし，残
dirty になったページや CPU レジスタやプログラムカウンタなどを移送する
の移送が終わった状態では，移送元ホストマシンと移送先ホストマシンに同様
想マシンイメージがある状態になる．ステージ 5 では移送先ホストが移送元ホ
に移送が終了したことを通知する．この通知により移送元ホストは整合性のと
メモリの移送を完了したとして，移送元の仮想マシンを廃棄する．よって仮想
ンのイメージは移送先のもののみになる．ステージ 6 では移送した仮想マシン
開する．この時デバイスドライバの設定や，仮想マシンの IP アドレスの設定
を行う．

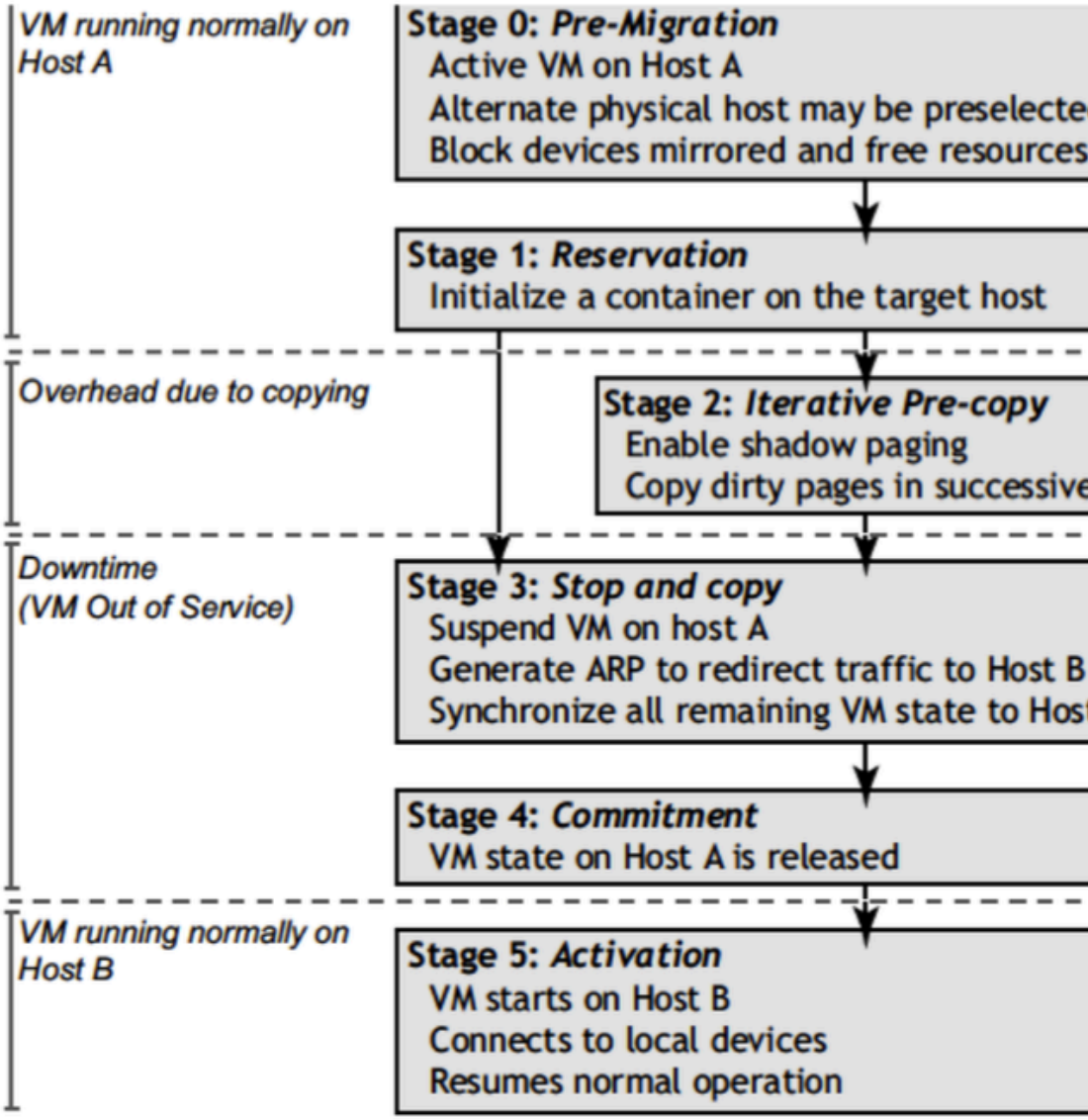

図 6　pre-copy の流れ

2.4　仮想マシン間のページ共有技術

仮想化技術により一つの物理マシンに複数台の仮想マシンが存在する場合，各々の

2.4.1 Transparent Page Sharing

仮想マシン間で重複した内容のメモリページを共有する技術は Disco[17] で *transpa page sharing* として導入された．同じ OS が起動していた場合，テキストデータ領 どが共有できる場合がある．他にも同じアプリケーションを動かしているときそのア ケーションのテキストデータ領域や，場合によっては使用するデータも共有できる場 ある．そのような場合重複したメモリページに対して一つの物理ページを用意して や どのメモリページもその一つの物理ページを参照する様にすればメモリの使用量を減 ことができる．重複排除したメモリページは read-only に設定しておき，書き込みを た時に page fault が起こるようにしておく．page fault が起きた場合，新しくペー 作った上で変更した内容のページを作成する．この機能は仮想マシン・モニタに実装 ておりゲストマシン OS にとっては名前の通り透過的なメモリシェアとなっている．

2.4.2 **メモリ共有**

ここではメモリを効率的に共有する方法 [16] を説明する．メモリを共有するとき 4kbyte のページ内容を全て比較していると計算量はかなり多くなってしまう．また ページ同士を総当たりで比べるとなるとページ数の自乗回の比較が必要になってし そこで効率的な比較をする方法としてメモリ内容から計算されるハッシュ値による シュテーブルの使用があげられる．(図 7) 具体的には，メモリの比較時にメモリの からハッシュ値を計算してハッシュテーブルから同じハッシュ値をもつページを引く し同じハッシュ値を持つページが存在すればページの内容を比較して，同じならば (Transparent Page Sharing) で共有する．もし同様のハッシュ値がなかった場合は シュテーブルに登録だけして終わる．このようにすることでページの内容が違う場 ハッシュ値の比較だけで済むため比較の計算量が減る．またページの比較回数もペー 自乗回かかっていたものがハッシュテーブルを利用して同じハッシュ値のみを検索し することで検索回数は大幅に減らすことができる．検索が O(1) の理想的なハッシュ ブルを作成した場合は検索回数はページ数回で済むことになる．しかし，仮想マシン ニタがラージページに対応しているとページサイズは 4kB から 2 MB と 512 倍に 上がるためページ共有の機会がとても少なくなる．その場合共有をあまりしないのに シュ値を計算するために CPU を使用することになる．Microsoft は仮想マシンのペ

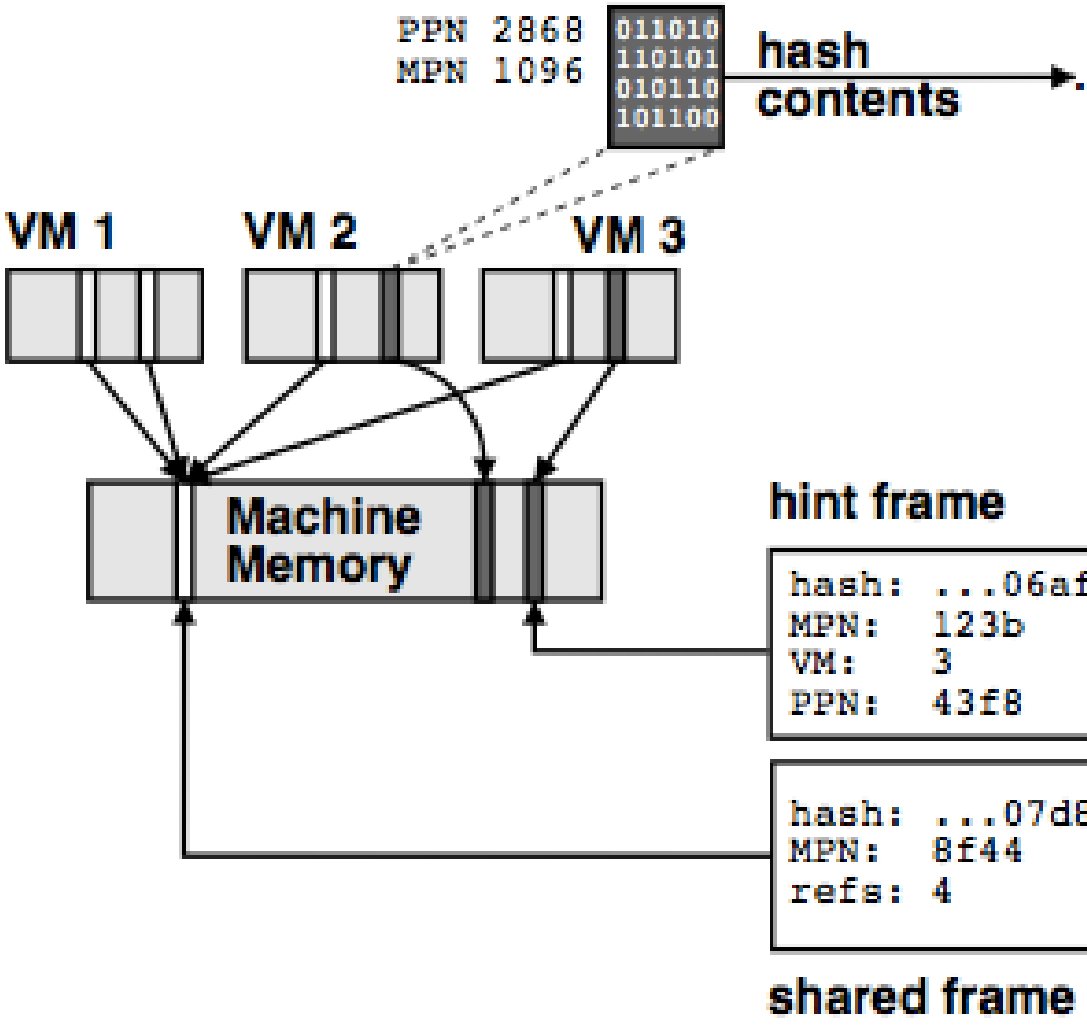

図７　ハッシュテーブルによるメモリ比較)

3 関連研究

3.1 MiyakoDori

ここはポストコピーにするか [Akiyama et al. IEEE Cloud' 12]

3.2 Towards Unobtrusive VM Live Migration for Cloud Computing P forms

ここは古藤さんの論文みせてもらおう [Koto et al. APSYS' 12]

3.3 まとめ

移送時のメモリ削減手法は色々あるが PaaS 環境における再利用可能なページを転しない手法がない．他の手法と比べ自分のを使うとどのようになるかの事実を述べる

4 提案手法

4.1 概要

PaaS 環境においては dst 側に転送したいページと同じページが存在する可能性い．(この言い方は主観的) 同じ設定の VM を src と dst で稼働させているような I 環境において，ページ転送を削減する．といった感じで書く．移送時間が減ると dirt 減って嬉しいってことあたりはここにかく

4.2 効率的な非移送ページの探索

VM 間での共有は常にハイパーバイザーで行われている (ということにして)．Paa 境では移送元で共有できているページは移送先にもある可能性が高いため既に共有済ページは移送先でも共有できる可能性が高いこと述べる．
共有ページだけみれば本提案で削減したいページについてはとても効率的に見つけれとができることを述べる．本当はハッシュコリジョンについての論文のリファーなどこでしたい

5 実装

がっつり書く

5.1 全体像

xen を用いることを述べる.
非転送メモリの決定フェーズを既存のライブマイグレーションの途中に設けて，それは既存のマイグレーションを（pre-copy) を行う.
モジュールの全体像も書く

5.2 （メモリ共有モジュール）

今回の実装についてはメモリ共有モジュールでのメモリ管理などが結構な量を占めるので，実装でページ数を稼ぐならメモリ共有モジュールも自分で作ってこれが大変たなど書けるのですが，本提案についてメモリ共有モジュールは本質的な実装ではなで書かないほうが良いでしょうか？

5.3 非転送メモリの確認方法

非転送ページ候補として共有時に作られたハッシュ値を用いて移送先に同じページるかどうかを確認する．そのために作ったハイパーコールとかも少し紹介する (?)

5.4 非転送ページの共有

非転送ページを確定したのち，移送先でページを共有することによって，ページの量を減らす

5.5 Live Migration への組み込み

既存の実装の dirty の管理方法，移送先でのアロックしたページの管理方法を説て，本提案実装が辻褄の合う様に組み込めていることを説明する

5.6　実験のスクリプト関連の苦悩とかも書いていいのかな笑

6 実験

6.1 非転送ページの量を調整した本実装の実験

6.1.1 目的

本提案で CPU やネットワークの消費を抑えることを示す．移送時間が減ることも

6.1.2 実験方法

PaaS 環境を想定して，各 VM に同じページをもたせて本提案の実験をした．
同じページの量は調整できる．

6.1.3 実験結果

費転送ページが多い程移送時間が短くなり CPU やネットワークの使用時間も削減ているので浪費を抑えることができた．
(移送時間が短くなったのでその分 CPU やネットワークの浪費は抑えられたというの持っていきかたは強引すぎるでしょうか．)

7 おわりに

7.1 conclusion

7.2 今後の課題

- 実際の PaaS 環境を想定した memcached などを用いた実験をする
- VM のメモリ量を増やしても同様に動作するかなどの検証

8 謝辞

9 参考文献

参考文献

[1] VirtualBox https://www.virtualbox.org/

[2] 仮想化入門 http://www.plathome.co.jp/solution/virtualserver/intro

[3] 浅見直樹 日経 BP 社「仮想化大全」 p14-17

[4] vSphere http://www.vmware.com/jp/products/vsphere

[5] XenServer http://www.citrix.co.jp/products/xenserver/xenserver.h

[6] Hyper-V http://www.microsoft.com/ja-jp/server-cloud/windows-ser

[7] Xen http://xenproject.org/

[8] KVM http://www.linux-kvm.org/page/Main_Page

[9] IT 用語辞典 http://e-words.jp/

[10] Azure http://azure.microsoft.com/ja-jp/

[11] Vmware vSPhere 5.1 vMotion Architecture, Performance, and Best Practice

[12] Hyper-V のライブマイグレーションの説明
http://technet.microsoft.com/ja-jp/library/hh831435.aspx

[13] Hyper-V 2.0 のライブ・ライブマイグレーションの基礎知識
http://www.atmarkit.co.jp/ait/articles/0912/16/news102.html

[14] Storage XenMotion: Live Storage Migration with Citrix XenServer

[15] Christopher Clark, Keir Fraser, Steven Hand, Jacob gorm Hansen, Eric
Christian Limpach, Ian Pratt and Andrew Warfield. Live migration of vir
machines, NSDI'05 Proceedings of the 2nd conference on Symposium on
worked Systems Design & Implementation - Volume 2, Pages 273-286 , 2005

[16] Carl A. Waldspurger. Memory Resource Management in VMware ESX Ser
Fifth Symposium on Operating Systems Design and Implementation (OSI
02), Pages 181-194, Dec. 2002.

[17] Edouard Bugnion, Scott Devine, Kinshuk Govil, and Mendel Rosenblum. "D

とりあえず見つけた論文少し目を通そうか