Vol.2013-OS-127 No.14
Vol.2013-EMB-31 No.14
2013/12/3

ネストしたVMを用いた仮想化システムの高速なソフトウェア若化

大庭 裕貴[1]　光来 健一[1,2]

概要：仮想化システムにおけるソフトウェア・エージングにはソフトウェア若化で対処することができる．ソフトウェア若化の典型的な例はシステムの再起動であるが，その際の仮想マシン（VM）のダウンタイムを削減するには，VM をあらかじめマイグレーションしておけばよい．しかし，マイグレーションはシステムに大きな負荷をかけてしまい，そのシステム上で動作する VM の性能にも影響を及ぼす．そこで本稿では，ネストした VM を用いて，低負荷で高速なマイグレーションを可能とする *VMBeam* を提案する．VMBeam では，同一ホスト上で二つの仮想化システムを動作させ，一方の仮想化システムを再起動する際にはすべての VM をもう一方の仮想化システム上にマイグレーションする．この際に，VM のメモリの内容を仮想ネットワーク経由で転送する代わりに，同一ホスト上にあることを利用して VM 間でメモリをコピーまたはスワップする．VMBeam を Xen に実装し，マイグレーションの時間の短縮および CPU 負荷の低下を確認した．

Fast Software Rejuvenation of Virtualized Systems Using Nested Virtualization

Hiroki Ooba[1]　Kenichi Kourai[1,2]

Abstract: Software rejuvenation can counteract software aging in virtualized systems. Its typical example is a system reboot. To reduce the downtime of virtual machines (VMs) during the reboot, VMs can be migrated in advance. However, VM migration stresses the system and affects the performance of the VMs running on top of it. In this paper, we propose *VMBeam*, which enables fast migration with low load using nested virtualization. VMBeam runs two virtualized systems at the same host. When one virtualized system is rejuvenated, VMBeam migrates the VMs to the other. At this time, VMBeam copies or swaps memory between VMs, which are located on the same host, instead of transferring memory contents via the virtual network. We have implemented VMBeam in Xen and confirmed the reduction of migration time and a drop of CPU utilization.

1. はじめに

クラウドコンピューティングをはじめとして，様々なシステムで仮想化が行われるようになってきている．このような仮想化システムでは一台の計算機の中で複数の仮想マシン（VM）を動作させることによって，物理マシンの台数を減らし，コストを削減することができる．しかし，仮想化システムは仮想化を行っていない従来のシステムと比べるとより長時間連続で運用されることが多いため，ソフトウェア・エージング [1] と呼ばれる現象が発生しやすくなる．ソフトウェア・エージングとは動作しているソフトウェアの状態が次第に劣化していく現象であり，メモリリークなどが原因として挙げられる．ソフトウェア・エー

[1] 九州工業大学
Kyushu Institute of Technology
[2] 独立行政法人科学技術振興機構 CREST
JST, CREST

ジングが起こるとシステムの性能が次第に低下していき，最悪の場合には想定外のシステムダウンが発生する可能性もある．

ソフトウェア・エージングの問題に対処するために，ソフトウェア若化と呼ばれる手法が提案されている [1]．ソフトウェア若化はシステムの状態をソフトウェア・エージングが起こる前の状態に戻すための手法である．ソフトウェア若化を行うことでシステムの性能を回復することができ，システムダウンを防ぐことができる．ソフトウェア若化の典型的な例はシステムの再起動である．しかし，仮想化システムを再起動するには，動作しているすべての VM を一旦停止させ，仮想化システムの再起動後に再び起動し直さなければならない．多くの VM の停止や起動には時間がかかるため，その間，サービスを提供できないダウンタイムが生じる．

このダウンタイムを削減するには，VM を動作させたまま別のホストに移動させるマイグレーションを用いることができる．仮想化システムを再起動する前にすべての VM を別のホストにマイグレーションしておけば，VM 上のサービスは再起動の影響を受けない．しかし，マイグレーションを行うにはネットワークを介した大量のデータ転送が必要となるため，ホストやネットワークに大きな負荷をかけてしまう．一方で，システム全体への影響を考えて，マイグレーション速度を抑えた場合には，ソフトウェア若化の完了までに時間がかかってしまう．

本稿では，ネストした VM（Nested Virtualization）を用いて，低負荷で高速なマイグレーションを可能とする *VMBeam* を提案する．VMBeam では，同一ホスト上で二つの仮想化システムを動作させ，一方の仮想化システムのソフトウェア若化の際にはもう一方の仮想化システムに VM をマイグレーションする．その際に，これらの仮想化システムが同一ホスト上にあることを利用して VM のメモリ転送を高速化する．我々は VMBeam を Xen 4.2 [3] に実装した．VMBeam を用いてメモリのコピーまたはスワップによるマイグレーションを行うことにより，通常のマイグレーションよりも低負荷で高速に行えることを確認した．

以下，2 章で仮想化システムの従来のソフトウェア若化の問題について述べ，3 章で VMBeam を提案する．4 章で VMBeam の実装について説明し，5 章で VMBeam を用いて行った実験について述べる．6 章で関連研究に触れ，7 章で本稿をまとめる．

2. 仮想化システムのソフトウェア若化

仮想化システムは長時間動作するソフトウェアであるため，ソフトウェア・エージング [1] が発生しやすい．ソフトウェア・エージングはメモリの解放し忘れや，オープンしたファイルの閉じ忘れなどのバグのために，システムの性能が次第に低下していく現象である．Xen において，VM

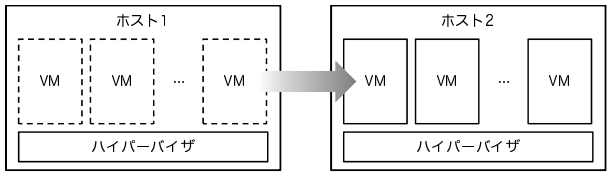

図 1 マイグレーションを活用したソフトウェア若化

のマイグレーションを繰り返すとドメイン 0 の空きメモリが減少していくことが報告されている [2]．また，VM のサスペンド・レジュームを繰り返すとドメイン 0 の空きディスク容量が減少していくことも報告されている．

このようなソフトウェア・エージングの問題に対処するために，ソフトウェアを正常な状態に戻すソフトウェア若化 [1] と呼ばれる手法が提案されている．その最も単純な方法として，システムを再起動することでソフトウェア若化を行うことができる．仮想化システムの場合，ハイパーバイザの再起動が必要となるが，その際にはハイパーバイザ上で動作しているすべての VM を一旦停止し，ハイパーバイザが起動してからすべて起動し直す必要がある．VM 上で動作している OS の再起動には時間がかかる上，近年の計算機の高性能化により多くの VM が集約されて動作している場合もある．そのため，VM 上で提供されているサービスのダウンタイムが長くなる傾向にある．

ソフトウェア若化に伴う VM のダウンタイムを削減するために，VM を動作させたまま別のホストに移動させることができるマイグレーションを活用することができる．ハイパーバイザのソフトウェア若化を行う前に，**図 1** のように，そのハイパーバイザ上で動作しているすべての VM をマイグレーションにより別のホストに移動する．その際の VM のダウンタイムはライブマイグレーションを用いることにより抑えることができる．その後でハイパーバイザを再起動することで，VM に影響を与えることなくソフトウェア若化を行うことができる．

しかし，VM のマイグレーションはシステムやネットワークに大きな負荷がかかる．VM をマイグレーションするには移動元のホストから移動先のホストに VM のメモリ内容を転送する必要がある．このメモリの転送は，すべての VM を移動させる場合には合計で数 GB から数十 GB ものデータ量になることもある．そのため，移動元と移動先のホストの CPU やメモリ帯域を占有することにより，ホストのシステムの性能低下を引き起こす．また，マイグレーション専用のネットワークを用意していない場合は，マイグレーション処理がネットワーク帯域を占有してしまい，ホストのネットワーク性能に大きな影響を及ぼす．これらの負荷により，そのホスト上で動作している VM の性能も低下してしまう．

このようなマイグレーション中のシステム全体の性能低下を抑えるには，マイグレーションの速度を抑える方法が

Vol.2013-OS-127 No.14
Vol.2013-EMB-31 No.14
2013/12/3

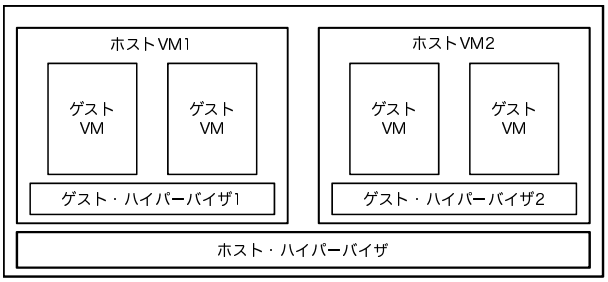

図 2 ネストした VM のシステム構成

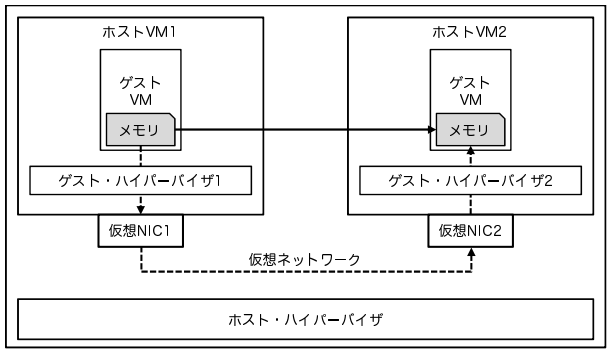

図 3 VM 間メモリコピーによるメモリ転送

考えられる．マイグレーションで用いるネットワーク帯域を制限することで，ネットワークへの負荷を軽減し，その結果，システムへの負荷も軽減することができる．その一方で，マイグレーションに時間がかかるようになることにより，ハイパーバイザの再起動が可能になるまでの時間が長くなり，仮想化システムのソフトウェア若化時間の増大につながる．

3. VMBeam

本稿では，仮想化システムの高速なソフトウェア若化を実現するために，ネストした VM を用いて低負荷で高速なマイグレーションを可能にする *VMBeam* を提案する．VMBeam では，同一ホスト上で二つの仮想化システムを動作させ，一方の仮想化システムを再起動する際には，もう一方の仮想化システム上に VM をマイグレーションする．この際に，VM のメモリ内容を仮想ネットワーク経由で転送する代わりに，同一ホストにあることを利用して VM 間でメモリを高速に転送する．この手法により，マイグレーションの高速化および負荷の軽減を図る．

3.1 ネストした VM の活用

同一ホスト上で二つの仮想化システムを動作させるために，VMBeam ではネストした VM を利用する．ネストした VM を用いると，VM の中で一つの仮想化システムを動作させることができる．ネストした VM を用いたシステム構成を**図 2** に示す．区別のために，通常の仮想化システムにおけるハイパーバイザ，VM をそれぞれホスト・ハイパーバイザ，ホスト VM と呼び，ホスト VM 内で動作するものをそれぞれゲスト・ハイパーバイザ，ゲスト VM と呼ぶ．

本稿では，ゲスト・ハイパーバイザをソフトウェア若化の対象とする．VM のマイグレーションやサスペンド，レジュームなどの操作を行う際にソフトウェア・エージングが起こりやすい [2] ため，それらの操作を行うゲスト・ハイパーバイザのほうが頻繁にソフトウェア若化を必要とすると考えられる．一方，ホスト・ハイパーバイザはこのような操作を行わないため，比較的，ソフトウェア・エージングが起こりにくいと考えられる．ホスト・ハイパーバイザのソフトウェア若化を行う際には，従来と同様に，ホスト VM を別のホストにマイグレーションしてから行う．

3.2 高速な VM マイグレーション

VM のマイグレーションを行う際には，移動元の仮想化システムから移動先の仮想化システムにマイグレーション要求が出される．移動先の仮想化システムはマイグレーションされてきた VM を復元するために，メモリや CPU などを初期化した空の VM を作成する．移動元の仮想化システムはマイグレーションする VM のメモリ内容や CPU の状態を仮想ネットワークを介して移動先の仮想化システムに転送する．そして，移動先では受け取ったメモリ内容や CPU の状態を空の VM に書き込むことで VM の移動を実現する．ネストした VM においては，VM の移動元と移動先が同一ホストであるため，仮想ネットワークを用いたデータ転送を高速に行える可能性があるが，ネットワーク仮想化のオーバヘッドも大きくなる．また，移動元と移動先の処理を同一ホストで行うため，システムへの負荷は通常のマイグレーションの 2 倍になる可能性がある．

VMBeam は，マイグレーションにおけるメモリ転送の高速化のために 2 つの手法を提供する．1 つ目は ***VM* 間メモリコピー**であり，ホスト・ハイパーバイザが移動元のゲスト VM のメモリを移動先の空のゲスト VM に直接コピーする．VM 間メモリコピーは，仮想ネットワークによるデータ転送の高速化である．VM 間メモリコピーによる転送を**図 3** に示す．VM 間メモリコピーを用いることでホスト VM 間での仮想ネットワークのエミュレーションが不要となる．また，セマンティクスは仮想ネットワークを使用した転送と同等であるため，ライブマイグレーションも可能である．しかし，通常のマイグレーションと同様に移動元と移動先でゲスト VM のためのメモリ領域を確保する必要があるため，マイグレーション中は VM の 2 倍のメモリが必要となる．

2 つ目の手法は ***VM* 間メモリスワップ**であり，ホスト・ハイパーバイザが移動元と移動先のゲスト VM のメモリを入れ替える．この手法は仮想ネットワークによるデータ転送の単純な高速化ではないが，移動先のゲスト VM が空

Vol.2013-OS-127 No.14
Vol.2013-EMB-31 No.14
2013/12/3

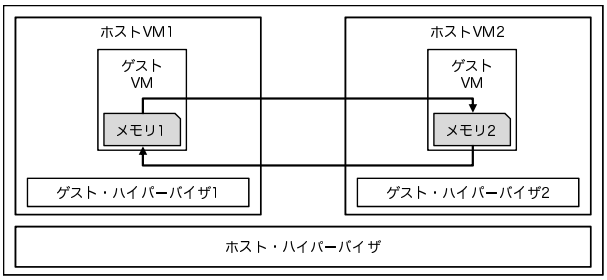

図 4 VM 間メモリスワップによるメモリ転送

であるため，マイグレーションとほぼ同等の効果を得ることができる．VM 間メモリスワップによる転送を**図 4** に示す．VM 間メモリスワップを用いることで，ゲスト VM のメモリをコピーする必要がなくなる．また，移動先でゲスト VM のメモリ領域を確保せずに済ませる（メモリフリップ）ことも可能になり，メモリの節約が可能である．しかし，一部でもメモリの入れ替えを行うと，移動元のゲスト VM は動作し続けられなくなるため，ライブマイグレーションを行うことは難しくなる．

4. 実装

4.1 Xen におけるネストした VM

我々は Xen 上で Xen を動作させることでネストした VM を構築した．ホスト・ハイパーバイザとして Xen 4.2 を動作させ，その上でホスト VM を HVM（完全仮想化）ゲストとして動作させた．次に，ホスト VM 上でゲスト・ハイパーバイザとして Xen 4.2 を動作させ，その上でゲスト VM を HVM ゲストとして動作させた．Xen では VM を管理するためにドメイン 0 と呼ばれる管理 VM が用いられるが，ホスト VM を管理する VM をホスト管理 VM，ゲスト VM を管理する VM をゲスト管理 VM と呼ぶ．

ネストした VM におけるメモリモデルを**図 5** に示す．Xen おいて，ホスト・ハイパーバイザはマシンメモリと呼ばれるマシン全体で管理している物理メモリを扱い，マシンメモリの一部をホスト VM に割り当て，疑似的な物理メモリとして扱わせる．これをホスト物理メモリと呼ぶ．ゲスト・ハイパーバイザはこのホスト物理メモリの一部をさらにゲスト VM に割り当てる．これをゲスト物理メモリと呼ぶ．それぞれのメモリにはメモリフレーム番号が振られ管理される．マシンメモリにはマシンフレーム番号（MFN），ホスト物理メモリにはホスト物理フレーム番号（HPFN），ゲスト物理メモリにはゲスト物理フレーム番号（GPFN）が振られる．MFN と HPFN，HPFN と GPFN の間の対応はそれぞれホスト・ハイパーバイザ，ゲスト・ハイパーバイザで管理される．

4.2 コピー・マイグレーション

VMBeam では，仮想ネットワークによるデータ転送の代わりにメモリ操作だけを行うことでマイグレーションの

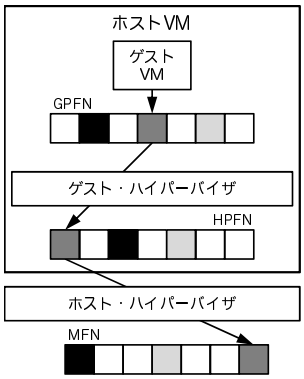

図 5 ネストした Xen におけるメモリモデル

高速化および負荷軽減を行う．このマイグレーションの流れを**図 6** に示す．Xen では，ゲスト VM のマイグレーションはマイグレーション元とマイグレーション先のゲスト管理 VM 間で行われる．従来のマイグレーションでは，マイグレーション元のゲスト管理 VM がゲスト VM のメモリ内容をネットワーク経由でマイグレーション先に送っていた．VMBeam ではその代わりに，ゲスト VM に割り当てられているホスト物理メモリの情報（HPFN）をホスト管理 VM で動作するメモリサーバに送る．ゲスト管理 VM が HPFN を取得できるようにするために，ゲスト・ハイパーバイザにハイパーコールを追加し，ゲスト VM の GPFN を HPFN に変換できるようにした．同時に，マイグレーション先のゲスト管理 VM は新たに作成したゲスト VM に割り当てたホスト物理メモリの HPFN をメモリサーバに送る．

これらのメモリ情報を受け取ったホスト管理 VM 上のメモリサーバは，ホスト・ハイパーバイザを呼び出すことによってゲスト VM 間で VM 間メモリコピーを行う．VMBeam では，Xen の標準の機能を用いて VM 間メモリコピーを実現した．メモリサーバで受け取った各ゲスト VM の HPFN を指定してゲスト VM のメモリページをホスト管理 VM のメモリ空間にマップし，移動元のゲスト VM のメモリ内容を移動先の空のゲスト VM のメモリにコピーする．現在の実装では，ライブマイグレーションには対応しておらず，ゲスト VM を停止させて全メモリのコピーを行っている．ゲスト VM を動作させたままコピーを行い，最終段階で差分のコピーを行うようにすることで，

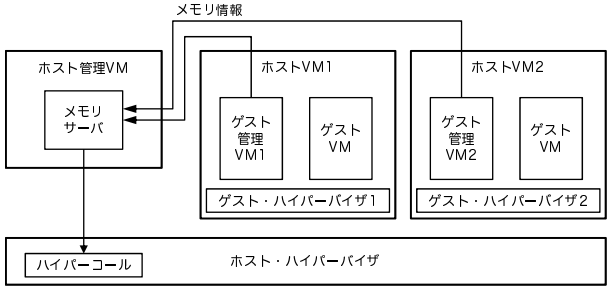

図 6 メモリ操作によるマイグレーション

Vol.2013-OS-127 No.14
Vol.2013-EMB-31 No.14
2013/12/3

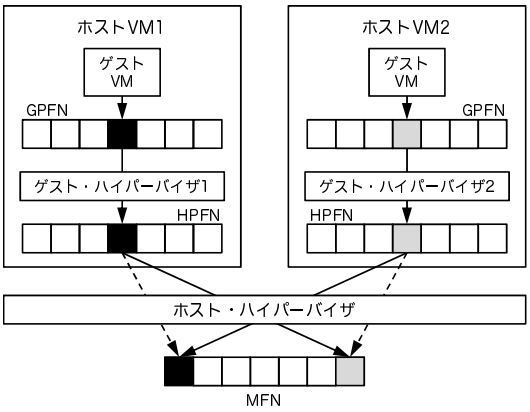

図 7 ホスト VM 間でのメモリスワップ

ライブマイグレーションに対応することもできると考えられる．

4.3 スワップ・マイグレーション

VM 間メモリスワップを用いたマイグレーションの流れもコピー・マイグレーションとほぼ同じである．VMBeam では，ホスト・ハイパーバイザがホスト VM 間でメモリを高速に入れ替えることで VM 間メモリスワップを行う．この機能はホスト・ハイパーバイザが二つのホスト VM へのメモリ割り当てを変更することで実現する．メモリ割り当てを変更するために，ホスト物理メモリとマシンメモリの間の対応を変更する必要がある．そこで，ホスト・ハイパーバイザが管理している，ホスト物理メモリからマシンメモリへの対応表である P2M テーブルを書き換え，**図 7** のようなメモリ割り当てに変更することで，二つのホスト VM 間でメモリを入れ替える．Intel の CPU の仮想化支援を用いる場合，P2M テーブルは EPT を用いて実装される．また，マシンメモリからホスト物理メモリへの対応表（M2P テーブル）もホスト・ハイパーバイザが管理しており，P2M テーブルを書き換える際には，M2P テーブルの書き換えも行う．

VM 間メモリスワップを行う際には，ゲスト VM を停止させてから，移動元と移動先のゲスト VM のメモリに関するエントリを P2M テーブルと M2P テーブルから削除する．そして，それらのエントリを逆にして移動元と移動先の P2M テーブルと M2P テーブルに追加し直す．加えて，そのページの所有者を入れ替え先のゲスト VM に変更する．また，ホスト VM に割り当てられているメモリの一覧もリストで管理しているため，入れ替えたメモリに応じてこのリストも更新する．

5. 実験

まず，ネストした VM を用いることによるオーバヘッドと仮想ネットワーク性能を計測する実験を行った．次に，マイグレーションの性能を調べるため，仮想ネットワークを用いたマイグレーション，コピー・マイグレーション，スワップ・マイグレーションそれぞれにおけるマイグレーション時間，CPU 負荷および，ダウンタイムを測定する実験を行った．実験には Intel Xeon E5-2665（8 コア，2.40GHz）の CPU，32GB のメモリ，1TB の HDD を搭載したマシンを用いた．ホスト管理 VM，ホスト VM，ゲスト管理 VM，ゲスト VM へのメモリ，CPU 割り当てを**表 1** に示す．ホスト・ハイパーバイザ，ゲスト・ハイパーバイザとしてともに Xen 4.2.2 を動作させ，ホスト管理 VM では Linux 3.2.0，ゲスト管理 VM では Linux 3.5.0 を用いた．本実験では**図 6** のように，ホスト・ハイパーバイザ上でホスト管理 VM と 2 つのホスト VM を動作させ，ホスト VM 上でゲスト管理 VM とゲスト VM を動作させた．

表 1 各 VM へのメモリ，CPU 割り当て

	メモリ [GB]	CPU
ホスト管理 VM	9.5	16
ホスト VM	10.0	6
ゲスト管理 VM	9.1	6
ゲスト VM	0.5	2

5.1 ネストした VM のオーバヘッド

UnixBench を用いて VM のネストによりどの程度の性能低下が生じるのかを調べた．Xen 4.2 を用いてネストしたゲスト VM とネストを行っていないホスト VM で UnixBench を実行した．また，比較のために，Xen-Blanket [4] を用いてネストしたゲスト VM についても UnixBench を実行した．Xen-Blanket ではゲスト VM を PV（準仮想化）ゲストとして動作させる点が異なる．UnixBenchn の各スコアを**図 8** に示す．この結果より，Xen 4.2 を用いてネストしたゲスト VM において演算処理，ファイルのコピー，システムコールのオーバヘッドに関しては性能低下はほとんどみられないが，execl 実行，プロセス生成，シェルスクリプト処理に関しては非常に性能が低下してしまうことが分かった．Xen-Blanket を用いてネストしたゲスト VM では，演算処理以外では 80%以上性能が低下することが分かった．ただし，仮想 EPT などを用いることで平均 20%程度の性能低下で済むという報告もされている [5].

5.2 仮想ネットワーク性能

マイグレーション時に利用するゲスト管理 VM 間のネットワーク性能を調べるために，Xen 4.2 におけるゲスト管理 VM 間，Xen-Blanket におけるゲスト管理 VM 間の仮想ネットワークの性能を比較する実験を行った．仮想ネットワークのスループットを iperf を用いて計測した値を**表 2** に示す．結果より，Xen 4.2 の仮想ネットワーク性能はギガビットイーサネットの性能よりも低いことが分かった．これはホスト管理 VM で NIC（e1000）のエミュレーショ

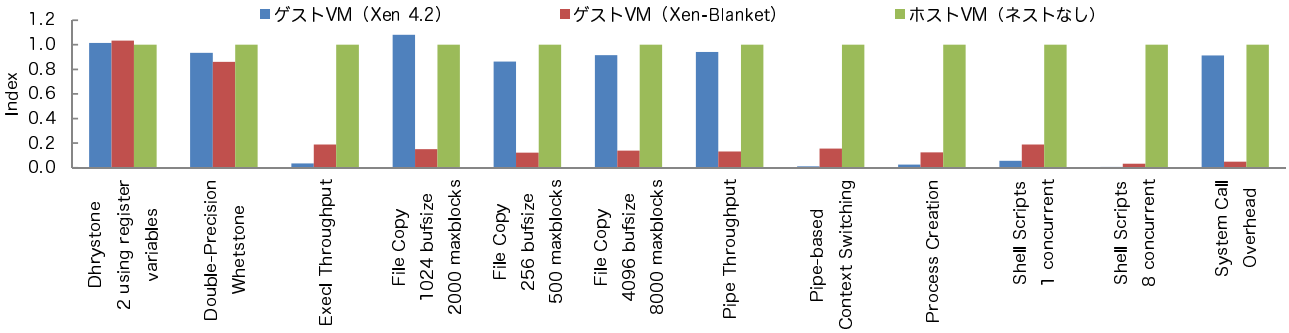

図 8　UnixBench によるスコア比較

表 2　仮想ネットワークの性能比較

	スループット [Mbps]
Xen 4.2	218
Xen-Blanket	1.0×10^4

ンを行うオーバヘッドによるものだと考えられる．一方，Xen-Blanket ではギガビットイーサネットの 10 倍の性能であった．これは Xen-Blanket のゲスト管理 VM では準仮想化ネットワークドライバが用いられており，ネットワーク仮想化のオーバヘッドが小さいためである．

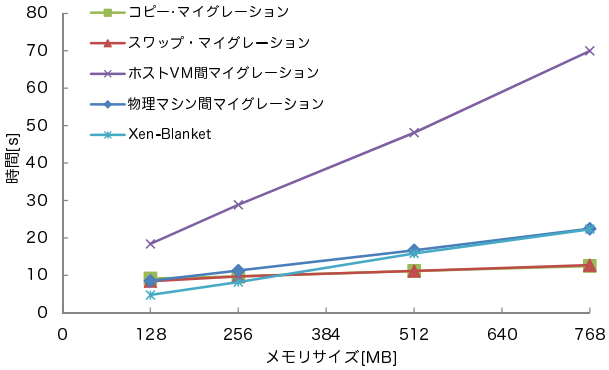

図 9　マイグレーション時間の比較

5.3　マイグレーションの動作テスト

コピー・マイグレーションおよびスワップ・マイグレーションの動作確認を行った．これらのマイグレーションによってゲスト VM を移動させた後もゲスト VM への SSH 接続を行うことができ，ゲスト VM が移動先のホスト VM 上で正常に動作を続けていることが確認できた．

5.4　マイグレーション時間

ゲスト VM に割り当てるメモリサイズを 128MB から 768MB まで増やした場合のマイグレーションにかかる時間を計測した．コピー・マイグレーション，スワップマイグレーション，ホスト VM 間での仮想ネットワークを用いたマイグレーション（Xen 4.2 および Xen-Blanket），および，物理マシン間の通常のマイグレーションにかかる時間をそれぞれ 5 回ずつ計測した平均値を**図 9** に示す．この結果より，どのマイグレーション手法でもマイグレーションにかかる時間はゲスト VM のメモリサイズに比例することが分かる．割り当てるメモリサイズが 768MB の場合でも，コピー・マイグレーションにかかる時間は 12.5 秒，スワップ・マイグレーションにかかる時間は 12.8 秒となった．ホスト VM 間での通常のマイグレーションと比較して，コピー・マイグレーション，スワップマイグレーションともに最大で 80%高速に行えることが分かった．また，物理マシン間での通常のマイグレーションと比較しても，スワップ・マイグレーション，コピーマイグレーションともに最大で 40%高速に行えることが分かった．

Xen-Blanket によるホスト VM 間での通常のマイグレーションはメモリサイズが小さいと最も高速であるが，メモリサイズが大きくなるとコピー・マイグレーションやスワップ・マイグレーションより時間がかかることが分かった．メモリサイズが 768MB の場合，コピーマイグレーションとスワップ・マイグレーションの方が 40%高速であった．ただし，Xen-Blanket は PV ゲストのマイグレーションとなるため，HVM ゲストをマイグレーションする他の手法とは単純に比較できない．

5.5　マイグレーション時の CPU 負荷

ゲスト VM に割り当てるメモリサイズを 128MB から 768MB まで増やした場合のマイグレーション中のマシン全体の CPU 負荷を計測した．コピー・マイグレーション，スワップ・マイグレーション，ホスト VM 間での通常のマイグレーション，物理マシン間での通常のマイグレーションの間の CPU 負荷の変化を**図 10** から**図 13** に示す．コピー・マイグレーション，スワップ・マイグレーションともに負荷最大時でもホスト VM 間での通常のマイグレーションよりも低負荷であることが分かる．また，物理マシン間での通常のマイグレーションと比較しても，より低負荷で行えていることが分かった．コピー・マイグレーションとスワップ・マイグレションを比較すると，負荷の高まるタイミングはずれているが，ほぼ同程度の負荷であることが分かった．

Vol.2013-OS-127 No.14
Vol.2013-EMB-31 No.14
2013/12/3

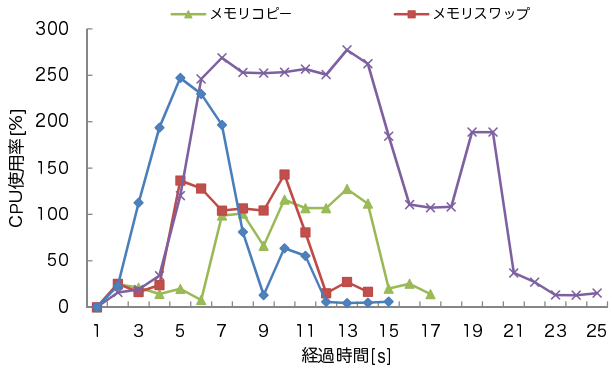

図 10 128MB 時の CPU 負荷

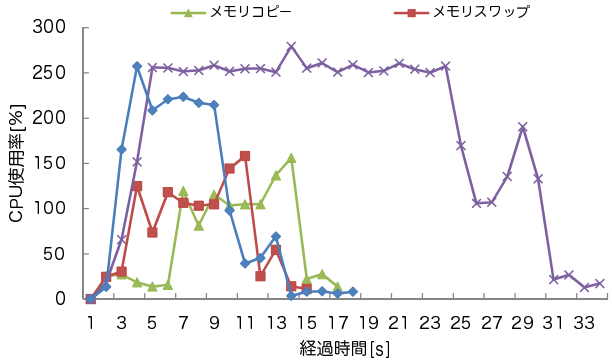

図 11 256MB 時の CPU 負荷

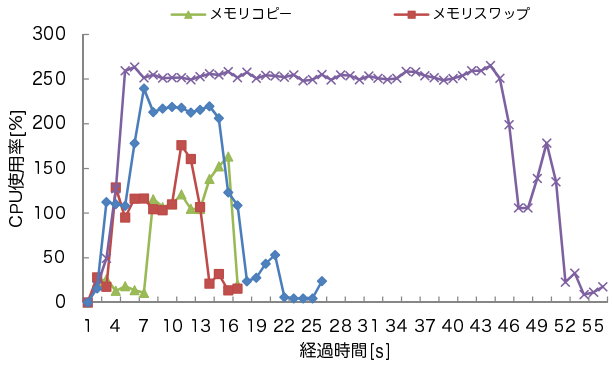

図 12 512MB 時の CPU 負荷

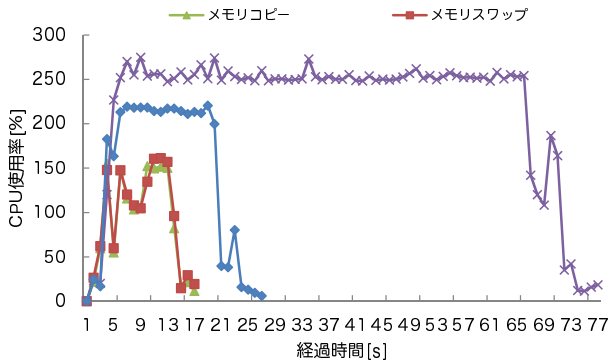

図 13 768MB 時の CPU 負荷

5.6 マイグレーション中のダウンタイム

ゲスト VM に割り当てるメモリサイズを 128MB から 768MB まで増やした場合のマイグレーション中のゲスト

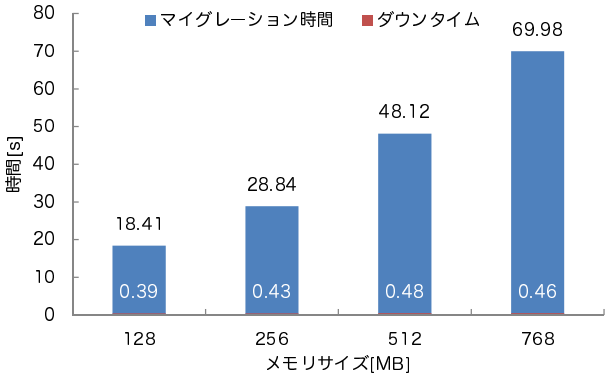

図 14 ホスト VM 間での通常のマイグレーション中のダウンタイム

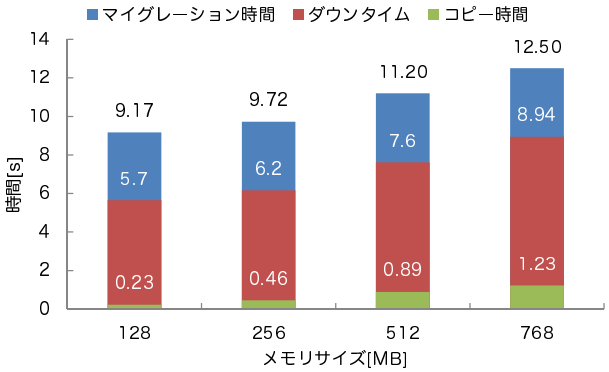

図 15 コピー・マイグレーション中のダウンタイム

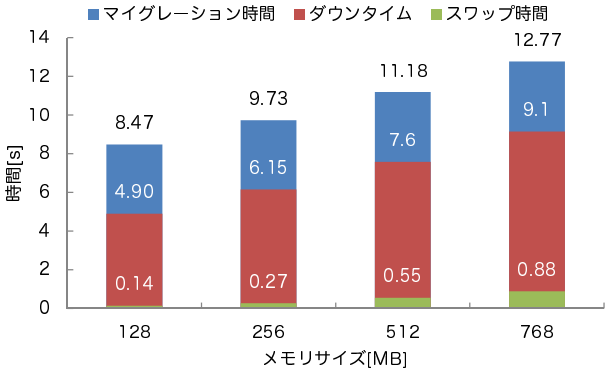

図 16 スワップ・マイグレーション中のダウンタイム

VM のダウンタイムを計測した．ホスト VM 間での通常のマイグレーション，コピー・マイグレーション，スワップ・マイグレーションを行っている間のマイグレーション時間に対するゲスト VM のダウンタイムをそれぞれ 5 回ずつ計測した平均値を**図 14**，**15**，**16** に示す．この結果より，ホスト VM 間での通常のマイグレーションではダウンタイムはほぼ生じていないことが分かった．また，コピー・マイグレーション，スワップ・マイグレーションのどちらを用いてもほぼ同じ時間のダウンタイムが生じることが分かった．ゲスト VM に割り当てるメモリサイズが 768MB の時，ダウンタイムは 9 秒程度であった．

その内，VM 間メモリコピー，VM 間メモリスワップに要している時間をみてみると 1 秒前後であることが分かっ

た．ダウンタイムの残り時間のほとんどは，ゲスト VM の情報をゲスト管理 VM 間で送信する時間および，メモリ情報をゲスト管理 VM とホスト管理 VM 間で送信する時間である．これらは仮想ネットワーク性能の影響を大きく受けており，仮想ネットワークを用いずに送信できるようにすることで大幅に短縮することが可能だと考えられる．

6. 関連研究

Xen-Blanket [4] はネストした VM をサポートしていない既存の仮想化システムの上でネストした VM を動作させることのできるシステムである．また，Xen-Blanket ではゲスト管理 VM に Blanket ドライバと呼ばれる準仮想化ドライバを導入することで，VM をネストすることによる I/O のオーバーヘッドを削減している．本研究でも Xen-Blanket の利用を検討したが，Xen-Blanket はゲスト VM として PV ゲストしかサポートしていないため採用しなかった．PV ゲストをマイグレーションする場合，VM 間メモリスワップを行う際にゲスト VM のメモリ上にある P2M テーブルを書き換える必要がある．しかし，P2M テーブルは不正な値が書き込まれないように書き込み保護がなされているため，VM 間メモリスワップを実装するのが難しい．HVM ゲストの場合，マイグレーション時にゲスト VM のメモリを書き換える必要がない．そのため，ゲスト VM として HVM ゲストをサポートしている Xen 4.2 を採用した．

Warm-VM Reboot [6] は VM を再起動せずにハイパーバイザだけを再起動するシステムである．ハイパーバイザを再起動する際に VM のメモリイメージをディスクなどの外部記憶ではなくメインメモリ上にそのまま保持することで高速に VM をサスペンドする．また，起動時にはメインメモリ上に残されたメモリイメージを再利用することで高速に VM のレジュームを行うことができる．しかし，このシステムではハイパーバイザおよび管理 VM を再起動している間は VM が停止してしまいダウンタイムが発生するため，提案手法とはソフトウェア若化へのアプローチが異なる．

ReHype [7] も VM を再起動することなくハイパーバイザのみを再起動することができるシステムである．ReHype では，管理 VM の再起動さえも行う必要がないため，Warm-VM Reboot よりも高速にハイパーバイザの再起動を行うことができる．さらに，ハイパーバイザの障害からも高確率で回復することが可能である．ただし，ReHype では，よりソフトウェア・エージングが起こりやすい管理 VM を再起動する際には，仮想化システム全体の再起動を必要とする．

7. まとめ

本稿では，仮想化システムの高速なソフトウェア若化のために，ネストした VM を用いて高速かつ低負荷なマイグレーションを可能にする VMBeam を提案した．VMBeam では，同一ホスト上で二つの仮想化システムを動作させ，仮想化システムのソフトウェア若化時には VM をもう一方の仮想化システムにマイグレーションする．このマイグレーションはVM間メモリコピーや VM 間メモリスワップを用いて高速化する．我々は Xen を用いて VMBeam の実装を行い，VM のマイグレーションを通常よりも高速かつ低負荷で行えることを示した．

現在の実装では，簡単のために，VM間メモリコピーや，VM 間メモリスワップを行うのに必要なゲスト VM のメモリ情報を仮想ネットワークを用いてメモリサーバへ転送している．そのため，仮想ネットワークを用いることによるオーバーヘッドが生じてしまっている．このオーバヘッドは，ハイパーコールを用いてゲスト・ハイパーバイザ経由でホスト・ハイパーバイザに直接情報を送れるようにすることで削減できると考えられる．また，VM 間メモリコピーを用いたライブマイグレーションの実現や VM 間メモリスワップを用いる際のダウンタイムの削減も今後の課題である．

参考文献

[1] Y. Huang, C. Kintala, N. Kolettis, and N. Fulton. Software Rejuvenation: Analysis, Module and Applications. In *Proceedings of the 25th International Symposium on Fault-Tolerant Computing*, pages 381—391, 1995.

[2] Machida, F., Xiang, J., Tadano, K., Maeno, Y.: Combined Server Rejuvenation in a Virtualized Data Center, *Proceedings of ATC*, 2012

[3] 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, *Proceedings of the nineteenth ACP symposium on Operating systems principles*, pp. 164-177（2003）

[4] Williams, D., Hani, J. and Hakim, W.: The Xen-Blanket: Virtualize Once, Run Everywhere, *Proceedings of ACM EuroSys*, Bern, Switzerland, April 2012

[5] Nakajima, J.: Making Nested VIrtualization Real by Using Hardware Virtualization Features, *LinuxCon Japan*, May 2013

[6] Kourai, K. and Chiba, S.: Fast Software Rejuvenation of Virtual Machine Monitors, *IEEE Transactions on Dependable and Secura Computing (TDSC)*, Vol.8, No.6, pp.839-851, November/December 2011

[7] Le, M. and Tamir, Y.: ReHype: Enabling VM Survival Across Hypervisor Failures, *7th ACM International Conference on Virtual Execution Environments*, Newport Beach, California, pp.63-74, March 2011