Vol.2015-OS-135 No.4
Vol.2015-EMB-39 No.4
2015/11/24

大容量メモリを持つ仮想マシンの分割マイグレーション

末竹 将人[1]　木津 巴都希[1]　Surote Wongpaiboon[2]　光来 健一[1]

概要：仮想マシン（VM）をサービスとしてユーザに提供する IaaS 型クラウドの発展に伴い，一台のサーバで多くの VM を稼働させるだけでなく，大容量メモリを持つ VM も提供されるようになってきた．一方で，VM のマイグレーションを行うには移送先のホストに十分な空きメモリ容量が必要となり，大容量メモリを持つ VM はマイグレーションを行うのが困難になるという問題がある．マイグレーションのために大容量のメモリを備えたホストを確保しておくのはコストの面から難しいことが多いためである．本稿では，大容量メモリを持つ VM を複数のホストに分割してマイグレーションすることを可能とするシステム *S-memV* を提案する．S-memV は VM の核となる情報と頻繁にアクセスされるメモリを移送先のメインホストに送り，メインホストに入りきらないメモリはサブホストに送る．VM がサブホストにあるメモリを必要とした時には，メインホストとサブホストの間でメモリをスワップする．S-memV はこのような 1 対 N マイグレーションに加えて，N 対 1 マイグレーションおよび複数ホストにまたがる部分マイグレーションもサポートする．我々は S-memV を KVM に実装し，仮想メモリを用いた従来手法よりもマイグレーション時間を短縮できることを示した．

1. はじめに

近年，クラウドコンピューティングの普及が進んでいる．そのサービス形態の 1 つである IaaS 型クラウドでは，仮想マシン (VM) をサービスとしてユーザに提供し，ユーザが必要な時に必要なだけ利用することができる．IaaS 型クラウドの発展に伴い，1 台のサーバに多くの VM を統合するだけでなく，数十〜数百 GB の大容量メモリを持つ VM も提供されるようになってきた．例えば，ビッグデータの解析には巨大なメモリを持つ VM が必要とされている．できるだけメモリ上にデータを保持することで，ビックデータをより高速に解析することができる．また，メモリ上に大量のデータを保持する高速なインメモリ・データベースを用いることもできる．

一方で，大容量メモリを持つ VM はマイグレーションを行うのが困難になるという問題がある．マイグレーションはホストをメンテナンスする際などに VM を停止させることなく他のホストへ移動させる技術である．VM のマイグレーションを行うには移送先のホストに十分な空きメモリ容量が必要となる．しかし，マイグレーションのために大容量メモリを備えた空きホストを常に確保しておくのは，可能だとしてもコストの大幅な増大を招く．VM のマイグレーションが行えなければ，ホストのメンテナンス中は VM を停止させなければならなくなり，VM 上のサービスが長時間中断されることになる．

本稿では，大容量メモリを持つ VM を複数のホストにマイグレーションすることを可能とするシステム *S-memV* を提案する．S-memV では，マイグレーション時の VM の移送先のホストは 1 台とは限らず，1 つの**メインホスト**と 0 台以上のの**サブホスト**からなる．S-memV は CPU やデバイスの状態のような VM の核となる情報を移送先のメインホストに送る．また，VM のがマイグレーション後に頻繁にアクセスすると考えられるメモリもできる限りメインホストに送る．一方，メインホストに入りきらないメモリはサブホストのいずれかに送る．マイグレーション後はメインホストで VM を動作させ，VM がサブホストにあるメモリを必要とした場合には，メインホストとサブホストの間でメモリをスワップする．また，S-memV では，このような 1 対 N のマイグレーションだけではなく，N 対 1 のマイグレーションや複数ホストの一部だけをマイグレーションする部分マイグレーションも可能とする．

我々は S-memV を KVM に実装し，VM の 1 対 N のマイグレーションを実現した．また，メインホストで動作する VM のメモリをサブホスト上で管理するメモリサーバを開発した．さらに，ホスト OS で Intel EPT を用いてメモリのアクセス頻度を取得する機構も開発した．実験より，S-memV は仮想メモリを用いた従来手法よりもマイグレーション時間を短縮できることがわかった．特に，メモリに

[1] 九州工業大学
Kyushu Institute of Technology
[2] Kasetsart University

負荷をかけている時にマイグレーション時間の増加を抑えられることがわかった．

以下，2 章で大容量メモリを持つ VM をマイグレーションする際の問題点について述べ，3 章で VM の分割マイグレーションを行う S-memV を提案する．4 章で S-memV の実装について説明し，5 章で S-memV 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. 大容量メモリを持つ VM のマイグレーション

VM マイグレーションは稼働している VM を停止させることなく別のホストへ移動させる技術である．マイグレーションを活用することで，VM が提供しているサービスを停止させることなくホストのメンテナンスを行うことができる．マイグレーションを行う際には，移送先ホストに新しい VM を作成し，移送元ホストで動いている VM のメモリの内容をネットワーク経由で移送先ホストの VM のメモリにコピーする．この間，移送元ホスト上で VM のメモリの内容は更新され続けているため，再度，更新されたメモリを転送する．これを繰り返して転送するメモリ量が十分小さくなったら移送元ホストの VM を停止し，VM の CPU などの状態および更新されたメモリを転送してマイグレーションを完了する．

近年，大容量メモリを持つ VM が使用されるようになってきた．例えば，Amazon EC2 では 244 GiB のメモリを持つ VM が提供されている．このような VM はビッグデータの解析やインメモリ・データベースなどのように，大量のデータを高速に扱う場合に用いられている．しかし，このような大容量メモリを持つ VM は，マイグレーション時に移送先を見つけることが困難になるという問題がある．移送先として大きな空きメモリ容量を持ったホストを確保し続けておくことは，コスト面からも難しいことが多い．もし，十分なメモリ容量を持つホストが多数の小さな VM を動かすために使われていた場合，まず，それらの VM をマイグレーションして必要な空きメモリ容量を確保する必要がある．

2.1 仮想メモリを利用したマイグレーション

従来，移送先ホストに十分な空きメモリ容量がない場合は仮想メモリが用いられてきた．仮想メモリは物理メモリに入りきらないメモリをディスク上のスワップ領域に待避することで，物理メモリよりも大きなメモリを扱うことができる技術である．マイグレーションにおいては VM のメモリが順番に転送されるため，転送先ホストの物理メモリに入りきらない VM のメモリはすべてスワップアウトされてしまう．さらに，メモリの再送時に移送先ホストの物理メモリ上にないページはディスクからスワップインされてから上書きが行われる．2 回目以降のメモリの再送では，書き換えられるページ数が少なければスワップはほとんど起こらないが，VM のメモリの大容量化に伴い，マイグレーション中に書き換えられるページ数が増えるとスワップが発生する．マイグレーション完了時には，頻繁に書き換えられるページは物理メモリ上にあるが，読み込みしか行われないページはスワップされている可能性もある．そのため，マイグレーション後にはディスクとの間で繰り返しスワップが起こる．メモリに比べてディスクは非常に遅いため，スワップを繰り返すと性能が著しく低下するという問題がある．

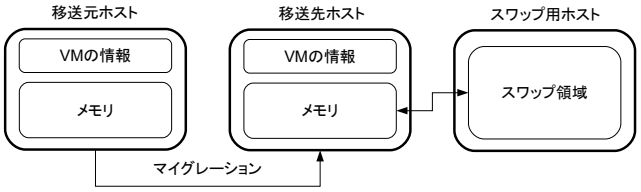

図 1 ネットワーク・スワップを用いたマイグレーション

ネットワーク・スワップ [1] は物理メモリに入りきらないメモリをディスクの代わりに，ネットワーク上の別のホストにスワップする技術である．ネットワーク・スワップを用いることで別のホストのメモリをネットワーク経由で利用することができる．ネットワークが十分高速であれば，ネットワーク・スワップはディスクを用いたスワップよりも高速に行うことができる．しかし，従来のマイグレーションでは VM のすべてのメモリが一旦，移送先ホストに転送されてからスワップ用ホストに再転送されるため，移送先ホストとスワップ用ホスト間のネットワーク帯域を消費してしまう．

2.2 ポストコピー・マイグレーション

上記のプレコピー・マイグレーションとは異なり，ポストコピー・マイグレーション [2] を用いるとスワップの頻度を減らすことができる．ポストコピー・マイグレーションは，VM の実行に必要な核となる情報のみを最初に転送し，すぐに移送先ホストの VM に切り替える．VM のメモリの転送方法には，VM がメモリを必要とした際に移送元から転送するオンデマンド転送と，メモリアクセスがなくても裏で転送を行うバックグラウンド転送がある．オンデマンド転送では，移送先ホストが使用頻度の低いメモリをスワップアウトすることができるため，プレコピー・マイグレーションのような強制的なスワップが起こらない．ただし，オンデマンド転送だけでは VM のメモリアクセスの遅延が大きくなるため，バックグラウンド転送と組み合わせて用いられる．バックグラウンド転送はプレコピー・マ

イグレーションと同様にアクセスの有無に関わらずメモリを転送するため，移送先ホストで頻繁にアクセスされるメモリがスワップアウトされてしまう可能性が高くなる．

3. S-memV

本稿では，大容量メモリを持つ VM を分割して複数のホストにマイグレーションすることを可能とする S-memV を提案する．

3.1 1 対 N マイグレーション

S-memV では，1 つのホストから複数のホストへのマイグレーションが可能である．移送先の複数のホストは図 1 のように 1 台のメインホストと 0 台以上のサブホストからなる．VM を動かす上で必要となる CPU やデバイスの状態などの核となる情報はメインホストに送る．また，VM が頻繁にアクセスすると考えられるメモリはできる限りメインホストに送り，移送先ホストの VM がオーバヘッドなしでアクセスできるようにする．一方，メインホストに入りきらない VM のメモリはサブホストのいずれかに転送する．この際に，空きメモリを考慮して最適なメインホストとサブホスト群を選択できるようにするために，全ホストの空きメモリを管理するサーバを用いる．移送元ホストはマイグレーション前にこのサーバにアクセスすることにより，マイグレーション先のホスト群を決定する．また，マイグレーション時に複数のホストにメモリを並列に転送することでマイグレーションの高速化を図ることができる．

マイグレーション後はメインホスト上で VM を動作させる．アクセスされたメモリがメインホスト上にない場合は，サブホスト上にある要求されたメモリとメインホスト上の使用頻度の低いメモリをスワップする．この挙動はネットワーク・スワップと同様である．しかし，S-memV はマイグレーション中にスワップを発生させない点が異なる．メインホストに入りきらないメモリはメインホスト経由でサブホストにスワップされるのではなく，直接，サブホストに転送される．そのため，マイグレーションで最初に全メモリを送る際も，メモリを再送する際も，無駄なネットワーク転送が行われることはない．

S-memV はプレコピー・マイグレーションだけでなく，ポストコピー・マイグレーションにも適用することができる．オンデマンド転送の場合は，VM が要求したメモリを移送元ホストから移送先のメインホストに転送する．メインホストの物理メモリに空きがない場合は，使用頻度の低いメモリをサブホストにスワップアウトする．バックグラウンド転送の場合は，プレコピー・マイグレーションと同様に，VM が頻繁にアクセスするメモリはメインホストに送り，メインホストに入りきらないメモリはサブホストに転送する．いずれの場合でも，移送先ホストに転送が完了したメモリについては必要に応じてメインホストとサブホストの間でスワップを行う．

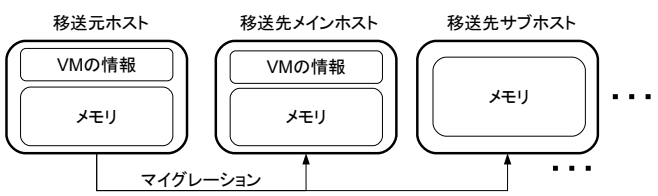

図 2 S-memV における 1 対 N マイグレーション

3.2 N 対 1 マイグレーション

S-memV を用いて複数のホストに分割マイグレーションした VM は，ホストのメンテナンス終了後や十分な空きメモリ容量を持ったホストが用意できた時などに，再び 1 台のホストで動作するようにマイグレーションすることができる．移送元のメインホストとサブホスト上の VM のメモリを並列に転送することにより，高速にマイグレーションを行うことができる．マイグレーション中にサブホストからメインホストにスワップインされたメモリについては，まだ転送されていない場合および書き換えられた場合にだけ移送先ホストに転送する．マイグレーション中にメインホストからサブホストにスワップアウトされたメモリについても同様である．

3.3 部分マイグレーション

S-memV では，VM が動作している複数のホスト全体または一部をメンテナンスできるようにするために，複数のホストにまたがる VM の全体または一部を別のホスト群にマイグレーションすることもできる．メインホスト上にある VM の本体をマイグレーションする際には，移送先のメインホストに VM の核となる情報や頻繁にアクセスするメ

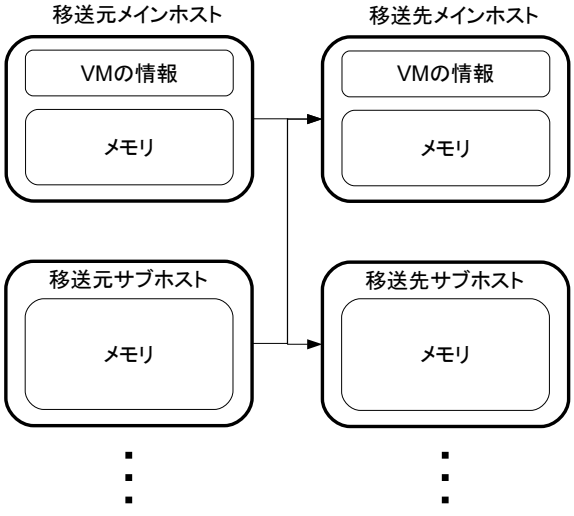

図 3 S-memV における部分マイグレーション

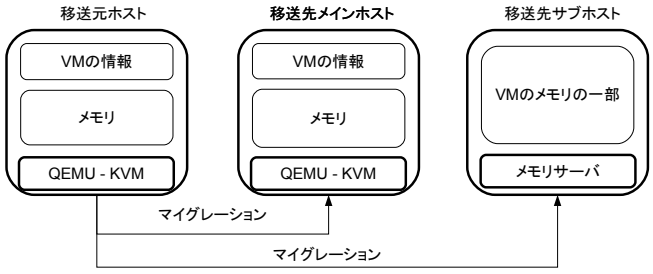

図 4 S-memV のシステム構成

モリを転送し，入りきらないメモリは既存のサブホストまたは新しく確保したサブホストに転送する．サブホスト上にある VM の一部をマイグレーションする際には，移送先のサブホストに VM のメモリの一部を転送する．

4. 実装

S-memV を QEMU-KVM 2.1.2 を用いて実装した．現在の実装では 1 対 N マイグレーションにのみ対応しており，N=2 を仮定している．

4.1 システム構成

S-memV のシステム構成を図 4 に示す．移送元ホストと移送先メインホストでは S-memV を実装した QEMU-KVM が動作し，マイグレーション前後で VM を動かす．移送先サブホストでは VM のメモリの一部を管理するメモリサーバを動作させる．

4.2 QEMU-KVM の拡張

QEMU-KVM はプレコピー・マイグレーションをサポートしている．まず，VM のメモリ全体をページ単位で移送先ホストの QEMU-KVM に転送する．その際には，メモリブロックのオフセットとページの内容を送る．移送先ホストの QEMU-KVM はそのオフセット情報を基に受信したページの内容を VM のメモリに書き込む．メモリ全体を転送した後，転送中に書き換えられたページが再送される．マイグレーションの最適化として，ページ全体が 0 のページについてはその情報だけが送られる．VM のメモリは mmap の無名マッピングを用いて確保されており，ページ全体が 0 の場合にはアクセスされるまで物理メモリの確保が行われない．

VM のメモリを分割してマイグレーションできるようにするために，QEMU-KVM のマイグレーション機構の拡張を行った．S-memV でマイグレーションを開始した時に，移送先のメインホストで動作している QEMU-KVM だけでなく，サブホストで動作しているメモリサーバにもネットワーク接続を行う．現在の実装ではサブホストは 1 台としている．実際のシステムでは，ネットワーク内で動作しているホストの空きメモリ容量や VM のメモリアクセス頻度を考慮して適切に VM のメモリを分割する必要がある．現在は簡単のために，VM の物理メモリアドレスに閾値を設定し，閾値より小さければメインホストにメモリを転送し，大きければサブホストに転送している．最初のメモリ転送時に送り先ホストを決定したらメモリページごとに送り先ホストを記録しておき，メモリ再送時には同じホストに転送する．

メインホストには従来と同じ情報を送るが，サブホストには物理メモリアドレスとページの内容を送信する．サブホストには VM のメモリブロックの情報がないため，メモリブロックのオフセットの代わりに絶対アドレスを送る．サブホストでの処理については 4.3 節で述べる．サブホストに送ったページについては，サブホストの IP アドレスと物理メモリアドレスの情報を移送先メインホストに送り，移送先メインホストは受け取った情報を基数木で管理する．

VM が virtio を利用していると分割マイグレーションが完了しなかったため，VM のコンフィグを変更して virtio を利用しないようにした．virtio は KVM における準仮想化デバイスであり，ディスクやネットワークなどで準仮想化ドライバを使って高速な I/O を可能にしている．また，メモリバルーニングにも用いられている．virtio を利用していると，マイグレーション処理中に QEMU-KVM がメインホストに存在しない VM のメモリにアクセスしているのではないかと思われる．この問題はメインホストとサブホスト間でのメモリのスワップ機構を実装すれば発生しなくなると考えられる．

4.3 メモリサーバ

メモリサーバは VM のメモリの一部を管理するサーバであり，サブホスト上で動作する．メモリサーバは VM のメモリを 4KB のページ単位で管理する．この管理にはメモリの利用効率が良いデータ構造である基数木を用いる．基数木は Linux カーネルにおいてページキャッシュの管理にも用いられている．メモリサーバは基数木を用いて，VM の物理メモリのアドレスをキーとして，対応する VM のメモリページ情報を管理する．

メモリサーバが VM の物理メモリアドレスと VM のメモリページの内容からなるスワップアウト要求を受信した時には，4KB のメモリを確保して送られてきた VM のメモリページの内容をコピーし，基数木に登録する．一方，VM の物理メモリアドレスからなるスワップイン要求を受信した時には，基数木を探索し，VM のメモリページの内容が見つかればそのデータを要求元に送信する．同時にそのデータを基数木から削除する．

4.4 VM のメモリアクセス頻度の取得

S-memV では VM によるアクセス頻度の高いメモリをで

きるだけメインホストに保持しておけるようにするために，拡張ページテーブル（EPT）をたどってアクセスビットが立っているページを取得する．そのために，QEMU-KVM は定期的に Linux カーネル内の KVM に対して ioctl システムコールを発行する．その際に VM のメモリサイズに対応するビットマップを確保し，ioctl の引数として KVM に渡す．KVM は VM のすべてのメモリページについて EPT をたどり，ページテーブルエントリ（PTE）を取得する．PTE のアクセスビットは対応するページにアクセスした時に 1 にセットされるため，アクセスビットの値を渡されたビットマップに記録する．その後，次の期間のアクセスを記録できるようにするために，アクセスビットをクリアして 0 にセットする．

QEMU-KVM は VM のメモリアクセス状況が記録されたビットマップを何回分か保持しておき，マイグレーション時やスワップ時にアクセス頻度の高いページや低いページを探すのに用いる．この処理については現在，未実装である．

5. 実験

S-memV の有効性を示すために，いくつかの実験を行った．実験には Intel Xeon E3-1270v2 3.5GHz （8 コア）の CPU，16GB のメモリを搭載したマシンを移送元ホストとし，Intel Xeon E5640 2.67GHz（4 コア）の CPU，2GB のメモリ，320 GB の SATA 2 HDD を搭載したマシン 2 台をそれぞれ移送先のメインホストおよびサブホストとして用いた．移送先ホストでは 4GB のスワップ領域を設定した．これらのマシンはギガビットイーサネットで接続した．移送先，移送元の OS には Linux 3.13.0 を用い，仮想化ソフトウェアには QEMU-KVM 2.1.2 を使用した．VM には仮想 CPU を 1 個，メモリを 2GB 割り当てた．

5.1 物理メモリとスワップ領域の使用量

S-memV を用いることでメモリの利用状況がどのように変化するかを調べるために，マイグレーションを行う前後で移送先メインホストの物理メモリとスワップ領域の使用量を測定した．まず，仮想メモリを利用する従来システムを用いて VM をマイグレーションした．物理メモリとスワップ領域の使用量の変化を図 5 に示す．移送先ホストでは，マイグレーション前から約 1GB の物理メモリが使用されていたため，マイグレーションされた VM は残りの約 1 GB の物理メモリを使用した．VM のメモリの残り 1 GB はスワップ領域に格納された．

次に，S-memV を用いて VM のマイグレーションを行った．物理メモリとスワップ領域の使用量の変化を図 6 に示す．この実験では VM のメモリの 512 MB だけをメインホストに転送するように閾値を設定したため，物理メモリの使用量が約 512 MB 増加した．VM の残りのメモリ 1.5 GB はサブホストに送られ，メインホストのスワップ領域は使用されなかった．

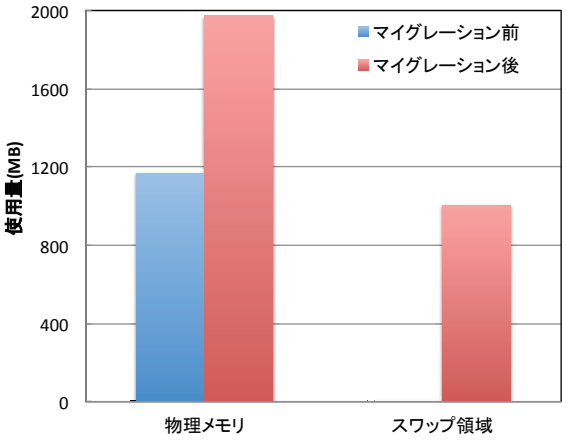

図 5 物理メモリとスワップ領域の使用量（従来）

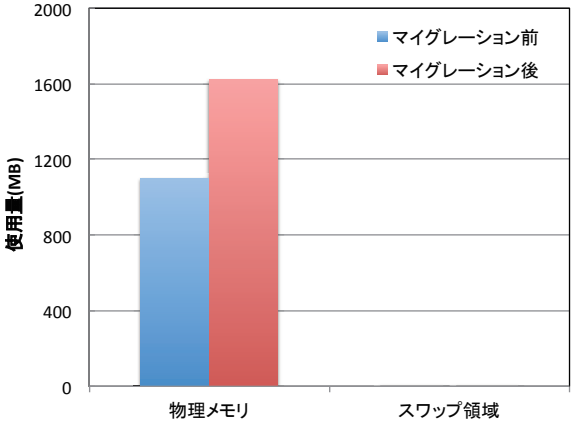

図 6 物理メモリとスワップ領域の使用量（S–memV）

5.2 マイグレーション時間

S-memV のマイグレーション性能を調べるために，マイグレーションにかかる時間を測定した．比較対象として，仮想メモリを利用する従来システムと，移送先メインホストに十分なメモリがある場合の従来システムを用いた．後者の実験は，移送先ホストのメモリを 4GB に増設して行った．この実験では，S-memV がサブホストに VM のメモリを送る閾値を 1GB とした．

まず，VM 内でアプリケーションを動作させずにマイグレーションを行った．マイグレーション時間を図 7 に示す．メモリが十分にある場合と比べて，メモリ不足によりスワップ領域が使われた場合には性能が 45 %低下した．それに対して，S-memV での性能低下は 28 %であった．この結果より，S-memV はスワップ領域を使用する場合よりもマイグレーション性能の低下を抑えることができることがわかった．

次に，VM 内で インメモリ・データベースの memcached [3] を動作させ，memaslap ベンチマーク [4] を用いてメモリ

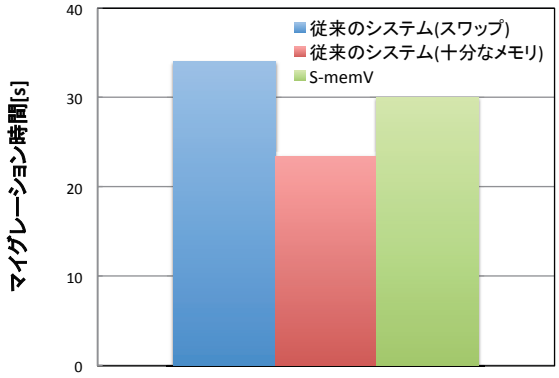

図 7 マイグレーション時間 (通常時)

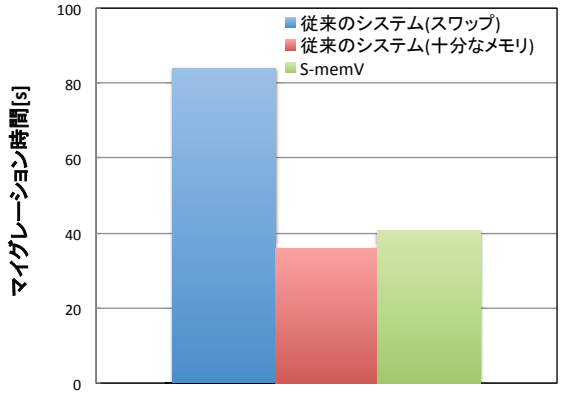

図 8 マイグレーション時間 (メモリ負荷時)

に継続的に負荷をかけた状態でマイグレーションを行った．memaslap の set と get の比率は 0.6 対 0.4 に設定した．実験結果を図 8 に示す．メモリが十分にある場合でも，上の実験と比べてマイグレーション時間が 1.7 倍に増加していることがわかる．これは VM によるメモリ書き換えによって大量のメモリが再送されたためである．メモリが十分にある場合と比べて，スワップ領域が使われた場合にはマイグレーション時間が 3.7 倍になった．メモリの再送によりスワップが頻発したためである．それに対して，S-memV での性能低下は 11 %であった．この結果より，S-memV ではメモリに負荷をかけてもマイグレーション時間の増加を抑えられることがわかった．

6. 関連研究

ポストコピー・マイグレーション [2] は，VM を実行する上で最低限，必要な情報だけを先に転送して移送先ホストの VM に切り替える手法である．移送先ホストに存在しないメモリは，VM が要求した時に移送元ホストから転送する．このオンデマンド転送の技術は，S-memV においてメインホストとサブホストの間でメモリをスワップする機構を実装するのに利用できると考えられる．

MemX [5] は，VM が複数のホストのメモリを利用することを可能にする．MemX-VM モードでは VM 内のゲスト OS がブロックデバイスを提供し，デバイス経由で他のホストのメモリへのアクセスを可能とする．MemX-DD モードでは Xen の Dom0 でブロックデバイスを提供する．一方，MemX-VMM モードでは，VM の拡張メモリを通して透過的に他のホストのメモリへのアクセスを可能とする．VM がそのホストに存在しない物理メモリ領域にアクセスすると，他のホストにあるメモリを取得する．このモードは S-memV の分割マイグレーション後の動作と同じである．しかし，MemX では VM のマイグレーション時に他のホストへメモリを転送しないため，分割マイグレーションを行うことはできない．

Scatter-Gather マイグレーション [6] は，ポストコピー・マイグレーションを行う際に，移送元ホストと移送先ホストの間でに複数の中間ホストを用いる．VM のメモリをできるだけ速く中間ホストに転送してしまうことにより，移送元ホストで VM が停止するまでの時間を短縮することができる．移送先ホストでは，ポストコピー・マイグレーションのオンデマンド転送やバックグラウンド転送を用いて中間ホストから VM のメモリを取得する．複数のホストに VM のメモリを転送する点では S-memV に似ているが，Scatter-Gather マイグレーションでは最終的に VM のすべてのメモリが移送先ホストに送られる．S-memV では移送先ホストに VM のメモリが入りきらないことを想定している点が異なる．

Virtual Multiprocessor [7] や vNUMA [8] では，複数のホストの CPU やメモリを用いて一つの VM を動かすことを可能とする．これにより，1 台のホストではリソースが不足していても複数のホストを用いることで巨大な VM を動かすことができる．分散共有メモリを用いることでメモリが存在するホストを意識することなく他のホストのメモリにアクセスができる．S-memV とは複数ホストのリソースを使用する点で類似しているが．これらのシステムでは複数のホストで 1 台の VM を動かすのに対し，S-memV では VM を動かすホストは 1 台で，残りのホストはメモリを提供するだけである．また，これらのシステムは VM のマイグレーションには対応していない．

7. まとめ

本稿では，大容量メモリを持つ VM を複数のホストに分割してマイグレーションすることを可能とするシステム S-memV を提案した．S-memV は移送先のメインホストに入りきらない VM のメモリをサブホストに転送する．マイグレーション後，VM はメインホストで動作し，必要に応じてメインホストとサブホストの間で VM のメモリをスワップする．S-memV を KVM に実装し，サブホストでメモリサーバを動作させて，1 対 N マイグレーションを実現した．

今後の課題は，VM をマイグレーションした後，メインホストとサブホストの間でメモリをスワップできるようにすることである．VM のメモリアクセス頻度の情報を利用してメインホストに転送するメモリを選択できるようにし，マイグレーション後のスワップを減らす必要もある．また，複数のホストで動作する VM をマイグレーションできるようにすることも予定している．

参考文献

[1] T. Newhall, S. Finney, K. Ganchev, and M. Spiegel. Nswap: A Network Swapping Module for Linux Clusters. In Proceedings of International European Conference on Parallel and Distributed Computing (2003).
[2] M.R. Hines, and K. Gopalan. Post-Copy Based Live Virtual Machine Migration Using Adaptive Pre-Paging and Dynamic Self-Ballooning. In Proceedings of International Conference on Virtual Execution Environments (2009).
[3] memcached – A Distributed Memory Object Caching System, http://memcached.org/.
[4] memaslap – Load Testing and Benchmarking a Server, http://docs.libmemcached.org/bin/memaslap.html.
[5] U. Deshpande, B. Wang, S. Haque, M. Hined, and K. Gopalan. MemX: Virtualization of Cluster-Wide Memory. In Proceedings of International Conference on Parallel Processing (2010).
[6] U. Deshpande, Y. You, D. Chan, N. Bila, and K. Gopalan. Fast Server Deprovisioning through Scatter-Gather Live Migration of Virtual Machines. 7th IEEE International Conference on Cloud Computing (2014).
[7] 金田憲二, 大山恵弘, 米澤明憲. 単一システムイメージを提供するための仮想マシンモニタ. 情報処理学会 ACS 論文誌, Vol.47, No.SIG 3, pp.27-39 (2006).
[8] M. Chapman and G. Heiser. vNUMA: A Virtual Shared-Memory-Multi Processor. In Proceedings of Conference USENIX Annual Technical Conference (2009).