Vol.2023-OS-160 No.2
2023/8/3

移送先を柔軟に変更可能なVMマイグレーション

緒方 彬人[1]　光来 健一[1]

概要：近年，クラウドでは大容量メモリを持つ仮想マシン (VM) が提供されている．VM はマイグレーションによって別のホストに移動させることができるが，大容量メモリを持つ VM ではより長い時間が必要になる．そのため，マイグレーション中に移送先ホストの負荷が高くなるなど，マイグレーション開始時とは状況が変わる可能性がある．そのような場合には，より負荷の低いホストに移送先を変更するなどの対処を行うことが望ましいが，マイグレーションを一旦，キャンセルして再度，すべての VM の状態を転送し直すと，マイグレーションが完了するまでの間に再び状況が変化する可能性が高くなる．本稿では，VM マイグレーションをキャンセルすることなく，移送先ホストを柔軟に変更することを可能にするシステム DCmigrate を提案する．DCmigrate は移送先ホストを変更した後，移送元ホストからはまだ転送していない VM の状態だけを転送する．一方，古い移送先ホストも受信済みの VM の状態を新しい移送先に転送する．このように VM の状態を 2 つのホストから並列に転送することで，移送先変更後のマイグレーションを高速化し，再度，状況が変化する可能性を低くする．DCmigrate を libvirt [5] と QEMU [2] に実装し，移送先変更後のマイグレーション時間を調べた．

1. はじめに

仮想マシン (VM) は IaaS 型クラウドだけでなく，PaaS 型クラウドや SaaS 型クラウドにおいても広く用いられている．近年，クラウドで利用される VM はメモリの大容量化が進んでいる．例えば，Amazon EC2 では最大 24TB のメモリを持つ VM が提供されている [1]．VM はマイグレーションと呼ばれる技術によって別のホストに移動させることができ，ホストのメンテナンスや負荷分散，VM の集約などのために用いられている．一方，マイグレーションには VM のメモリサイズに比例した時間がかかるため，大容量メモリを持つ VM ではより長い時間が必要になる．

そのため，マイグレーション中に移送先ホストの負荷が高くなったりネットワークが混雑したりするなど，マイグレーション開始時とは状況が変わる可能性がある．そのような場合には，より負荷の低いホストや混雑していないネットワークで接続されたホストに移送先を変更するなどの対処を行うことが望ましい．しかし，VM の移送先ホストを変更するにはマイグレーションを一旦，キャンセルする必要があり，それまでに転送された VM の状態はすべて破棄される．そして，再度，すべての VM の状態を転送し直す必要があるため，マイグレーションが完了するまでの間に再び状況が変化する可能性が高くなる．

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

そこで本稿では，VM マイグレーションをキャンセルすることなく，移送先ホストを柔軟に変更することを可能にするシステム DCmigrate を提案する．DCmigrate は移送元ホストにおいて移送先を新しいホストに切り替え，まだ転送していない VM の状態だけを新しい移送先ホストに転送する．一方，変更前の移送先ホストは受信済みの VM の状態を破棄せず，新しい移送先ホストに転送する．このようにして，移送元ホストと古い移送先ホストの双方から新しい移送先ホストに並列に VM の状態を転送し，新しい移送先ホストでは VM の状態を並列に受信する．その結果，移送先変更後のマイグレーション時間を短縮することができる．

我々は DCmigrate を libvirt 8.9.0 と QEMU 7.2.0 を拡張することで実装した．移送ツールの virsh は各ホストで動作する管理サーバの libvirtd に対してリモート手続き呼び出し (RPC) を実行し，実行中のマイグレーションを部分的にキャンセルして新しい移送先ホストへのマイグレーションを開始する．libvirtd は VM を実行している QEMU にコマンドを送信して移送先ホストを切り替え，新しい移送先ホストに VM の状態を並列に転送する．DCmigrate を用いて移送先ホストの変更とその後のマイグレーションにかかる時間を測定し，マイグレーションをキャンセルして新しい移送先ホストにマイグレーションし直した場合にかかる時間との比較を行った．

2 章では VM マイグレーション中に状況が変化した際の

問題点について述べる．3 章では実行中のマイグレーションの移送先ホストを変更可能にする DCmigrate を提案する．4 章では DCmigrate の実装について述べる．5 章では DCmigrate を用いて移送先ホストを変更した実験について述べる．6 章で関連研究に触れ，7 章で本論文をまとめる．

2. VM マイグレーション中の状況の変化

VM マイグレーションは移送元ホストから移送先ホストに VM の状態を転送することによって行われる．マイグレーションを開始すると，まず，移送先ホストにおいて空の VM を作成し，VM の状態が送られてくるのを待つ．移送元ホストがメモリデータを移送先ホストに転送すると，移送先ホストではそのデータを作成した VM のメモリに書き込む．マイグレーション中に移送元ホストの VM によって更新されたメモリについては，そのデータを移送先ホストに再送する．転送すべきメモリデータが閾値を下回ったら移送元の VM を停止させ，残りのメモリデータおよび仮想デバイスの状態を転送する．移送先ホストは受信したすべての VM の状態を VM に反映させたら VM を再開する．

マイグレーション中に転送するデータの中で VM のメモリが最も大きな割合を占めるため，マイグレーションには VM のメモリサイズに比例した時間がかかる．近年，VM のメモリの大容量化が進んでおり，Amazon EC2 は最大 24TB のメモリを持つ VM を提供している．このような大容量メモリを持つ VM をマイグレーションするには非常に長い時間が必要となる．例えば，VM のメモリサイズが 352GB の場合，10 ギガビットイーサネットを用いても 7 分以上かかることが報告されている [7]．ホストの負荷が高い場合やネットワークが混雑している場合にはさらに長い時間がかかる．

このように，大容量メモリを持つ VM のマイグレーションには時間がかかるため，マイグレーションの実行中にマイグレーション開始時とは状況が変わる可能性がある．例えば，移送先として負荷の低いホストを選ぶのが一般的であるが，時間が経つとそのホストの負荷が高くなる場合がある．また，移送元ホストからのネットワークが混雑していない移送先ホストを選んだとしても，時間が経つとそのネットワークが混雑する場合がある．マイグレーションにかかる時間が長くなるほど，このような状況の変化が起きる可能性が高くなる．

マイグレーション中に状況が変化した場合には，図 1 のように移送先ホストを変更することが望ましい場合がある．例えば，移送先としてより負荷の低いホストがある場合には，移送先ホストを変更することでマイグレーションを高速化し，マイグレーション後の VM の性能を向上させることができる．また，混雑していないネットワークでつながれたホストがある場合，移送先をそのホストに変更することでマイグレーションを早く終わらせることができ，

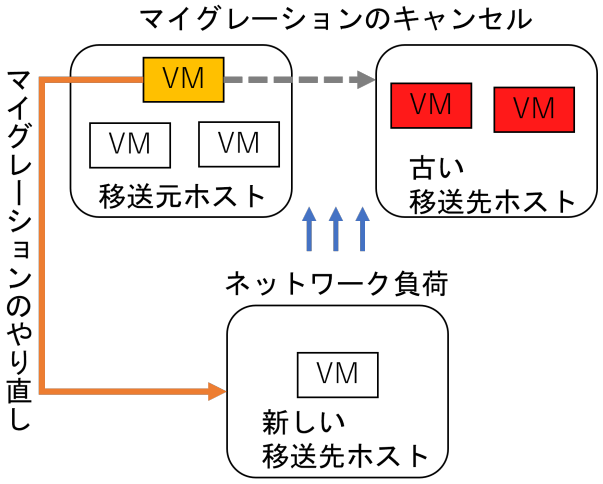

図 1 状況変化による移送先ホストの変更

マイグレーション後の VM のネットワーク性能も向上させることができる．

しかし，VM の移送先ホストを変更するには実行中のマイグレーションを一旦，キャンセルする必要がある．マイグレーションをキャンセルするとそれまでに転送された VM の状態は移送先ホストにおいてすべて破棄される．その後で，新しい移送先ホストに対して VM のマイグレーションを一からやり直す必要がある．再度，VM のすべての状態を転送し直す必要があるため，VM のメモリサイズに応じたマイグレーション時間がかかる．そのため，新しいマイグレーションが完了するまでの間に，再び状況が変化する可能性が高くなる．

3. DCmigrate

本稿では，VM マイグレーションをキャンセルすることなく柔軟に移送先ホストを変更可能にするシステム DCmigrate を提案する．移送先ホストを変更する際に，DCmigrate は図 2 のように移送元ホストにおいて移送先を新しいホストを切り替え，古い移送先ホストにまだ転送していない VM の状態と再送が必要な VM の状態だけを新しい移送先ホストに転送する．一方，変更前の古い移送先ホストは移送元ホストから転送済みの VM の状態を破棄せず，新しい移送先ホストに転送する．これにより，移送元ホストと古い移送先ホストの双方から新しい移送先ホストへ並列に VM の状態を転送する．

このように VM の状態を並列に転送することで，マイグレーションを一からやり直す場合と比べて，移送先ホストを変更した後のマイグレーション時間を短縮することができる．そのため，移送先変更後のマイグレーション中に再度，状況が変化して，再び移送先を変更しなければならなくなる可能性を低くすることができる．また，移送元ホストをメンテナンスする際には作業により早く取り掛かることができるようになり，負荷分散を行う際には性能をより

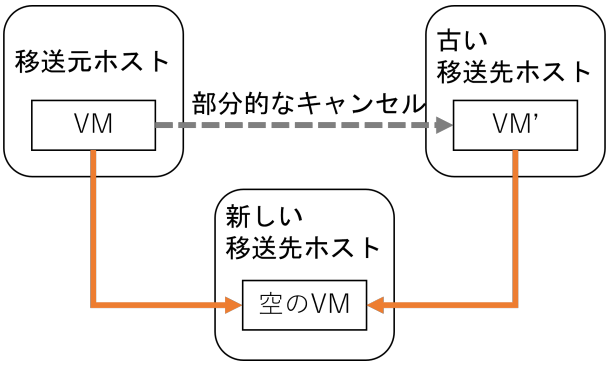

図 2 DCmigrate によるマイグレーション

早く改善することができるようになる．それに加えて，移送元ホストから転送しなければならない VM の状態が減るため，移送元ホストの負荷を減らすとともにネットワーク負荷を分散させることもできる．

DCmigrate は移送先ホストを変更する際に，移送元ホストにおいて実行中のマイグレーションを部分的にキャンセルする．この部分的なキャンセルはマイグレーションを中断するが，VM の状態の転送は継続する．そして，元のマイグレーションの状態を引き継いで，新しい移送先ホストへのマイグレーションを開始する．移送元ホストは新しい移送先ホストに接続し，VM の状態の転送先を切り替える．そして，元のマイグレーションの続きから VM の状態を転送する．その際に，新しい移送先ホストが必要とする，古い移送先ホストに転送済みの VM のサイズなど情報については再送する．

同時に，DCmigrate は古い移送先ホスト対しても移送元ホストと同様に，実行中のマイグレーションを部分的にキャンセルさせる．この部分的なキャンセルはマイグレーションの移送先としての処理を終了させ，VM の状態の受信を行わないようにする．一方で，移送元ホストから受信済みの VM の状態は破棄せずに保持する．そして，新しい移送先ホストに接続し，保持している VM の状態を転送する．

さらに，DCmigrate は新しい移送先ホストにおいて移送元ホストと古い移送先ホストからの VM の状態の並列受信を開始する．そして，双方から受信した VM の状態を統合して完全な VM の状態を構築する．その際に，移送元ホストで VM によって更新されたメモリデータが再送されると，2 つのホストから重複してメモリデータを受信する可能性がある．その場合には移送元ホストから転送されたメモリデータの方が常に最新であるため，そのメモリデータを利用する．すべての VM の状態を受信するまで待って VM を再開することにより，マイグレーションを完了する．

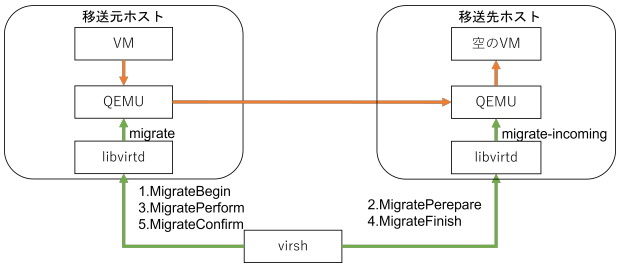

図 3 従来のマイグレーションの流れ

4. 実装

我々は DCmigrate を libvirt と QEMU に実装した．VM を管理するために，各ホストで libvirtd と呼ばれる管理サーバが動作する．libvirtd に対して，virsh などの移送ツールが libvirt ライブラリを用いてリモート手続き呼び出し (RPC) を実行することで，VM のマイグレーションを実行する．また，KVM は QEMU を用いて VM を動作させ，QEMU がマイグレーション処理を行う．

4.1 従来の VM マイグレーションの流れ

VM と移送先ホストを指定して virsh の migrate コマンドを実行することで，VM をマイグレーションする．virsh は指定された移送先ホストに ssh を用いて接続し，libvirtd が作成した Unix ドメインソケット経由で libvirtd との通信を行う．移送先ホストとの接続後，virsh は RPC を用いて図 3 のように移送元ホストと移送先ホストの libvirtd の計 5 つの関数を実行する．

まず，virsh は移送元ホストの libvirtd の MigrateBegin 関数を実行する．この関数は virsh から渡された移送先ホストやマイグレーションのオプションの情報を受け取り，移送元におけるマイグレーションジョブを開始する．そして，マイグレーションを行う VM のドメイン定義 XML を生成して返す．

次に，virsh は移送先ホストの libvirtd の MigratePrepare 関数を実行する．この関数では移送先におけるマイグレーションジョブを開始し，移送元ホストの libvirtd で生成された VM のドメイン定義 XML を用いて移送先ホストに新しい空の VM を作成する．そして，その VM を実行している QEMU に対して QEMU マシンプロトコル (QMP) を用いて migrate-incoming コマンドを送信する．このコマンドを受信した QEMU はマイグレーション用のソケットを作成し，移送元ホストからの接続を待つ．

その後，virsh は移送元ホストの libvirtd の MigratePerform 関数を実行する．この関数ではマイグレーション対象の VM を実行している QEMU に対して QMP を用いて migrate コマンドを送信する．このコマンドを受け取った QEMU は移送先ホストの QEMU に接続し，VM の状態

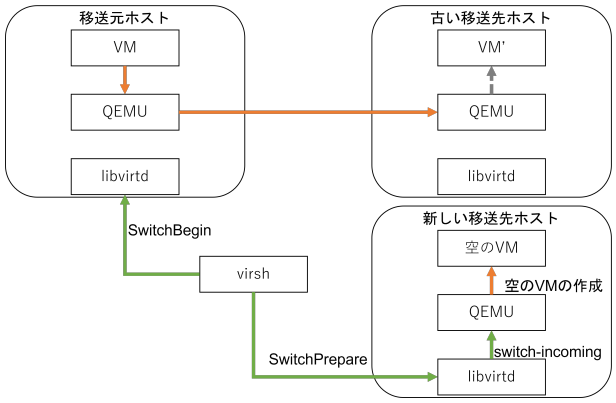

図 4 移送先変更の準備

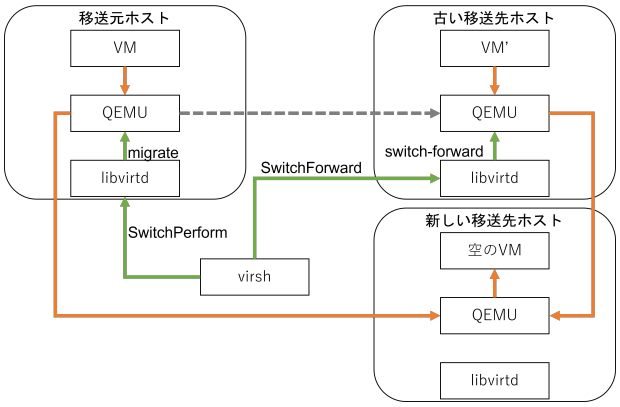

図 5 新しい移送先への状態転送

の転送を開始する．MigratePerform 関数は QEMU が VM の状態を送信し終わるまで待つ．

移送元ホストから VM の状態の転送が完了すると，virsh は移送先ホストの libvirtd の MigrateFinish 関数を実行する．この関数は VM の状態の受信を完了するまで待ち，すべての状態を受信すると移送先におけるマイグレーションジョブを終了して VM を再開する．

最後に，virsh は移送元ホストの libvirtd の MigrateConfirm 関数を実行する．この関数は状態の転送が完了した VM を終了させ，移送元におけるマイグレーションジョブを終了する．

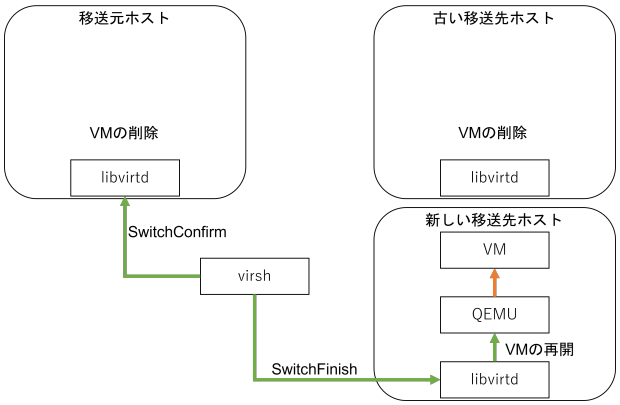

図 6 マイグレーションの終了処理

4.2 移送先ホストの変更の流れ

DCmigrate は VM と新しい移送先ホストを指定して virsh の switch コマンドを実行することで，実行中のマイグレーションの移送先ホストを変更する．virsh はマイグレーション開始時と同様に，指定された新しい移送先ホストに ssh を用いて接続し，Unix ドメインソケット経由で libvirtd との通信を行う．その後，virsh は RPC を用いて移送元ホスト，古い移送先ホスト，新しい移送先ホストの libvirtd の計 6 つの関数を実行する．これらの関数は従来の VM マイグレーションで用いられている関数に似た 5 つの関数と，新しく追加された 1 つの関数からなる．

virsh はまず，図 4 のように移送元ホストの libvirtd の SwitchBegin 関数を実行し，移送先ホストを切り替える．次に，新しい移送先ホストの libvirtd の SwitchPrepare 関数を実行し，VM の状態を受信できるようにする．そして，図 5 のように古い移送先ホストの libvirtd の SwitchForward 関数を実行し，受信済みの VM の状態を新しい移送先ホストに転送する．同時に，移送元ホストの libvirtd の SwitchPerform 関数を実行して，VM の残りの状態を新しい移送先ホストに転送する．この転送が完了すると，図 6 のように新しい移送先ホストの libvirtd の SwitchFinish 関数を実行して VM の状態の受信完了を待ち，VM を再開する．最後に，移送元ホストの libvirtd の SwitchConfirm 関数を実行して，元の VM を終了させる．

4.3 移送元での切り替え処理

移送元ホストの libvirtd で実行される SwitchBegin 関数はまず，マイグレーションジョブを実行中のスレッドにキャンセル通知を送信する．このマイグレーションスレッドは MigratePerform 関数で VM の状態の転送完了を待っており，キャンセル通知を受け取ると待機を終了する．通常，マイグレーションをキャンセルする際には QEMU に対して QMP で migrate_cancel コマンドを送信し，QEMU での VM の状態転送を終了する．しかし，SwitchBegin 関数ではこのコマンドを送信しないようにすることで，QEMU による VM の状態転送を継続する．元のマイグレーションスレッドはマイグレーションジョブを終了させると，新しいマイグレーションスレッドにキャンセル完了を通知する．通知を受けたマイグレーションスレッドは新しいマイグレーションジョブを開始し，MigrateBegin 関数と同様に VM のドメイン定義 XML を生成して返す．

SwitchPerform 関数はまず，マイグレーションジョブのキャンセルにより正常な状態ではなくなったマイグレーションのステータスを更新する．ステータスは QEMU に対して QMP で query-migrate コマンドを送信することで取得する．その後，新しい移送先ホストの QEMU に接続

し，そのソケットのファイル記述子を渡すために移送元ホストの QEMU に対して QMP で getfd コマンドを送信する．次に，QEMU に対して QMP で switch コマンドを送信し，MigratePerform 関数と同様に，新しい移送先ホストに VM の状態がすべて転送されるのを待つ．

switch コマンドを受け取った QEMU は getfd コマンドによって渡されたファイル記述子を用いて I/O チャネルを作成し，新しい移送先ホストへの VM の状態の転送に用いる QEMUFile を取得する．そして，マイグレーションの情報が格納されている MigrateState とメモリの情報が格納されている RAMState の中の QEMUFile を古い移送先ホストへの転送に使われていたものから切り替える．その後，VM のサイズなどのメタデータを新しい移送先ホストに一括で送信する．このメタデータは元のマイグレーション時に送信したものを保存しておくことにより，送信にかかる時間を削減する．そして，VM の状態の転送を新しい移送先ホストに対して再開する．

SwitchConfirm 関数は MigrateConfirm 関数と同様に，VM を終了させ，移送元におけるマイグレーションジョブを終了する．

4.4 古い移送先での切り替え処理

古い移送先ホストの libvirtd で実行される SwitchForward 関数はまず，実行中のマイグレーションジョブをキャンセルし，QEMU に対して QMP で switch-forward コマンドを送信する．このコマンドを受け取った QEMU は VM の状態の受信処理を終了し，移送元ホストから受信済みの VM のメモリデータを新しい移送先ホストに転送する．

QEMU は受信済みのメモリデータを新しい移送先に転送できるようにするために，移送先ホストの変更前に受信したメモリページを受信ビットマップに記録しておく．受信ビットマップは移送元ホストからメモリページを受信した時に対応するビットをセットし，新しい移送先ホストに転送を行った時にビットをクリアする．古い移送先でのこれらの処理は現在のところ，未実装である．

4.5 新しい移送先での処理

新しい移送先ホストの libvirtd の SwitchPrepare 関数はまず，移送先におけるマイグレーションジョブを開始する．そして，SwitchBegin 関数の実行によって移送元ホストの libvirtd で生成されたドメイン定義 XML を用いて新しい空の VM を作成する．その後，その VM を実行している QEMU に VM の状態を並列に受信するための switch-incoming コマンドを送信する．このコマンドを受け取った QEMU は VM の状態を受信するためのスレッドを 2 つ作成する．これらのスレッドはそれぞれ，移送元ホストからの受信と古い移送先からの受信に用いられる．

新しい移送先の QEMU も古い移送先ホストと同様の受信ビットマップを作成する．移送元ホストまたは古い移送先ホストから VM のメモリデータを受信した時に，対応するビットをセットする．この受信ビットマップを用いることにより，ビットがセットされているメモリページのデータを受信した場合，移送元ホストからの場合は VM のメモリに書き込み，古い移送先ホストからの場合は破棄する．これにより，常に最新のメモリデータのみを VM に反映させることができる．

SwitchFinish 関数は移送元ホストと古い移送先ホストから VM のすべての状態を受信するまで待つ．VM の状態の受信が完了すると，移送先におけるマイグレーションジョブを終了して VM を再開する．新しい移送先ホストでの古い移送先ホストからの受信処理は現在のところ，未実装である．

5. 実験

DCmigrate を用いて VM のマイグレーション中に移送先ホストの変更を行えることを確認する実験を行った．実験には表 1 の 3 台のマシンを用いた．マイグレーションを行う VM には 4 つの仮想 CPU と 4GB のメモリを割り当てた．また，DCmigrate による移送先変更およびその後のマイグレーションにかかる時間の測定を行った．比較として，マイグレーションをキャンセルして新しい移送先ホストにマイグレーションし直す従来手法についても調べた．

5.1 移送先変更の確認

マイグレーションの実行中に DCmigrate を用いて移送先ホストを変更し，VM が新しい移送先ホストにマイグレーションされることを確認する実験を行った．この実験では，移送元ホストが VM の状態の転送を開始する前に移送先ホストの変更を行った．これは，移送先ホストの変更後に古い移送先ホストから新しい移送先ホストへの VM の状態の転送がまだ実装できていないためである．このタイミングで移送先ホストを変更できるようにするために，移送元ホストの QEMU が VM の状態の転送を開始する前で待つようにし，移送先ホストを切り替えた後で再開するようにした．

実験の結果，指定した移送先ホストに VM がマイグレーションされ，正常に動作していることが確認できた．移送先ホストの変更時に，virsh の migrate コマンドで実行していた元のマイグレーションは図 7 のように終了した．そして，新しい移送先ホストへのマイグレーションの完了時に図 8 のように virsh の switch コマンドの実行が完了した．

5.2 移送先変更の性能

マイグレーションの実行中に DCmigrate を用いて移送先ホストを変更し，新しい移送先ホストへのマイグレーションを完了するまでにかかる時間を測定した．比較とし

表 1　実験に使用したホスト

項目	移送元ホスト	古い移送先ホスト	新しい移送先ホスト
CPU	Intel Core i7-10700	Intel Xeon W-2245	Intel Core i7-10700
メモリ	128 GB	64 GB	128 GB
OS	Linux 5.15	Linux 5.15	Linux 5.15
QEMU	7.2.0		
libvirt	8.9.0		

```
ogata2@ogata2:~/libvirt-8.9.0$ ./build/tools/virsh migrate test qemu+ssh://ogata3@ogata3/system
error: operation aborted: job 'migration out' canceled by client
```

図 7　移送先変更による migrate コマンドの終了

```
ogata2@ogata2:~/libvirt-8.9.0$ ./build/tools/virsh DCmigrate test qemu+ssh://ogata1@ogata1/system
```

図 8　switch コマンドによる移送先ホストの変更

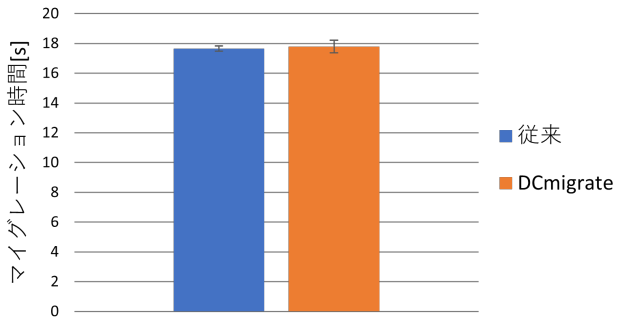

図 9　移送先変更後のマイグレーション時間

て，マイグレーションを一旦，キャンセルして新しい移送先ホストへのマイグレーションを一から実行した場合にかかる時間についても調べた．マイグレーションのキャンセルは virsh の domjobabort コマンドを用いて行い，シェルスクリプトでマイグレーションのキャンセルと再実行を連続して行った．

それぞれの場合で 10 回ずつ測定を行った時の実験結果を図 9 に示す．DCmigrate は元のマイグレーションを完全にはキャンセルせずに新しいマイグレーションに引き継ぐため，マイグレーションをキャンセルする場合と比べて高速化が期待できる．しかし，DCmigrate による高速化は確認できなかった．古い移送先ホストから並列に VM の状態を転送できるようになれば，移送先ホストの変更後のマイグレーションが高速化できると考えられる．

6. 関連研究

Acinonyx [4] は VM マイグレーション中にネットワーク経路を切り替えることでマイグレーション時間を短縮することができる．そのために，複数のネットワーク経路を持つデータセンタにおいて，ホスト間で発生する大量のデータフローを Software Defined Networking (SDN) を用いて動的にスケジューリングする．しかし，移送先ホストを変更することはできないため，移送先ホストの負荷が高い場合や混雑したネットワークを経由しないと移送先ホストに到達できない場合には改善が見込めない．DCmigrate と併用することにより，移送先ホストの変更に加えてネットワーク経路の明示的な変更も行えるとより効果的である．

VM マイグレーションの途中で状況が変わった時，通常のマイグレーションから分割マイグレーション [6] に切り替えることで，残りの VM の状態を転送するホストを変更することができる．分割マイグレーションは VM のメモリを分割して複数の移送先ホストに転送する手法である．これにより，負荷の低いホストに VM の状態を転送することが可能になる．しかし，分割マイグレーション後はホスト間でリモートページングを行ってメモリデータを転送する必要があるため，VM の性能が低下する．また，最初から分割マイグレーションを行う場合とは異なり，アクセスされることが予測されるメモリを VM が実行されるメインホストに転送することができないため，さらに VM の性能が低下する可能性が高い．

そこで，これら 2 つの移送先ホストから 1 つのホストに VM の状態を転送する統合マイグレーション [3] を行うことも考えられる．これにより，リモートページングを行わずに VM を実行できるようになり，VM の性能を回復させることができる．DCmigrate は分割マイグレーションへの切り替えと統合マイグレーションを同時に行う手法と考えることができる．ただし，DCmigrate ではマイグレーションが完了するまで VM は移送元ホストで動作し続けるため，リモートページングにより性能が低下することはない．

プレコピーマイグレーションをキャンセルした時には移送元ホストの VM が実行を継続することができるため，別の移送先ホストを選択してマイグレーションをやり直すことができる．一方，ポストコピーマイグレーションをキャンセルすると VM の状態が移送元ホストと移送先ホストにまたがっているため，VM の実行を継続できなくなる．キャンセラブルなポストコピー VM 移送 [8] では，移送先

ホストで稼働中の VM が更新したメモリを移送元ホストに転送してメモリの同期をとる．これにより，マイグレーションをキャンセルした時に移送元ホストで VM の実行を継続することができる．DCmigrate をポストコピーマイグレーションに適用すれば，移送先を変更する際にマイグレーションをキャンセルできるようにするために VM のメモリを同期する必要はなくなる．

7. まとめ

本稿では，VM マイグレーションをキャンセルすることなく，移送先ホストを柔軟に変更することができるシステム DCmigrate を提案した．DCmigrate は移送先ホストを変更した後，移送元ホストからはまだ転送していない VM の状態だけを転送する．一方，古い移送先ホストも受信済みの VM の状態を新しい移送先に転送する．このように VM の状態を並列に転送することで，移送先ホストの変更後のマイグレーションを高速化する．実験により，DCmigrate を用いて移送元ホストにおいて移送先ホストを切り替え，新しい移送先ホストに VM をマイグレーションできることが確認できた．

今後の課題は，古い移送先ホストから新しい移送先ホストに VM の状態を並列転送できるようにすることである．実装が必要な機能としては，古い移送先ホストから新しい移送先ホストへの VM の状態の転送および，新しい移送先ホストでの VM 状態の並列受信である．また，移送先ホストの変更後のマイグレーション時間を短縮するために，移送元ホストと古い移送先ホストそれぞれが転送を行う VM の状態の送信量を最適化できるようにする必要がある．例えば，マイグレーションが半分以上進んでいる時に移送先ホストを変更すると，古い移送先ホストから転送される VM の状態の割合が高くなり，移送元ホストからの転送との十分な並列化が行えない場合がある．そのため，ホストの負荷やネットワークの速度差も考慮して，2 つのホストからの転送がほぼ同時に完了するようにする必要がある．

謝辞 本研究の一部は，JST，CREST，JPMJCR21M4 の支援を受けたものである．また，本研究の一部は，国立研究開発法人情報通信研究機構の委託研究（05501）による成果を含む．

参考文献

[1] Amazon Web Services, Inc. Amazon EC2 High Memory Instances. https://aws.amazon.com/ec2/instance-types/high-memory/.
[2] F. Bellard. Qemu. https://www.qemu.org/.
[3] T. Kashiwagi and K. Kourai. Flexible and Efficient Partial Migration of Split-memory VMs. pages 248–257, 2020.
[4] A. Nadjaran Toosi and R. Buyya. Acinonyx: Dynamic Flow Scheduling for Virtual Machine Migration in SDN-Enabled Clouds. pages 886–894, 2018.
[5] Red Hat, Inc. libvirt: The virtualization API. https://libvirt.org/.
[6] M. Suetake, T. Kashiwagi, H. Kizu, and K. Kourai. S-memV: Split Migration of Large-Memory Virtual Machines in IaaS Clouds. pages 285–293, 2018.
[7] S. Tauchi, K. Kourai, and L. Rahim. Optimizing VMs across Multiple Hosts with Transparent and Consistent Tracking of Unused Memory. pages 467–477, 2021.
[8] 小川遥加, 山田浩史, 十文字優斗, and 阿部芳久. キャンセラブルなポストコピー VM 移送. pages 1–12, 2017.