

研究最終報告

シンクライアントシステムのための通信状況に応じた
仮想マシンの最適配置手法

**OVCT (Optimal placement method of Virtual machine in accordance with
Communication situation for Thin client)**

指導教授

慶應義塾大学 理工学部 情報工学科

重野 寛 教授 印

担当教授

(日工専) 情報工学科

清水 富門 教授 印

報告者

(日工専) 情報工学科

第 53 期研究科生 深堀 秀治 印

研究期間

2013 年 4 月 1 日 ～ 2014 年 3 月 26 日

報告年月日

2014 年 3 月 26 日

要旨

近年，クライアント側に入出力機能のみを持たせ，サーバ側でアプリケーションやファイルなどの資源を管理するシンククライアントシステムが多くの企業に導入されている．シンククライアントシステムでは，アプリケーションやファイルなどの資源を一元管理することができるため管理コストが削減でき，クライアント側に資源を持たないため情報漏洩を防止することもできる．また，導入のネックとなっていた導入コストの肥大化もサーバ側を仮想化することで1台の物理マシンを複数のクライアントが利用可能になったことにより解決されつつある．しかしながら，シンククライアントシステムでは，入出力がネットワークを介するため，シンククライアント端末とサーバの通信が多くのネットワーク機器を経由すると性能が低下してしまう．

そこで，本研究では，シンククライアントシステムの性能低下防止とネットワーク機器の負荷を軽減するため，ライブマイグレーションと **OpenFlow** を用いる．ライブマイグレーションは仮想マシンを停止することなく別の物理マシンに移動する技術であり，ライブマイグレーションを利用し，シンククライアント端末と仮想マシンを近隣に配置することでシンククライアントシステムの性能低下を最小限にすることができる．しかし，ライブマイグレーションを行うとネットワーク機器のルーティング情報などのネットワーク設定を手動で行わなければならないため，管理者の負担が大きい．そこで，本研究ではネットワーク仮想化技術 **OpenFlow** を用いてライブマイグレーションを行った際，自動でネットワーク設定を変更するシステムを提案する．

目次

1. 緒言	1
1.1. 背景	1
1.2. 目的	2
2. 使用技術・技術動向.....	3
2.1. シンククライアントシステム.....	3
2.1.1. シンククライアントシステムとは.....	3
2.1.2. シンククライアントシステムの種類.....	4
2.1.3. シンククライアントのユースケース	5
2.2. サーバ仮想化	6
2.2.1. サーバ仮想化とは.....	6
2.3. ライブマイグレーション.....	7
2.3.1. ライブマイグレーションとは.....	7
2.3.1. ライブマイグレーションの手順.....	7
2.4. OpenFlow.....	8
2.4.1. OpenFlowとは.....	8
2.4.2. OpenFlowの動作の流れ.....	8
3. 研究内容	10
3.1. 既存手法	10
3.1.1. PBA (Pattern-based Virtual Desktop Allocation)[4]	10
3.2. 提案手法	11
3.2.1. 概要	11
3.2.2. 通信の検知（第1フェーズ）	12
3.2.3. 配置場所の選択（第2フェーズ）	13
3.2.4. ライブマイグレーション（第3フェーズ）	14
3.2.5. ネットワーク設定の変更（第4フェーズ）	15
4. 提案実装	16
4.1. 実装方法	16
4.1.1. 仮想マシン.....	16
4.1.2. ライブマイグレーション.....	17
4.1.3. OpenFlowコントローラ	18
4.1.4. OpenFlowスイッチ.....	19
4.1.5. シンククライアント.....	20
4.2. 実装環境	21
4.2.1. マシン	21

4.2.2. ネットワーク	22
4.3. 実装構成	23
4.3.1. システム概要図	23
4.3.1. システム詳細図	24
5. 評価	25
5.1. 通信速度	25
5.1.1. 評価方法	25
5.1.1. 評価結果	26
5.2. パケット中継数・量	27
5.2.1. 評価方法	27
5.2.2. 評価結果	27
5.3. 処理時間	29
5.3.1. 評価方法	29
5.3.2. 評価結果	29
5.4. 結果の考察	30
6. 結言	31
6.1. まとめ	31
6.2. 今後の課題	31
6.2.1. 仮想マシンの配置場所最適化	31
6.2.2. 実用的なネットワークへの対応	31
7. 謝辞	32
8. 参考文献	33

1. 緒言

本節では，本研究の背景や目的について述べる．

1.1. 背景

近年，シンククライアントシステムが情報漏洩防止，管理コスト削減の観点から多くの企業で導入されている．しかしながら，シンククライアントシステムには大きく2つの問題点がある．1点目は，最近まで1ユーザあたりにシンククライアント端末と物理的なサーバが1台ずつ必要であったため，導入コストが高価な点であった．そこで最近，導入されているのがサーバ仮想化である．サーバを仮想化することでお互い干渉することなく1台の物理サーバを複数のユーザで共有して利用することができる．下図では，シンククライアントに対する仮想化導入率を示している．2008年には10%未満であったが2013年には35%と非常に多くの割合で導入されていることが分かる．また，仮想化することで仮想マシンを停止することなく別の物理マシンに移動する技術であるライブマイグレーションが実行できるようになる．

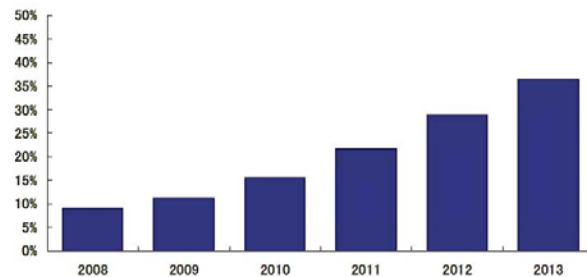


図 1 国内法人向け PC およびシンククライアント稼働台数に対する
クライアント仮想化導入率予測[1]

2点目は，画面描画やキーボード入力などの入出力の転送がネットワークを介するため，動作が遅延してしまう点である．2点目の問題点は，ネットワーク機器の性能，ネットワーク回線の速度に依存してしまう．そこで，本稿ではネットワーク機器や回線に依存せずにシンククライアントシステムの性能を向上させることを目的に，サーバを仮想化することにより可能になったライブマイグレーションを使い，シンククライアント端末，仮想マシンを近隣に配置することで，この問題点を解決する手法を提案する．

1.2. 目的

本研究の目的として主に 2 点を挙げる．1 点目は，シンククライアントシステムの性能を向上させることである．シンククライアントシステムの性能向上のためには，シンククライアント端末，仮想マシン間の通信速度を向上させる必要があり，本研究では，ライブマイグレーションを用い，シンククライアント端末と仮想マシンを近隣に配置することで達成する．2 点目は，ネットワーク負荷の軽減である．ネットワーク負荷軽減のためには，シンククライアント端末，仮想マシン間のパケットを中継するネットワーク機器を減らす必要があり，本研究では，1 点目と同様にライブマイグレーションを用い，シンククライアント端末と仮想マシンを近隣に配置することで達成する．また，筆者は業務でシンククライアントに携わっており，本研究を通じて **OpenFlow** やライブマイグレーションなどの最先端の技術に触れ，業務に活かすことも目的とする．

2. 使用技術・技術動向

本節では，本研究で使用する主な技術について述べる．

2.1. シンククライアントシステム

2.1.1. シンククライアントシステムとは

シンククライアントシステムとは，クライアント側に入出力機能のみを持たせ，サーバ側で実際の処理やアプリケーション，ファイルなどの資源を管理するシステムである．シンククライアントシステムでは，アプリケーション，ファイルなどの資源を保持しているサーバを一元管理することができるため管理コストが削減できる．また，クライアントに資源を持たないため情報漏洩を防止できる．しかし，入出力がネットワークを介するため，動作が遅延してしまうという欠点もある．

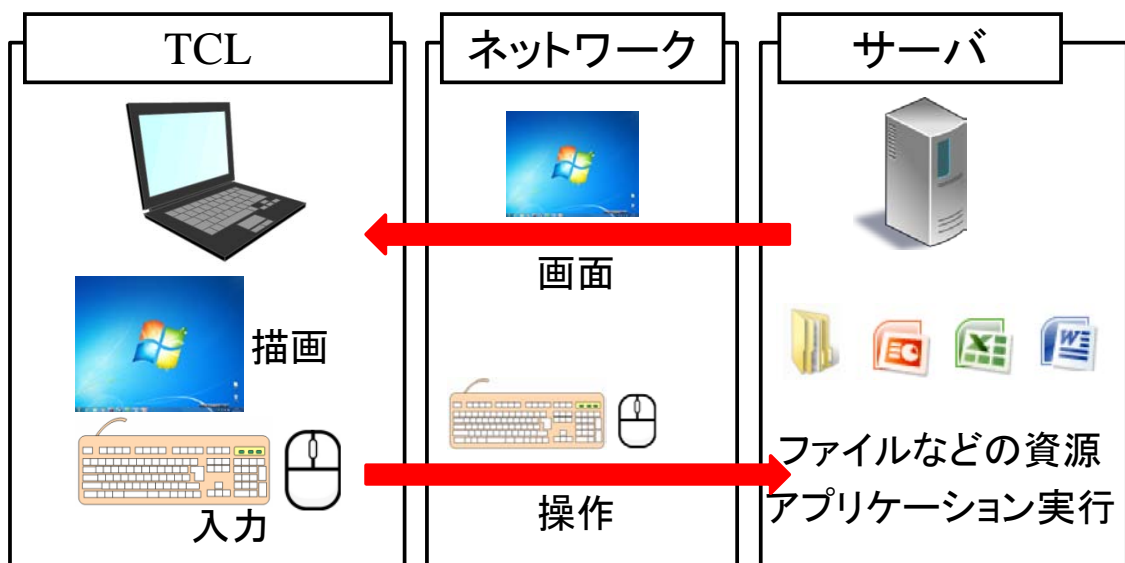


図 2 シンククライアントシステム

2.1.2. シンククライアントシステムの種類

シンククライアントシステムは、「クライアントブレード型」「ターミナルサービス型」「仮想PC型」の主に3種類がある。

クライアントブレード型は、シンククライアント端末1台に対して1台の物理サーバを割り当てる方式である。本型は、ユーザが独立した物理サーバを使用できるため、アプリケーションの互換性などを考慮する必要が無く自由度が高い。しかし、ハードウェア台数が多くなってしまうため導入コストや管理コストが高くなる。

ターミナルサービス型は、1台のサーバにひとつのサーバOSを稼働させ、その上で複数のクライアント向けアプリケーションを稼働させる方式である。本型は、1台の物理サーバを複数のユーザで共有できるため、導入コストや管理コストが低い。また、ユーザ間でサーバのハードウェア資源を自由に共用できるためハードウェア資源を有効利用できる。しかし、複数のユーザで1つのOSを共有するため、自由度は低い。

仮想PC型は、1台のサーバでハイパーバイザーを稼働させ、その上で仮想マシンを複数稼働させる方式である。本型は、1台の物理サーバを複数のユーザで共有できるため、導入コストや管理コストが低い。また、ユーザ間でサーバのハードウェア資源を自由に共用できるためハードウェア資源を有効利用できる。そして、ユーザごとにOSを利用できるため、自由度も高い。本研究でも、コストが低く自由度の高い仮想PC型を利用し提案を行う。

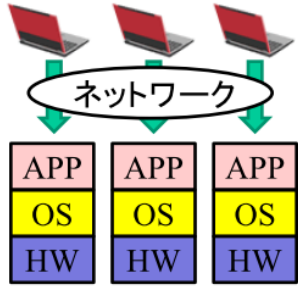
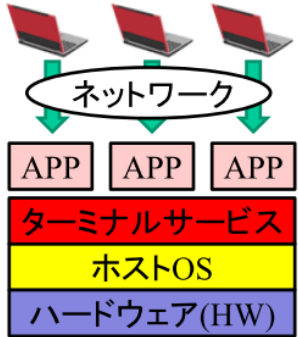
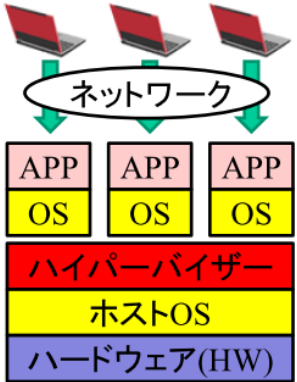
名称	クライアントブレード型	ターミナルサービス型	仮想PC型
方式			
コスト	高	低	低
自由度	高	低	高

図 3 シンククライアントシステムの種類

2.1.3. シンククライアントのユースケース

シンククライアントのユースケースは大きく2点ある。1点目は、通勤などの移動中の利用である。シンククライアント端末は、端末自体に情報を持たないため、紛失しても情報漏洩に繋がらない。そのため、紛失する可能性の高い移動中の利用が可能である。2点目は、他の拠点へ出張・異動である。通常、PCは出張・異動先に持ち出さなければ自身の環境を再現することは不可能だが、シンククライアントであれば、出張・異動先にシンククライアント端末があれば、端末自体を持ち出さなくても自身のサーバに接続することで、自身の環境を再現することが出来る。

本研究では、2点目の「他の拠点へ出張・異動」について着目する。下図のように東京から福岡に出張・異動する際に、シンククライアント端末を持ち出さなくても自身のサーバに接続が出来れば環境を再現することが出来る。しかし、シンククライアント端末とサーバの距離が離れてしまうと、距離に応じて回線の混雑や減衰による遅延が発生しやすくなり性能が低下してしまうという問題点がある。そこで、本研究では福岡に出張した際は、サーバである仮想マシンを福岡に配置するなど、シンククライアントの位置に応じて仮想マシンを配置し、シンククライアントシステムの性能低下を防止しようと考えた。

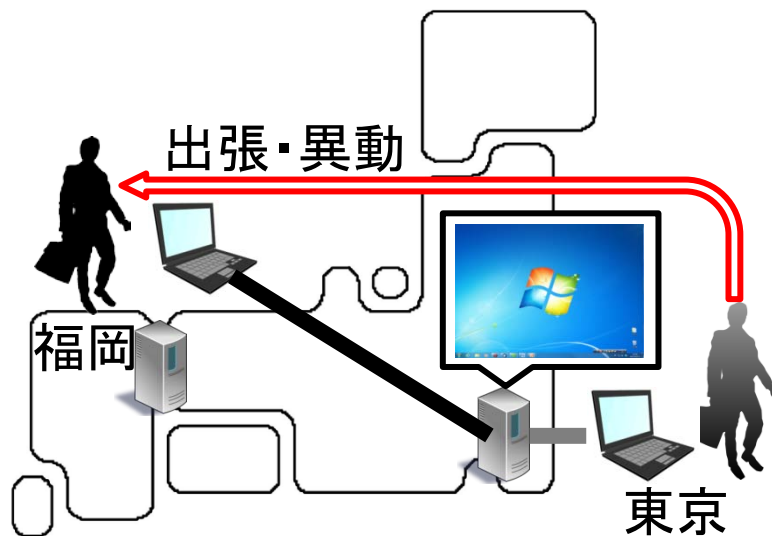


図 4 シンククライアントのユースケース

2.2. サーバ仮想化

2.2.1. サーバ仮想化とは

サーバ仮想化とは、1 台の物理マシンを仮想的なマシンに分割し、それぞれ別の OS やアプリケーションを動作させる技術である。サーバ仮想化によって、お互い干渉することなく 1 台の物理サーバを複数のユーザで共有して利用することができるため、導入コストや運用コストを削減できる。また、稼働しているサーバの資源に合わせて、仮想サーバを追加、移動することで余剰なリソースを無くし、サーバを有効活用できる。また、仮想マシンを移動させる技術、マイグレーションが可能になる

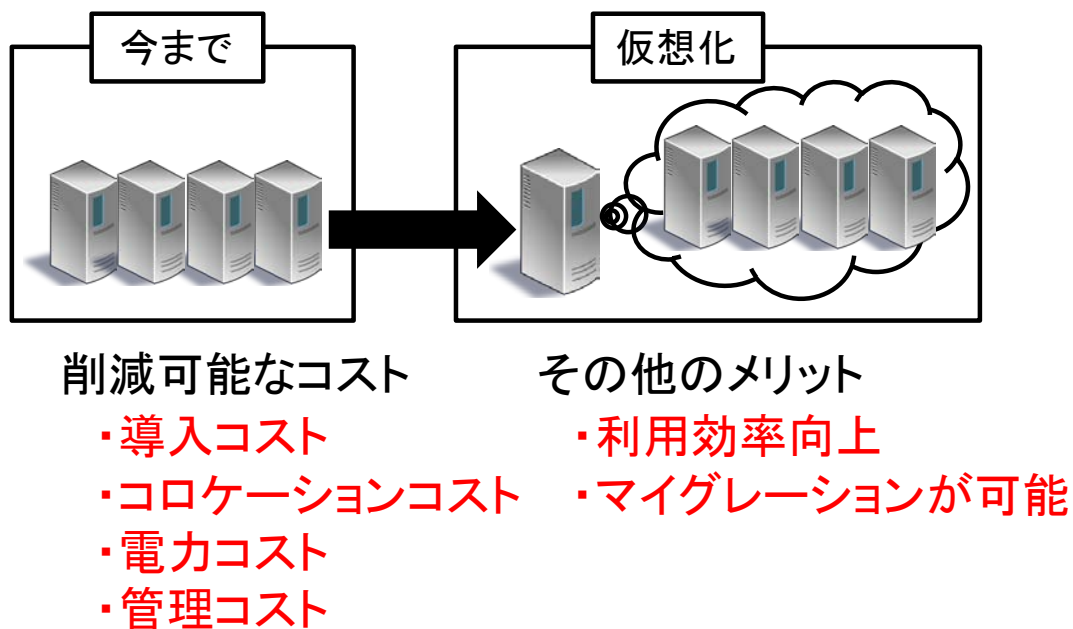


図 5 サーバ仮想化

2.3. ライブマイグレーション

2.3.1. ライブマイグレーションとは

ライブマイグレーションとは、仮想マシンを停止することなく別の物理マシンに移動する技術である。ライブマイグレーションによって、物理マシンの負荷やネットワークの負荷に応じて仮想マシンを容易に移動することができるため、仮想マシンの集約や分散が容易にできる。

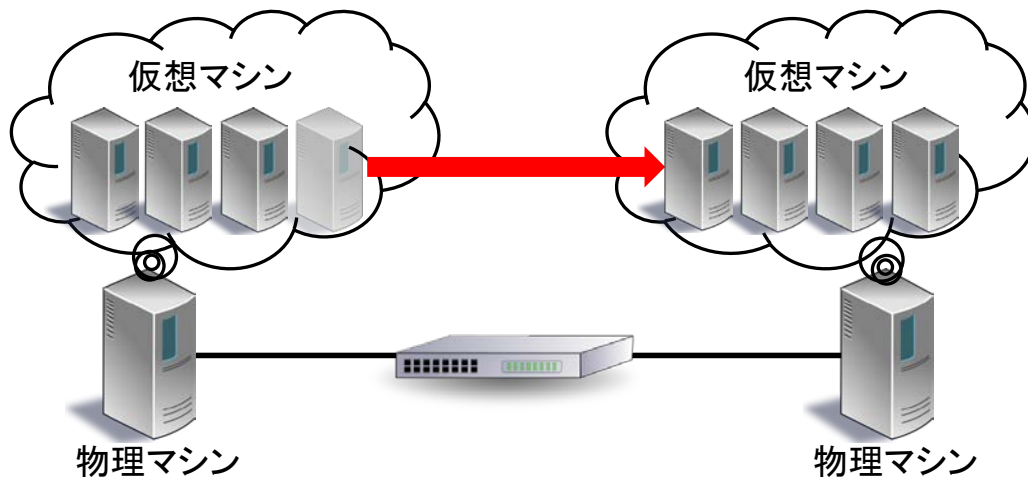


図 6 ライブマイグレーション

2.3.1. ライブマイグレーションの手順

下記にライブマイグレーションの手順[2]について示す。

- ① 共有ファイルシステム上に仮想マシンのイメージファイルを格納する
- ② 移行元の物理マシンで仮想マシンを起動する
- ③ 移行先の物理マシンで仮想マシンを待ち受け状態で起動する
- ④ マイグレーション開始処理を行う
- ⑤ 移行元の物理マシンの仮想マシンの仮想 CPU の状態を移行先の物理マシンに転送する
- ⑥ 移行先の物理マシンの仮想 CPU を再開
移行元の物理マシンの仮想 CPU を停止
- ⑦ 移行先の物理マシンの保持していないメモリページへのリクエストが発生した場合、該当ページとその周辺ページを移行元の物理マシンからリアルタイムで取得する。
バックグラウンドで移行元の物理マシン上で動作していた仮想マシンのメモリページを先頭から順に転送する

2.4. OpenFlow

2.4.1. OpenFlowとは

OpenFlow[3]とは、SDN (Software Define Network) の一種でネットワークをソフトウェアで制御する技術である。ネットワーク機器を制御部である OpenFlow コントローラと駆動部である OpenFlow スイッチに分け、1 台の OpenFlow コントローラで複数の OpenFlow スイッチを制御する。OpenFlow コントローラで条件と処理を定義する。そして定義を基に OpenFlow スイッチがフローテーブルを作成し、処理を行う。OpenFlow では、OpenFlow コントローラでネットワークを一元管理できるため、ネットワーク構成の変更が容易にできる。

2.4.2. OpenFlowの動作の流れ

下図においてホスト A からホスト B へパケットを送信する場合の OpenFlow の動作を下記に示す。

① パケット受信

ホスト A から送信されたパケットを受信する

② フローテーブルを参照

フローテーブルを参照し、受信したパケットが条件に当てはまるか調査する。当てはまった場合は、条件に合った処理を行い。当てはまらなかった場合は、③へ。

フローテーブル…OpenFlow スイッチが保持するルーティングのためのテーブル。条件(送信元・宛先 IP アドレス/MAC アドレス/ポート番号/VLAN ID/プロトコルなど)と対応した処理(転送/破棄/書換)が格納されている。また、フローテーブルの中身をフローエントリといいエントリごとに統計情報(送信パケット数/送信パケット量)を保持している。

③ コントローラに処理を問い合わせ

該当する条件が無かった場合は、コントローラにパケットをどのように処理すればよいか問い合わせる。

④ 処理を指示

コントローラは、問い合わせの内容から処理を決定し、スイッチに指示する。

⑤ 処理をテーブルに追加

スイッチは、受け取った処理をフローテーブルに書き込む

⑥ パケット転送

受け取った指示通りに処理を行う

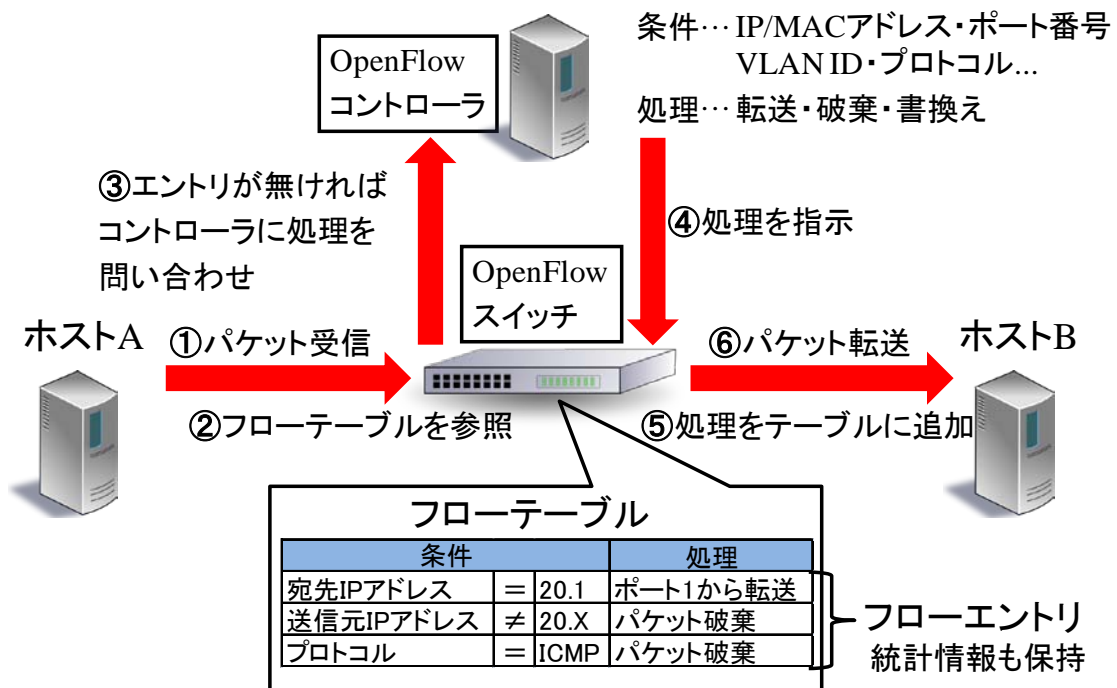


図 7 OpenFlow の動作の流れ

3. 研究内容

本節では、本研究の既存手法と提案手法について述べる。既存手法については OpenFlow が比較的新しい技術であることとシンククライアントが特殊なシステムであることから少なくなっている。

3.1. 既存手法

3.1.1. PBA (Pattern-based Virtual Desktop Allocation)[4]

PBA は、仮想マシンの資源使用量が一定周期でパターン化することを利用して、各使用パターンの相関を考慮し、相関の低い仮想マシン同士を同じ物理サーバへ配置することで、資源の競合を抑える手法である。下図のような、3 種類の仮想マシンの使用パターンが存在したとき、これら異なるパターンを 1 台の物理マシンに配置することで資源の競合を抑え、仮想マシンのパフォーマンスを維持しつつ、物理マシンの台数を削減することを目的としている。本研究のシンククライアントシステムの性能向上とは観点が異なるが、シンククライアントにおける仮想マシンの配置という視点で同様であったため既存手法とした。

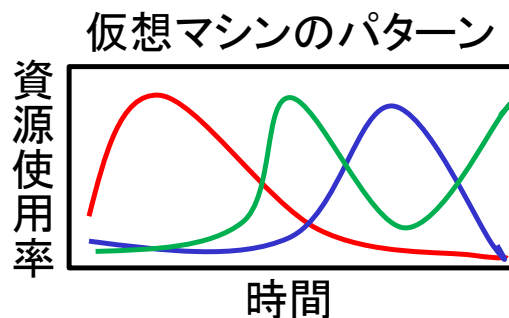


図 8 資源使用率の使用のパターン

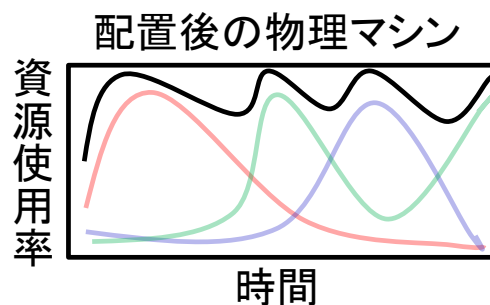


図 9 配置後の物理マシンの資源使用率

3.2. 提案手法

本章では，シンククライアント端末と仮想マシン間の通信に応じて仮想マシンを配置するために，シンククライアントシステムのための通信状況に応じた仮想マシンの最適配置手法 OVCT (Optimal placement method of Virtual machine in accordance with Communication situation for Thin client) を提案する．

3.2.1. 概要

OVCT は 1.2 の目的で挙げた「シンククライアントシステムの性能向上」と「ネットワーク負荷の軽減」を達成するために，OpenFlow のネットワーク全体を監視する機能，ネットワーク設定を自動変更する機能とライブマイグレーションを使い，シンククライアント端末，仮想マシン間の経路を自動的に短縮する．

OVCT による仮想マシンの配置は 4 つのフェーズに分けられる．第 1 フェーズは，通信の検知である．通信の検知では，シンククライアント端末，仮想マシン間の通信を検知する．第 2 フェーズは，配置場所の選択である．どの物理マシンに仮想マシンを配置すれば最適であるのか検討し，選択する．第 3 フェーズは，ライブマイグレーションである．第 2 フェーズで選択された物理マシンへ仮想マシンを配置する．第 4 フェーズは，ネットワーク設定の変更である．ライブマイグレーションによって必要になったネットワーク設定を変更する．以降，各フェーズの動作について述べる．

3.2.2. 通信の検知（第 1 フェーズ）

通信の検知では、OpenFlow を使いシンククライアント端末と仮想マシンの通信を検知する。OpenFlow コントローラでネットワーク全体の通信を監視し、宛先ポート番号 $dp = 5900$ (VNC) である通信を検知する。そして、送信元 IP アドレスからシンククライアント端末が接続されているスイッチ TCL_SWID、宛先 IP アドレスから仮想マシンが接続されているスイッチ VM_SWID を調査し、 $TCL_SWID \neq VM_SWID$ であった場合、シンククライアントと仮想マシンが遠方に配置されている可能性があると判断し、第 2 フェーズへと移行する。

下図では、シンククライアント端末が SW1、仮想マシンが稼働している物理マシン 2 が SW3 に接続している状態で、シンククライアント端末から仮想マシンに VNC 接続しているため、 $TCL_SWID \neq VM_SWID$ となり、シンククライアントと仮想マシンが遠方に配置されていると判断される。

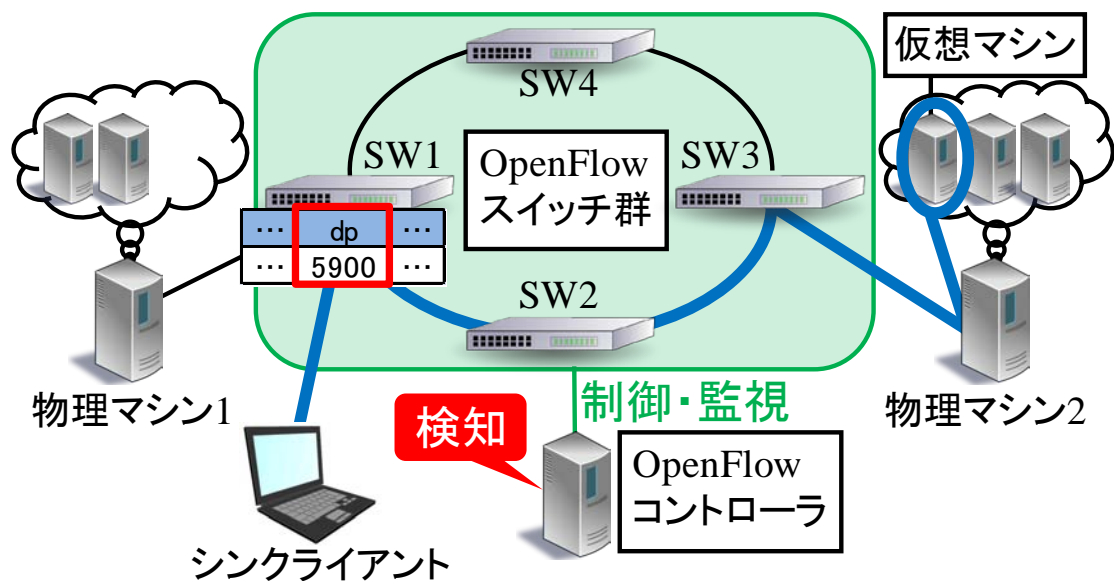


図 10 シンククライアントと仮想マシンの通信を検知

3.2.3. 配置場所の選択（第2フェーズ）

配置場所の選択では，第1フェーズで検知したパケットから，現在，仮想マシンがどの物理マシン上で稼働しているか調査する．また，ネットワーク構成から仮想マシンをどの物理マシンに配置すればシンククライアント端末，仮想マシン間のネットワーク経路が最短になるか調査し選択する．今回の実装では，シンククライアント端末と仮想マシンが同一スイッチ配下に配置されている場合を最短であるとした．そして，選択された物理マシンへSSHで接続し，空きCPU使用率と空きメモリを調査する．指定した空きCPU使用率と空きメモリが存在した場合，第3フェーズへと移行する．

下図では，シンククライアント端末と同じSW1に接続されている物理マシン1が配置場所として選択される．そして，物理マシン1のCPU使用率と空きメモリを調査し，空き容量が十分であるため，配置場所として物理マシン1が決定される．

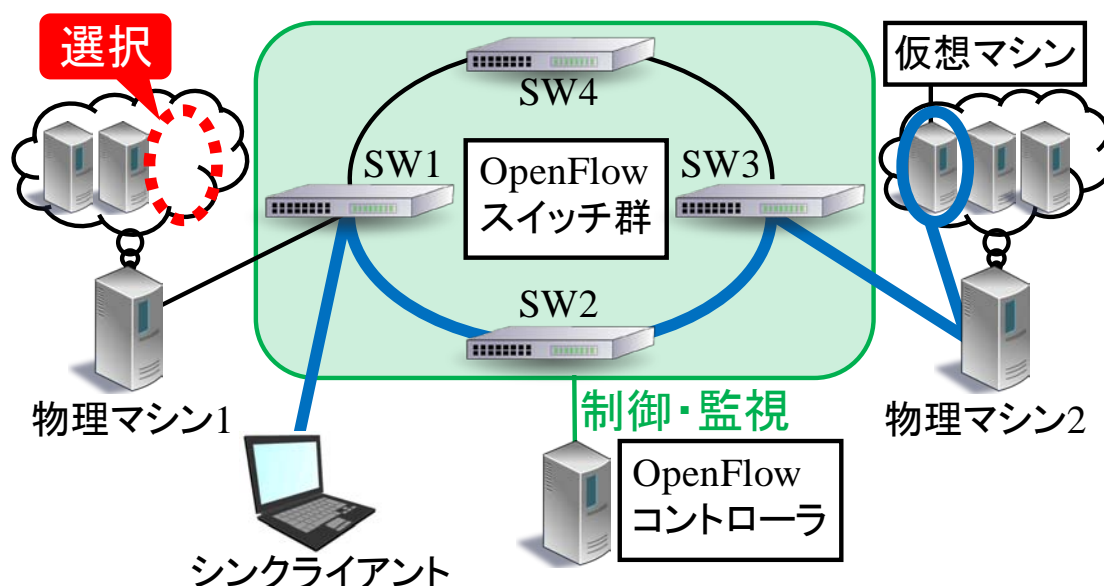


図 11 配置場所の選択

3.2.4. ライブマイグレーション（第3フェーズ）

ライブマイグレーションでは、第2フェーズで選択された配置場所へと仮想マシンをライブマイグレーションする。現在、仮想マシンが稼働している物理マシンへSSHで接続し、第2フェーズで選択された物理マシンへ仮想マシンをライブマイグレーションするように指示する。ライブマイグレーションが完了後、第4フェーズへと移行する。

下図では、物理マシン2から物理マシン1へライブマイグレーションしている。

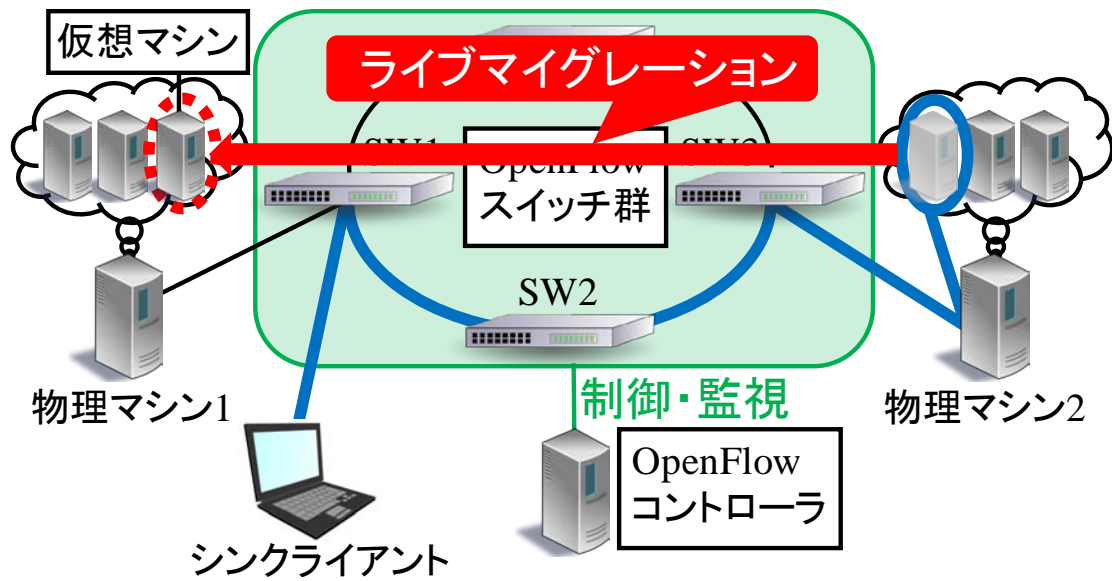


図 12 仮想マシンをシンククライアントの近隣にライブマイグレーション

3.2.5. ネットワーク設定の変更（第 4 フェーズ）

ネットワーク設定の変更では、OpenFlow を使い、第 3 フェーズでライブマイグレーションされた仮想マシンに対するルーティング情報を自動で変更する。まず、ライブマイグレーションした仮想マシンに対するルーティング情報を OpenFlow スイッチから削除する。そして、変更した仮想マシンに対するルーティング情報を OpenFlow スイッチに追加する。

下図では、仮想マシンが物理マシン 2 上で動作していたため、仮想マシン宛のルーティングは物理マシン 2 になっている。ライブマイグレーション後は、仮想マシンは物理マシン 1 上で動作するため、仮想マシン宛のフローエントリをすべて削除したのち、仮想マシン宛の packets を物理マシン 1 に送るようにエントリを追加する。

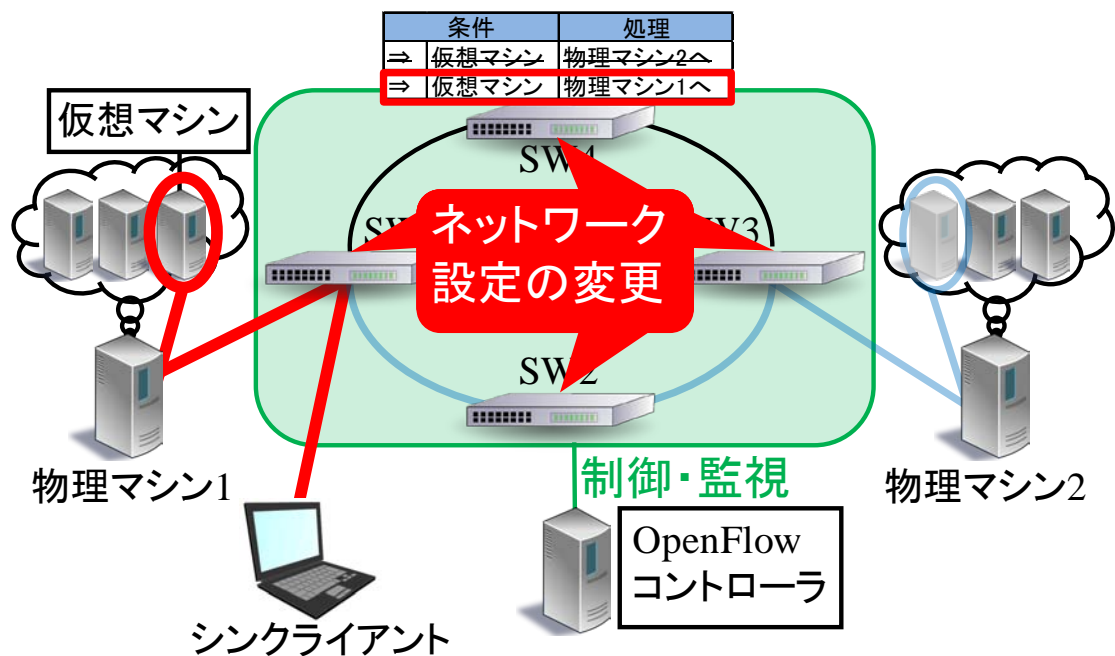


図 13 ネットワーク設定の変更

4. 提案実装

本章では、実装した OVCT についての実装方法や構成について述べる

4.1. 実装方法

本章では、OVCT の実装方法について述べる

4.1.1. 仮想マシン

本提案では、ライブマイグレーションを行うためにシンクライアントの接続先として仮想マシンを使用した。仮想化のハイパーバイザとして Linux のカーネルに統合されており、導入が容易で取り扱いやすい KVM を用いた。

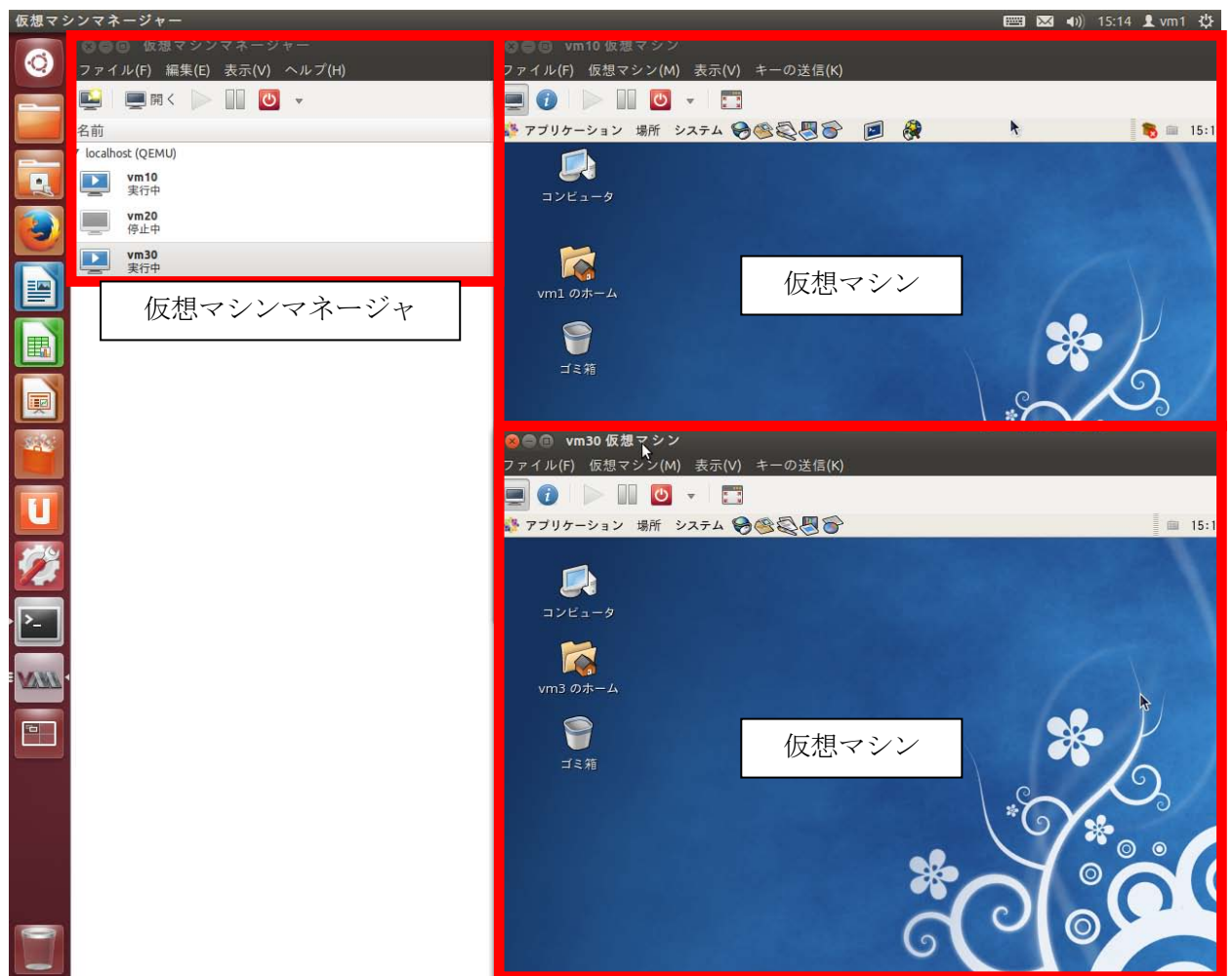


図 14 仮想マシンが動作する様子

4.1.2. ライブマイグレーション

本提案では、仮想マシンの配置場所を変更するためにライブマイグレーションを用いる。ライブマイグレーションを使用するためには、仮想マシンのイメージファイルをネットワーク上に格納しなければならない。そこで、今回は NFS (Network File System) 上にイメージファイルを格納した。そして、各物理マシンで NFS を同一ディレクトリにマウントすることで、ライブマイグレーションを行った。物理マシン 1、物理マシン 2 で NFS 上の「/home/NFS/NFS」を「/mnt-nfs」にマウントする。そうすることで、仮想マシンのイメージファイルである「vm1.img」に同じディレクトリパスでアクセスすることが出来る。このように仮想マシンのイメージファイルを物理マシン同士で共有することで、CPU とメモリの内容のみをネットワークを介して転送することで、ライブマイグレーションが可能になる。

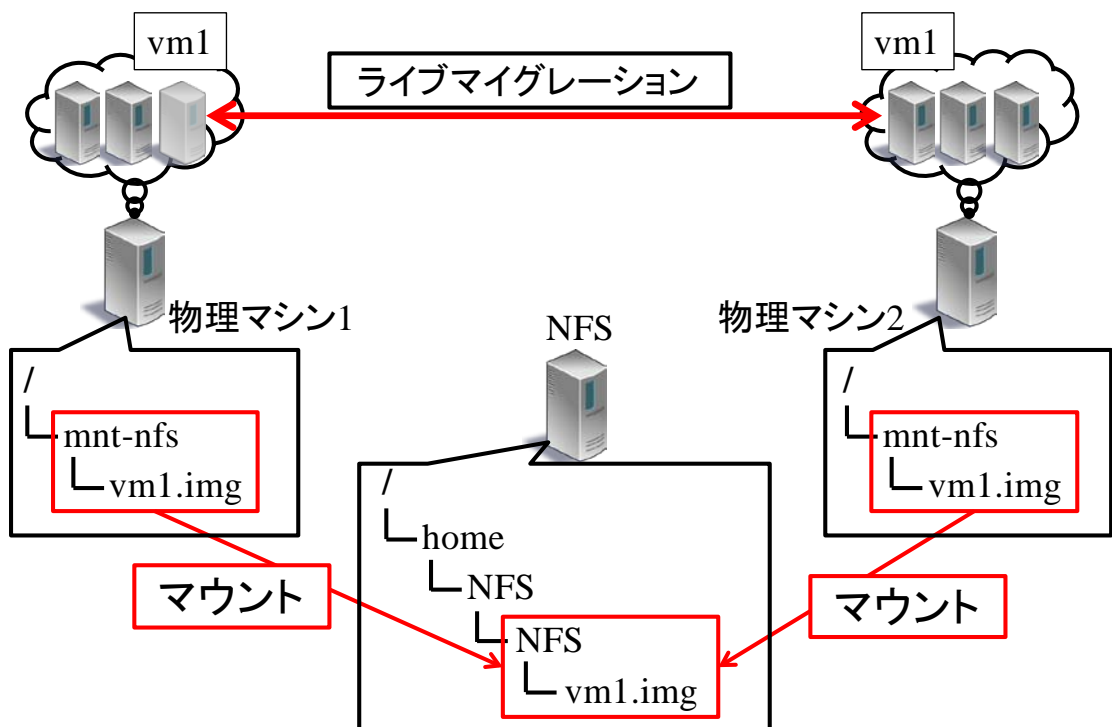


図 15 ライブマイグレーション時の構成

4.1.3. OpenFlowコントローラ

本提案では、通信の検知及びネットワーク設定の自動変更のために OpenFlow を用いる。OpenFlow を動作させるには OpenFlow コントローラと OpenFlow スイッチが必要である。OpenFlow コントローラの開発には、他の開発環境よりも生産性に優れている[5]プログラミングフレームワークである Trema を使用した。OpenFlow コントローラでは、統計情報を取り扱うことも可能で、フローエントリに有効期限を設け、有効期限が切れた際にそのエントリが何回実行されどれだけのパケット量を転送したかが分かる。

```

simple-router.rb
146 { message.ipv4_daddr }"
147   info " VNC: #{ message.ipv4_daddr } → #
148   { pm_dst } → #{ message.ipv4_daddr }"
149   info " LM: #{ pm_dst } → #
150   { message.ipv4_daddr } → #{ mg_dst }"
151
152   #LM移行元とVMのホスト名を調べる
153   pm_dst_name = Resolv.getname(pm_dst.to_s)
154   vn_name = Resolv.getname(message.ipv4_daddr.to_s)
155
156   #マイグレーション実行
157   cmd1 = "virsh migrate --live --persistent "
158   cmd2 = " qemu+tcp://root@"
159   cmd3 = "/system"
160
161   cmd = cmd1 + vn_name + cmd2 + mg_dst.to_s + cmd3
162
163   info " マイグレーション実行 #{ Time.now }"
164   info " cmd: #{ cmd }"
165
166   Net::SSH.start(pm_dst.to_s, pm_dst_name, :keys =>
167     ["/home/openflow/.ssh/id_rsa"], :password => '') do |ssh|
168     stdout = ""
169     stderr = ""
170     ssh.exec!(cmd) do |channel, stream, data|
171       stdout << data if stream == :stdout
172       stderr << data if stream == :stderr
173     end
174     print(stdout)
175   end
176   info " マイグレーション完了 #{ Time.now }"
177
178   #ルーティング情報の書き換え
179   info " -----ネットワーク設定変更のための情報を表
180   示-----"
181   info " TCLが接続されたスイッチ: #{ dpid }"
182   info " 接続先PMが接続されたスイッチ: #{ pm_dip }"
183   info " 接続元TCLのIPアドレス: #
184   { message.ipv4_daddr }"
185   info " 接続先PMのIPアドレス: #
186   { pm_dst }"
187   info " 接続先VMのIPアドレス: #
188   { message.ipv4_daddr }"
189   info " 接続元TCLのIPアドレス: #
190   { vn_name }"
191   info " LM移行元のIPアドレス: #{ pm_dst }"

```

```

テキストエディター
openflow@openflow:~/デスクトップ/trema-0.4.5/src/examples
接続元TCLのIPアドレス: 192.168.150.10
接続先PMのIPアドレス: 192.168.200.10
接続先VMのIPアドレス: 192.168.120.10
接続先VMのホスト名: vm10
LM移行元のIPアドレス: 192.168.200.10
LM移行先のIPアドレス: 192.168.100.10
移行するVMのIPアドレス: 192.168.120.10
VNC: 192.168.150.10 → 192.168.200.10 → 192.168.120.10
LM: 192.168.200.10 → 192.168.120.10 → 192.168.100.10
-----設定ファイルを変更-----
-----古いフローエントリを削除-----
-----古いフローエントリを削除完了-----
古いフローエントリを削除 Mon Mar 10 16:06:00 +0900 2014
VNC接続検知!!
異なるスイッチへ接続中!! Mon Mar 10 16:06:26 +0900 2
014
ssl: 192.168.100.10
マイグレーションOK Mon Mar 10 16:06:33 +0900 2014
-----検知した情報とマイグレーション情報を表示-----
TCLが接続されたスイッチ: 2
接続先PMが接続されたスイッチ: 3
接続元TCLのIPアドレス: 192.168.150.10
接続先PMのIPアドレス: 192.168.200.10
接続先VMのIPアドレス: 192.168.120.30
LM移行元のIPアドレス: 192.168.200.10
LM移行先のIPアドレス: 192.168.100.10
移行するVMのIPアドレス: 192.168.120.30
VNC: 192.168.150.10 → 192.168.200.10 → 192.168.120.30
LM: 192.168.200.10 → 192.168.120.30 → 192.168.100.10
マイグレーション実行 Mon Mar 10 16:06:33 +0900 2014
cmd: virsh migrate --live --persistent vm30 qemu+tcp://root
@192.168.100.10/system
マイグレーション完了 Mon Mar 10 16:09:27 +0900 2014
-----ネットワーク設定変更のための情報を表示-----
TCLが接続されたスイッチ: 2
接続先PMが接続されたスイッチ: 3
接続元TCLのIPアドレス: 192.168.150.10
接続先PMのIPアドレス: 192.168.200.10
接続先VMのIPアドレス: 192.168.120.30
接続先VMのホスト名: vm30
LM移行元のIPアドレス: 192.168.200.10
LM移行先のIPアドレス: 192.168.100.10
移行するVMのIPアドレス: 192.168.120.30
VNC: 192.168.150.10 → 192.168.200.10 → 192.168.120.30
LM: 192.168.200.10 → 192.168.120.30 → 192.168.100.10
-----設定ファイルを変更-----
-----古いフローエントリを削除-----
-----古いフローエントリを削除完了-----
古いフローエントリを削除 Mon Mar 10 16:09:27 +0900 2014

```

図 16 Trema でのプログラミング

使用するメソッドとして主に下記のようなものがある。

メソッド名	説明
start	プログラム開始時に呼び出されるメソッド
packet_in	未知のパケットが到着した際に呼び出されるメソッド
send_flow_mod_add	フローエントリを追加するメソッド
send_flow_mod_delete	フローエントリを削除するメソッド
send_flow_mod_modify	フローエントリを書換するメソッド
send_packet_out	パケットを転送するメソッド

図 17 Trema の主なメソッド

4.1.4. OpenFlowスイッチ

OpenFlow スイッチには、安価で入手が容易な WHR-G301N を使用した。WHR-G301 は通常はルータであるが、有志が公開しているファームウェア[6]を書き換えれば、性能は期待できないものの、OpenFlow スイッチとして利用することが出来る。作成した OpenFlow スイッチは Telnet で接続することで設定の変更が行える。また、OpenFlow からの指示でフローテーブルにエントリが書き込まれ、指示された通りに処理を行う。



図 18 WHR-G301N

フローエントリの実際のデータを下図に示す。

```
openflow@openflow: ~  
root@OpenWrt:/# dpctl dump-flows unix:/var/run/dp0.sock  
stats_reply (xid=0x9184219a): flags=none type=1(flow)  
  cookie=2040, duration_sec=213s, duration_nsec=406000000s, table_id=1, priority=65535, n_packets=206, n_bytes=219208, idle_timeout=0, hard_timeout=600, ip, nw_src=192.168.100.10, nw_dst=192.168.20.10, actions=mod_dl_src:00:00:00:00:10:40, mod_dl_dst:00:14:85:03:1b:21, output:4  
  cookie=2037, duration_sec=543s, duration_nsec=519000000s, table_id=1, priority=65535, n_packets=481, n_bytes=167490, idle_timeout=0, hard_timeout=600, ip, nw_src=192.168.20.10, nw_dst=192.168.100.10, actions=mod_dl_src:00:00:00:00:10:10, mod_dl_dst:00:00:00:00:20:20, output:1  
  cookie=1, duration_sec=255782s, duration_nsec=362000000s, table_id=1, priority=65535, n_packets=61977, n_bytes=6593752, idle_timeout=0, hard_timeout=0, tcp, in_port=5, tp_src=6633, actions=  
root@OpenWrt:/#
```

図 19 フローテーブル

各エントリには、下記のような意味があり、コントローラからの指示によって格納され、スイッチはエントリ通り処理を行う。

```
cookie=2087, 任意の用途に使える整数, フロー整理等を利用
priority=65535, 優先順位 (デフォルトで最優先 0xffff)
n_packets=751, ...この条件のパケットが中継された回数
n_bytes=969166, ...この条件のパケットが中継された量
idle_timeout=0, ...フローが一定時間参照されなかった場合に破棄されるまでの秒数
hard_timeout=600, ...フローが追加されてからの寿命
ip,nw_src=192.168.100.10, ...送信元 IP アドレス
nw_dst=192.168.20.10, ...宛先 IP アドレス
actions=...ここから処理に関する記述
mod_dl_src:00:00:00:00:10:40, ...送信元 MAC アドレスを書換
mod_dl_dst:00:14:85:03:1b:21, ...宛先 MAC アドレス
output:4...転送に利用するスイッチのポート
```

4.1.5. シンククライアント

シンククライアントには、通常のノート PC を使用した。通常のノート PC に VNC クライアントソフトを導入し、仮想マシンへ VNC 接続することでシンククライアントの代用とした。

4.2. 実装環境

本章では、OVCTの実装環境についてまとめる。

4.2.1. マシン

使用したマシンを下图に示す

物理マシン		
ハードウェア	台数	2台
	CPU	Core i7(3.4GHz)
	メモリ	8GB
ソフトウェア	OS	Ubuntu 12.04.3 LST 64bit
	ハイパーバイザー	qemu-kvm 1.0
仮想マシン		
ハードウェア	台数	2台
	メモリ	1GB
ソフトウェア	OS	CentOS 5.3 32bit
NFS		
ハードウェア	台数	1台
	CPU	Pentium 4(3.2GHz)
	メモリ	3.2GB
ソフトウェア	OS	CentOS 5.9 32bit
シンククライアント		
ハードウェア	台数	2台
	CPU	Core i3(2.13GHz)
	メモリ	2GB
ソフトウェア	OS	Fedora 13
	VNCソフト	vinagre 2.30.2

図 20 マシン構成



図 21 使用したマシン群

4.2.2. ネットワーク

使用したネットワーク機器を下図に示す.

OpenFlowコントローラ		
ハードウェア	台数	1台
	CPU	Pentium 4(3.2GHz)
	メモリ	3.2GB
ソフトウェア	OS	Ubuntu 12.04 LST 64bit
	フレームワーク	trema 0.4.5
	言語	ruby 1.8.7
	プロトコル	OpenFlow 1.0
OpenFlowスイッチ		
ハードウェア	台数	3台
	製品名	WHR-G301N
ソフトウェア	OS	OpenWrt
	ファームウェア	OpenFlow 1.0 for WHR-G301N
	プロトコル	OpenFlow 1.0
ルータ		
ハードウェア	台数	1台
	製品名	WZR-HP-AG300HA

図 22 ネットワーク機器構成



図 23 使用したネットワーク機器

4.3. 実装構成

本章では、OVCTの実装構成についてまとめる。

4.3.1. システム概要図

作成したシステムの概略図を下図に示す。OpenFlow コントローラ (OFC) で OVCT プログラムが動作しており、接続している OpenFlow スイッチ (OFS) を制御し、ルーティング及び通信の検知、配置場所の検討、ライブマイグレーション指示、ネットワーク設定の変更を行っている。OFS2, OFS3 にそれぞれシンククライアント (TCL) と仮想マシンが接続されている。このような状態で、シンククライアントから仮想マシンへ VNC 接続を行うことで仮想マシンの配置が変更され、シンククライアント、仮想マシン間のネットワーク経路を短縮する。

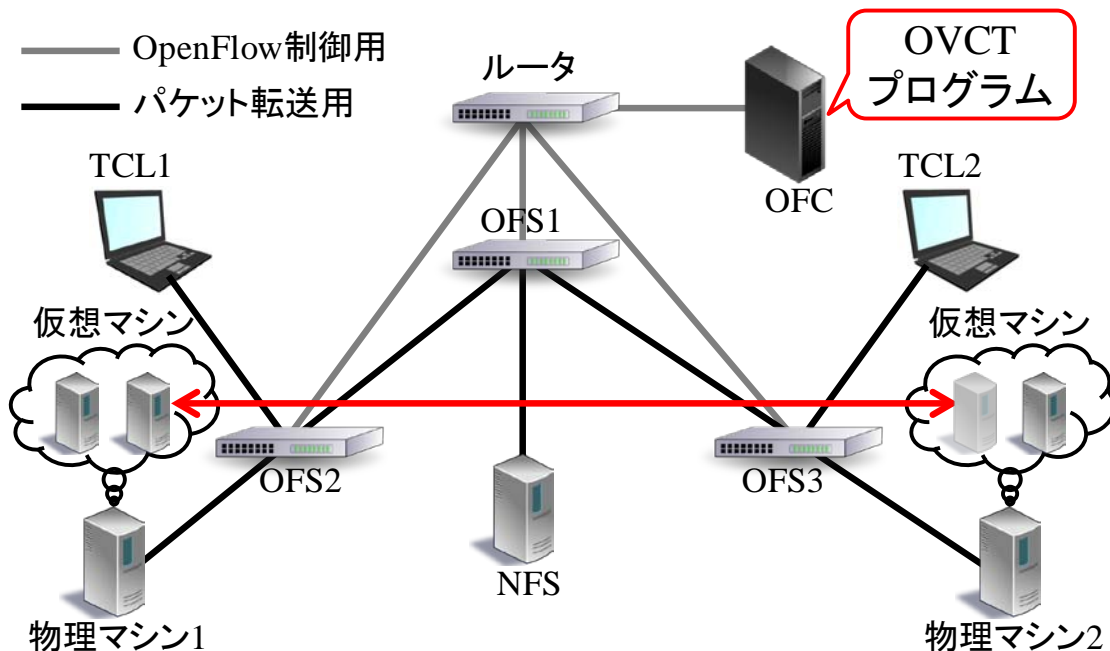


図 24 システム概略図

4.3.1. システム詳細図

作成したシステムの詳細図を下図に示す.

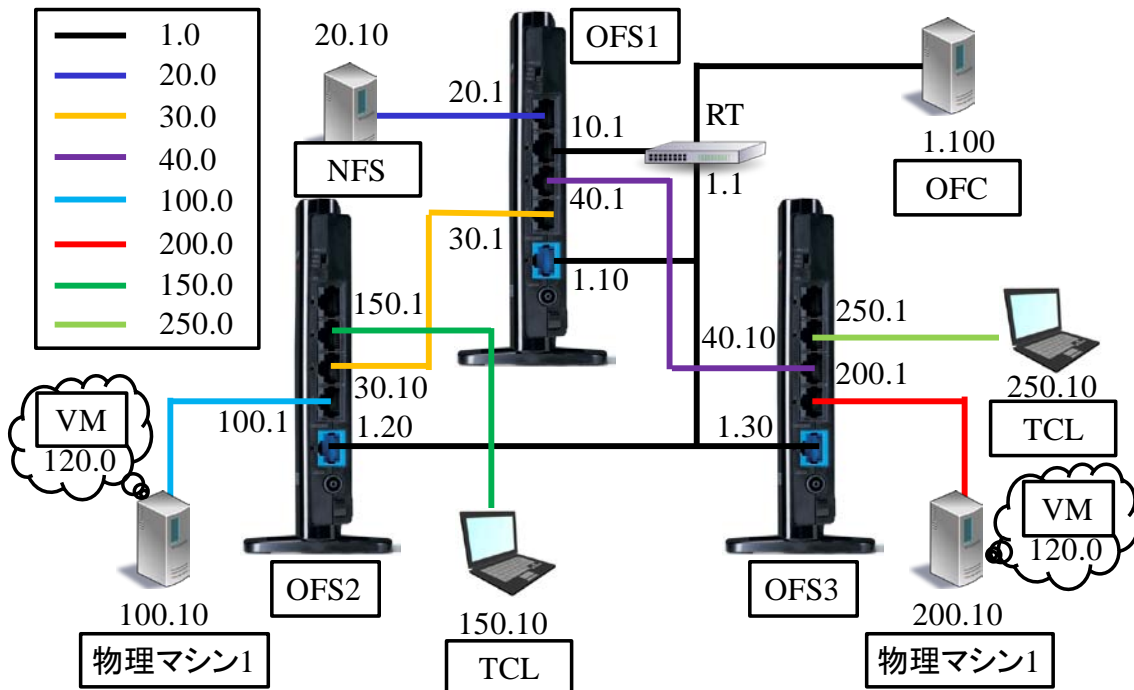


図 25 システム詳細図

5. 評価

本章では、OVCT の評価を述べる

5.1. 通信速度

5.1.1. 評価方法

本提案の目的であるシンククライアントシステムの性能向上が図れているか評価するため、シンククライアント端末、仮想マシン間の通信速度を計測した。計測には ping コマンドを使用し、下図に構成と一例を示す。下図のように、シンククライアント端末を 2 台、仮想マシンを 2 台用意し、OVCT 適用前の OpenFlow ネットワークと OVCT 適用後を比較した。また、より正確な値を取得するため、20 回分の応答時間の平均を使用した。

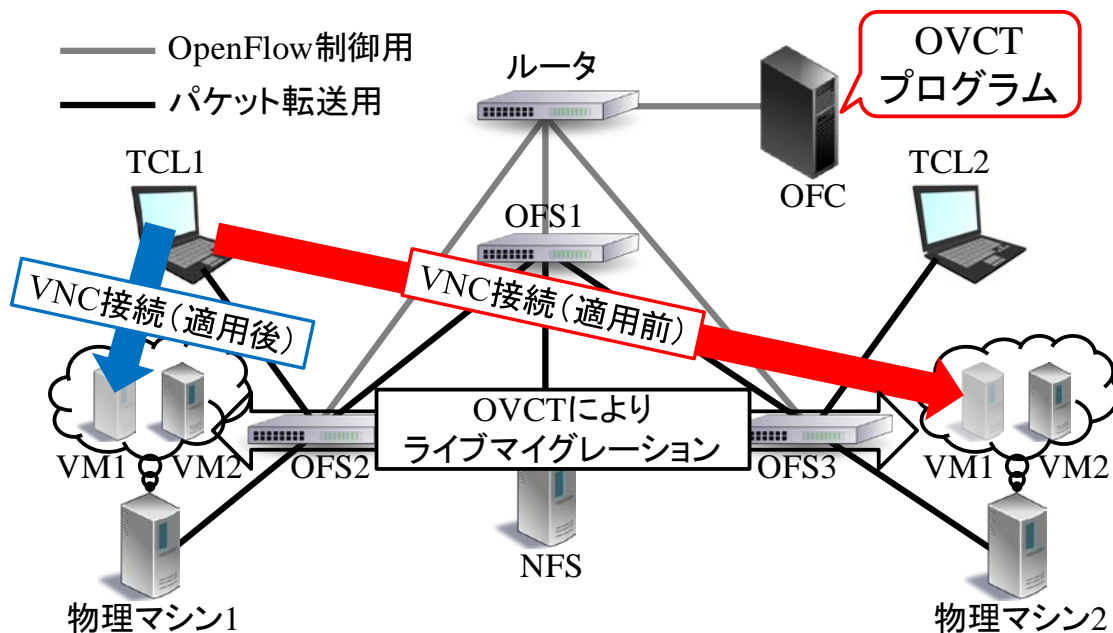


図 26 評価のためのシステム構成

5.1.1. 評価結果

下図に評価結果を示す。図の通り、平均応答速度が適用前 6.26[ms]であるのに比べ適用後 2.84[ms]と約 55%応答速度が削減されていることが分かる。理由としては、適用前に比べ、ホップ数を2削減できたためであると考えられる。以上のことより、シンクライアントの性能向上が図れたことが分かった。

(単位:ms)		宛先			
送信元		適用前		適用後	
		VM1	VM2	VM1	VM2
	TCL1	6.02	5.59	2.77	2.75
	TCL2	6.88	6.55	2.98	2.83
平均		6.45	6.07	2.88	2.79
		6.26		2.84	

図 27 ping 応答時間の平均

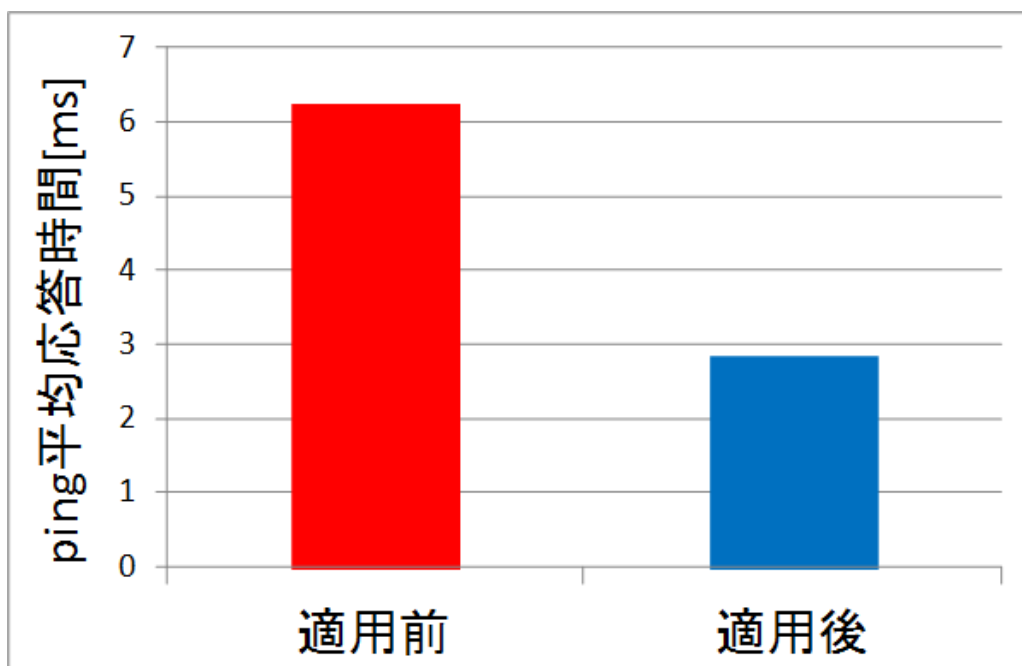


図 28 ping 応答時間のグラフ

5.2. パケット中継数・量

5.2.1. 評価方法

本提案の目的であるネットワーク負荷の軽減が図れているか評価するため、ネットワーク全体のスイッチが中継したパケット数・量を計測した。計測には OpenFlow の統計情報取得機能を使用した。構成は、5.1.1 評価方法と同様である。以上のような構成で 10 時間動作させ、時間経過による累計パケットの変化を取得した。統計情報は、本研究では評価にのみ利用したが、ネットワーク管理等を行う際に非常に有用であると言える。

5.2.2. 評価結果

下図に評価結果を示す。図の通り、累計中継パケット数は常に適用前に比べ適用後が少なくなっており削減できたことが分かる。また、10 時間駆動時には最大 32% 中継数を削減していることが分かる。累計中継パケット数は、1 時間目はライブマイグレーションによるパケットで傾きが適用前に比べ適用が大きくなっているが、その後は小さくなり 8.5 時間目で逆転している。つまり、累計中継パケット数の観点で 9 時間以上駆動させた場合に本提案が有効であることが分かる。一般的なユーザの 1 日のマシンの起動時間は少なくとも、業務時間（8 時間）＋休憩時間（1 時間）を含めた 9 時間以上であることが考えられるため、本提案は有効であると言える。以上のことより、ネットワーク全体のパケットを削減し、ネットワークの負荷を削減できたことが分かる。

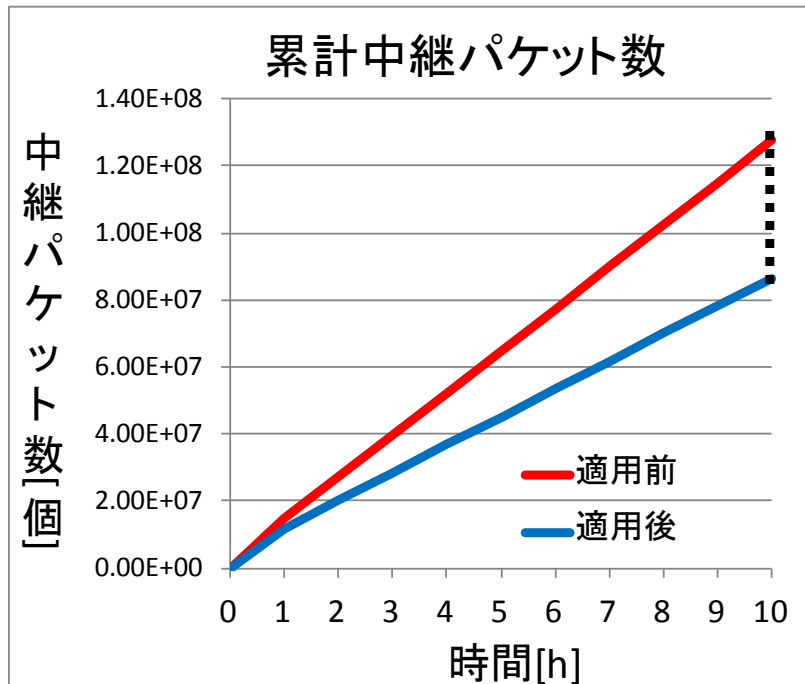


図 29 累計中継パケット数のグラフ

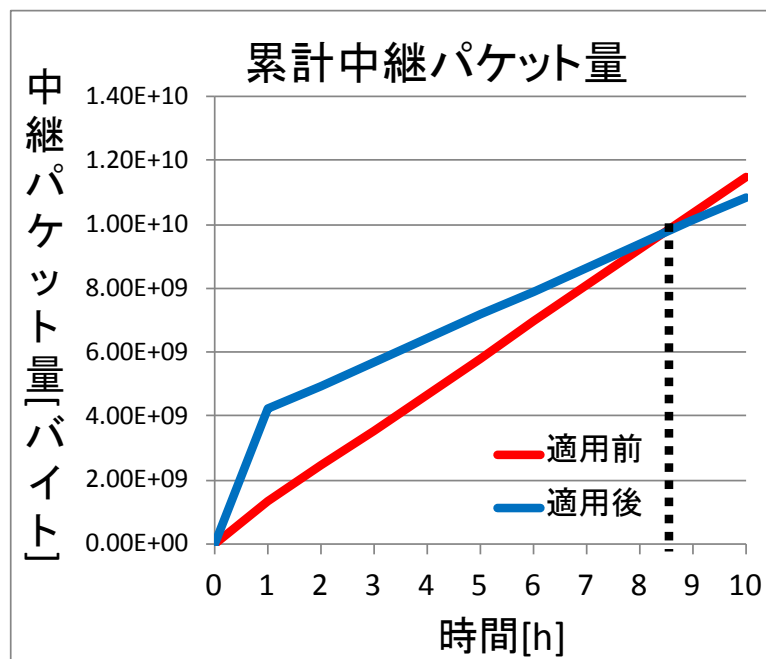


図 30 累計中継パケット量のグラフ

5.3. 処理時間

5.3.1. 評価方法

本提案が実用的であるか判断するために、各種処理に掛かった時間を計測した。測定には、プログラム内に埋め込んだ時間計測用の処理とパケットキャプチャソフトである Wireshark を使用した。構成は、5.1.1 評価方法と同様である。以上のような構成でシンククライアントから遠方に配置されている仮想マシンに VNC 接続を行い、通信の検知、配置場所の検討、ライブマイグレーション、ネットワーク設定の変更に掛かった時間を測定した。

5.3.2. 評価結果

下図に評価結果を示す。配置場所の検討では、SSH で物理マシンに接続し空きリソースの情報を取得する。その際、SSH での接続確立に時間掛かるため、全体を通して 10 秒と長くなっている。ライブマイグレーションは 210 秒と長くなっている。ライブマイグレーションが実行されている時間は仮想マシンの性能が著しく低下するため、現実的な時間ではない。しかしながら、今回実験に使用した自作 OpenFlow スイッチの性能が製品に比べ悪いことも関係していると考えられる。ダウンタイムは、15 秒と長くなっているがこの間、ポップアップなどでユーザに通知することで実用は可能ではないかと考えられる。

(単位:s)

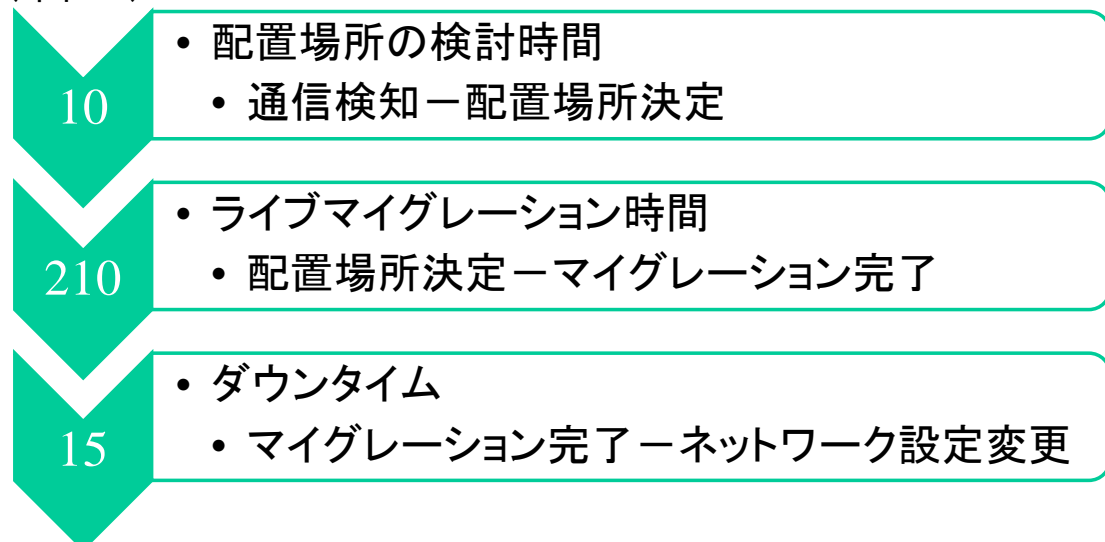


図 31 処理に掛かる時間

5.4. 結果の考察

本提案では、通信速度、パケット中継数・量、処理時間の3つの観点で評価を行った。

通信速度は、シンククライアント、仮想マシン間の経路を短縮し、ホップ数を削減することでルーティングに掛かる時間を削減し、速度を向上できた。しかし、通信速度向上のためには、ホップ数を削減するだけでなく、より性能の高いネットワーク機器を介すことやネットワークの混雑により迂回することも有効である。そのため、配置場所の検討の際に、各物理マシンからシンククライアントに通信速度計測コマンド等を実行し、配置場所の決定を行うことで、様々な状況で、シンククライアント、仮想マシン間の通信速度を向上できるのではないかと考えられる。

パケット中継数・量は、シンククライアント、仮想マシン間の経路を短縮し、中継するスイッチを削減することでネットワーク負荷の軽減を図れた。しかし、シンククライアントや仮想マシンが多数存在し、頻繁に移動するネットワークでは、シンククライアント、仮想マシン間を近隣に配置するメリットよりもライブマイグレーションに掛かるコストの方が多くなってしまうことが考えられる。そのため、本提案を適用する際には、「何時間以上利用する場合は近隣に配置する」といった閾値をマシンの台数や移動頻度から決定する必要がある。

処理時間は、どの処理も実用的な時間よりも長くなっている。しかし、これは今回実験に使用した自作 OpenFlow スwitch の性能が製品に比べ悪いことが大きく影響していると考えられ、製品である OpenFlow スwitch であれば短縮できると考えられる。また、本提案では空きリソース取得際に SSH を使用しており、この SSH をサーバ・ネットワーク統合監視ツールなどの機能を使うことで短縮できると考えられる。

6. 結言

本章では，研究のまとめと今後の課題について述べる．

6.1. まとめ

近年，クライアント側に入出力機能のみを持たせ，サーバ側でアプリケーションやファイルなどの資源を管理するシンククライアントシステムが多くの企業に導入されている．しかしながら，シンククライアントシステムでは，入出力がネットワークを介するため，シンククライアント端末とサーバの通信が多くのネットワーク機器を経由すると性能が低下してしまう．そこで，本稿では，ライブマイグレーションを使用し，シンククライアント端末や仮想マシン間の通信に応じて仮想マシンを配置するための手法，通信状況に応じた仮想マシンの最適配置手法 OVCT (Optimal placement method of Virtual machine in accordance with Communication situation) を提案した．また，提案システムの実装を行い，提案適用前のネットワークと比較実験を行った．その結果，通信速度を 55%削減，中継パケット数を最大 32%削減，9 時間以上稼働させることで中継パケット量を削減出来ることが確認でき，研究の目的であったシンククライアントシステムの性能向上とネットワーク負荷の軽減を達成した．また，本提案を情報処理学会第 76 回全国大会で発表した．

6.2. 今後の課題

6.2.1. 仮想マシンの配置場所最適化

本提案では，仮想マシンの配置場所を決定する際に，ホップ数のみを条件としている．しかし他にも通信速度，NAS や NFS との位置関係，ライブマイグレーション負荷などを考慮して決定する必要がある．

6.2.2. 実用的なネットワークへの対応

本提案では，スイッチ 3 台，シンククライアント 2 台，仮想マシン 2 台で実験を行っている．しかし，実環境ではネットワークは複雑で各マシンも数百台から数万台と幅広いため，そのような条件の際にも様々な観点で配置場所を検討する方法が必要になる．

7. 謝辞

はじめに、本研究を行うに当たり研究室に受け入れて頂いた慶應義塾大学 理工学部情報工学科 岡田 謙一 教授、並びに重野 寛 教授に感謝申し上げます。

とりわけ重野教授には、定期的にディスカッションを行って頂き、多大なご助力を頂きました。また、実装に行き詰った際にも適切なアドバイスを頂き、問題解決を行うことが出来ました。誠にありがとうございました。

続いて、ともに研究を行った同じ VN 班の鈴木 瑛識氏に感謝の意を表します。研究内容に関することはもちろんのこと、より良い研究室活動を送るためにご助力頂き、ありがとうございました。

重野研究室の皆様には、日々研究を進める上で様々な形で御支援と御協力を頂きました。又、業務時間外におきましてもスポーツや食事、懇親会など参加させて頂き、充実した研究室生活を送ることが出来ました。ありがとうございました。

担当教授を務めて頂いた（日工専）清水富門氏をはじめ情報工学科教員の方々には、研究内容や日々の生活に関する事など、様々な面で支えて頂きました。心より感謝申し上げます。

茨城管理グループの皆様には、住居の手配から日々の生活まで多くの御支援を賜り、研究に専念できる環境を用意して頂きました。心より感謝申し上げます。

（人総セ）和田 深雪氏、（Iサ本）白井 太士、様々な面で支えて頂きました。心より感謝申し上げます。

そして、（Iサ本）CL 技 G 梅澤 克之氏をはじめ、（Iサ本）CL 技 G の皆様に御礼申し上げます。皆様の御協力と御理解が無ければ、本研究はもとより（日工専）本科生における多くの貴重な経験もありませんでした。改めて、心より感謝申し上げます。

最後に、この 1 年間私を助けて下さった全ての方々に、この場を借りまして心から深く感謝申し上げます。

8. 参考文献

- [1] クライアント仮想化市場は09年1265億円、2013年までのCAGRは27.5%と高成長.
http://biz.bcnranking.jp/article/news/0911/091125_120996.html(2014 年3 月14 日現在).

- [2] 広淵崇宏,中田秀基,伊藤智,関口智嗣. "高速マイグレーションを利用した仮想マシン配置最適化システムの検討." 研究報告システムソフトウェアと オペレーティング・システム(OS) 2010.4: 1-13.

- [3] 馬場達也ほか . OpenFlow徹底入門 . 東京 , 翔泳社 , 2013 , 470p .

- [4] カオレタンマン,萱島信. "仮想デスクトップ配置アルゴリズムに関する検討."
情報処理学会研究報告. マルチメディア通信と分散処理研究会報告 2011.47 (2011): 1-8.

- [5] 高宮安仁ほか . OpenFlow実践入門 . 東京 , 技術評論社 , 2013 , 336p .

- [6] OpenFlow in theBox. <http://openflow.inthebox.info/> (2014 年3 月14 日現在).

以上