

# 経路選択を用いた広域ネットワーク通信の 遅延削減方式の提案と評価

澤 勇太 服部 直也 馬場 貴成 松原 大典

日立製作所中央研究所 〒185-8601 東京都国分寺市東恋ヶ窪一丁目 280 番地

E-mail: {yuta.sawa.eh, naoya.hattori.vn, takashige.baba.hh, daisuke.matsubara.pj}@hitachi.com

**要旨** 広域ネットワーク(WAN)には、低遅延を要求するアプリケーションや多少の遅延を許容するアプリケーション等、遅延要求の面で多種のアプリケーションのデータが流れる。しかし常に最短経路を選択する従来の WAN 通信においては、遅延を許容するデータも同一の経路を使用することが、低遅延の要求に応える際の欠点となっている。本研究ではこの欠点への対策として、MPLS-TP 等の経路指定技術を基に、遅延の異なる複数の経路からアプリケーション毎に経路を選択する方式を提案する。遅延要求に 20 ミリ秒～1 秒の幅を持たせた複数のマイクロベンチマークを用い、提案方式を評価した。その結果、最も低遅延を要求するデータの平均遅延が、経路選択を行わない場合と比べて最大 42% 低減した。

**キーワード** WAN 遅延 経路選択

## Delay Reduction Method in Wide-Area Network Communication using Path Selection

Yuta SAWA Naoya HATTORI Takashige BABA and Daisuke MATSUBARA

Central Research Laboratory, Hitachi, Ltd. 1-280 Higashi Koigakubo, Kokubunji, Tokyo, 185-8601 Japan

E-mail: {yuta.sawa.eh, naoya.hattori.vn, takashige.baba.hh, daisuke.matsubara.pj}@hitachi.com

**Abstract** In Wide-Area Network (WAN), there are many types of application data flowing, which request various delays; some application data request low-delay and the others do not. In the traditional WAN communications, all of the application data flow the same path, so that it is difficult to attend the demand for low-delay request. In this research, we propose a system which select different path for each application using routing techniques such as MPLS-TP. We evaluate the proposed system using micro-benchmarks which have different demands for delays. The demands of the benchmarks are from 20 milliseconds to 1 second. As a result, the average of the delays of applications requesting lowest delay was at most 42% decreased.

**Keyword:** WAN, delay, path selection

### 1. はじめに

広域網(Wide-Area Network, WAN)の通信データ量は増加し続けており、2014 年には 2009 年の 4 倍になると考えられている [1]。WAN を流れるデータには、個人用途のデータや社会インフラシステム[2,3]のデータ等、多くの種類がある。データの種類の違いによっては、データを送信してから受信するまでの時間(遅延)に対する要求が異なり、例えば災害・医療分野においては、低遅延通信が求められている [4,5]。

WAN における遅延は、主に経路の使用可能帯域、拠点間の伝送遅延、そして拠点間で送受信されるデータサイズによって決定されることが知られている [6,7]。その中で、同一の環境でアプリが動作する限り、経路

の伝送遅延やデータサイズは、通信時に変化することは無いが、使用可能帯域は時間とともに変化する。

使用可能帯域の変化は、主として経路共有で生じる。近年増加しているデータセンタ内の複数のアプリケーション(以下アプリ)による単一経路共有は、その典型例である。経路を共有する場合、経路の帯域も複数アプリで共有するため、使用可能帯域が減少し、遅延が増加する。結果として、低遅延を求めめるアプリに対して、低遅延を提供できないことがある。単一経路を複数アプリが共有する場合、この問題は避けられない。

一方、複数経路を共有する方法として、近年 MPLS-TP[8]や OpenFlow[9]等の経路指定技術についての研究が進んでいる。これらの経路指定技術は通過す

るルータやスイッチなどのネットワーク機器を指定することでユーザやアプリ毎に異なる経路を指定出来る。

本研究では、アプリの低遅延要求の充足に、このような経路指定技術が有用であると考えた。本研究では、アプリごとに適切な経路を指定することで、複数アプリが経路を共有する場合でも低遅延要求を満たす方式を提案し、評価する。

## 2. 従来の経路選択方式と問題点

本章では、WAN の複数経路を選択する従来方式と、従来方式が解決できない問題点を述べる。

### 2.1. 拠点間経路選択方式

WAN には多くの拠点がつながれており、拠点間で通信を行う際には WAN を提供する通信キャリアが自動的に経路を割り当てる。その際、最短以外の経路も使用することで、各経路の遅延はなるべく小さく、かつ帯域が十分に割り当たるようになっている。具体的なキャリアによる経路選択方式としては、より空いている経路に重みづけをする[10]、あるいは複数の経路の候補を列挙し、その中から帯域を有効に利用できるものを選択する[11]ことで、通信の集中を避ける方法がある。

### 2.2. 用途別回線選択方式

アプリユーザが複数の経路として、複数の回線を契約する。契約した回線を占有利用することで、それぞれ別経路としてアプリ別に使い分ける方法がある。例えば企業において比較的安く広帯域のインターネットと、高価で狭帯域だが伝送遅延が小さい専用線を用意する。緊急だがデータサイズの小さい情報は、専用線を用いて別拠点に転送する。この場合、専用線を用いているアプリについては、低遅延要求を満たすことが出来る。

### 2.3. 従来方式の問題点

データセンタ内で複数のアプリを動作させ WAN を利用する場合は、図 1 に示す形態が一般的である。アプリユーザは、それぞれアプリをデータセンタ内で動作させている。アプリユーザは、データセンタ事業者が通信キャリアと契約した、WAN 上のいくつかの経路を用いて通信する。

この形態に拠点間経路選択方式を用いる場合、通信キャリアにとっては同じデータセンタ事業者からの通信であるため、アプリを区別することが出来ない。結果として、全てのアプリが経路を共有し、低遅延要求を満たし難い。一方、用途別回線方式を用いる場合、低遅延要求するアプリが、経路を占有することが前提である。複数の低遅延要求するアプリがあった場合、通信キャリアはデータセンタ事業者に多数の占有可能な経路を提供してもらう必要があるが、アプリごとに

占有経路を用意するのは現実的ではない。

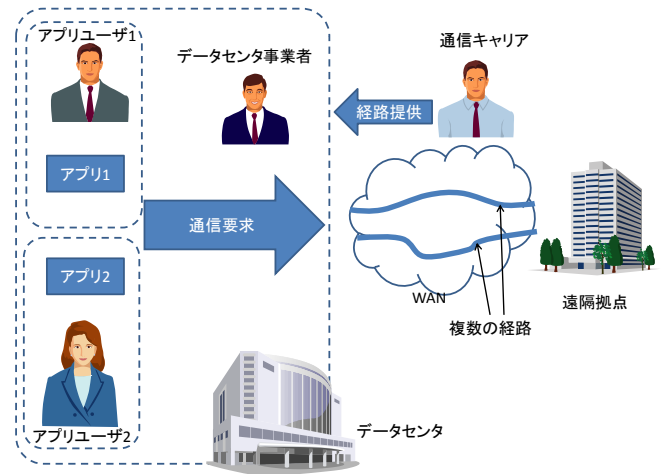


図 1 本研究の対象とする WAN 利用の形態

このように、経路の共有の非効率性が原因で低遅延要求の充足が出来ていないことが従来の問題点である。

## 3. 本研究における経路選択方式

### 3.1. 本研究の提案方式

本研究では 2 章で述べた問題点に対し、効率的な経路共有をすることでアプリの遅延要求を可能な限り満たすための方式を提案する。本方式は、以下の三つの要素から構成されている。第一に、3.2 節で説明する経路及びアプリの特性情報を対象とする。第二に、集めた特性情報から、経路を共有した場合の遅延について 3.3 節で説明するモデル化を行う。第三に、3.4 節で説明するアルゴリズムを用いて経路の最適化を行う。

### 3.2. 経路とアプリの特性

まず、経路とアプリの遅延に影響する特性について整理する。単一経路を単一のアプリが使用する場合、遅延に影響する特性として良く知られている[5,6]ように、経路については帯域(W)と伝送遅延(T)、アプリについてはデータ長(L)がある。また、本研究では各アプリがある時間以内で通信を完了することを望むと言う、遅延要求(F)があることを想定する。

次に単一の経路を複数のアプリが共有する場合を考える。この場合、アプリ同士が帯域を共有するため実際に使用可能な帯域が減少する。あるアプリが使用可能な帯域は、他のアプリが単位時間当たりのデータ送信量によって変化する。この単位時間当たりのデータ送信量は、前述したアプリのデータ長(L)に加えて、送信間隔(E)を用いることで表現できる。ちなみに、本研究中における送信間隔は、あるデータを送信開始してから、次のデータを送信開始するまでの時間のことを指す。この様子を図 2 に示す。

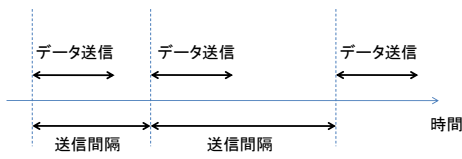


図 2 送信間隔の定義

本節では、複数の経路と複数のアプリに対して、これらのパラメータを区別するため、以下のように名前を付ける。

ある2つの拠点 A, B 間にある経路が  $n$  個、この拠点 A, B 間を用いて通信を行うアプリが  $m$  個あるとして、経路をそれぞれ  $R_1, R_2, \dots, R_n$ 、アプリをそれぞれ  $D_1, D_2, \dots, D_m$  と呼ぶ。

経路  $R_i (i=1..n)$  の帯域を  $W_i$ 、伝送遅延を  $T_i$  と呼ぶ。また、アプリ  $D_j (j=1..m)$  の平均データ長を  $L_j$ 、平均送信間隔を  $E_j$ 、そして要求遅延を  $F_j$  と呼ぶ。参考のため、図 3 に経路数  $n$  が 2 で、アプリ数  $m$  が 3 の場合を示した。

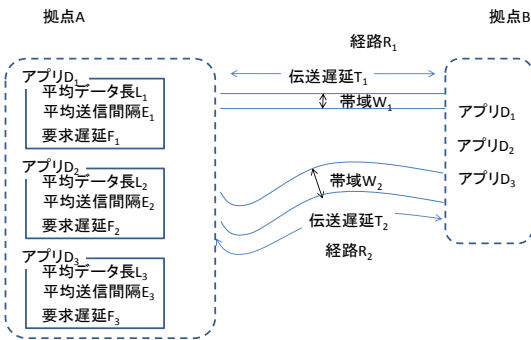


図 3 パラメータ化した WAN の構成例 (経路数  $n=2$ , アプリ数  $m=3$ )

本研究では、この状況で全てのアプリ  $A_j$  の要求遅延  $F_j$  を満たす経路の割り当てを行うことで、アプリの低遅延要求に応えることを目指す。しかしある割り当てを用いた際に、実際の遅延が要求遅延を満たすことが出来るかどうかの判定基準が無ければ、割り当ての十分性を主張できない。

従って次節では、ある経路割り当てを行った場合の各アプリの遅延をモデル化し、予測する方法について示す。

### 3.3. 遅延予測モデル

本節では、前節で定義したパラメータを用いて、アプリの遅延を予測するためのモデルを構築する。本説では、あるアプリ  $D_a$  が割り当てられている経路を、 $S(D_a)$  と表現する。

まず一つのアプリ  $D_a$  が経路  $R_b$  を占有した状態を考える。すなわち、 $S(D_a)=R_b$  かつ  $a \neq c$  ならば  $S(D_c) \neq$

$R_b$  である、この状況で、アプリ  $D_a$  の実際の遅延  $H_{a,b}$  を

$$H_{a,b} = f(T_b, L_a, W_b) = T_b + L_a / W_b \quad (1)$$

という式でモデル化した。このモデルは、データ転送にかかる遅延は経路の伝送遅延、経路の帯域、そしてアプリのデータ長によって決定されるという良く知られたモデル[5,6]によって構成されている。

次に、複数のアプリによって、経路  $R_b$  が共有されている状態を考える。この状況では、他のアプリの通信によって、使用可能帯域が減少する。経路  $R_b$  に割り当てられたアプリ  $D_a$  にとっての使用可能帯域の減少  $Q_a$  は、平均的には経路を共有しているアプリ、すなわち  $S(D_a)=S(D_k)$  となるアプリの平均使用帯域の総和で示されると考えられる。言い換えると、

$$Q_a = \sum_{1 \leq k \leq n, k \neq a, S(D_k)=S(D_a)} (L_k / E_k) \quad (2)$$

であると想定される。

これから、アプリ  $D_a$  の使用可能帯域が、 $Q_a$  だけ減少したと満たすことが出来るため、アプリ  $D_a$  の遅延  $H'_{a,b}$  を以下のようにモデル化した。

$$H'_{a,b} = f'(T_b, L_a, W_b, Q_a) = T_b + \frac{L_a}{W_b - Q_a} \quad (3)$$

このモデル化した遅延が、要求遅延と同等になると、すなわち、

$$F_a \leq H'_{a,b} \quad (4)$$

を満たすことが出来れば、アプリ  $D_a$  の遅延要求を満たせると言える。次節では、全てのアプリ  $D_a$  について、この遅延制約を満たすためのアルゴリズムを解説する。

### 3.4. 経路選択アルゴリズム

本節では、前節で構築したモデルを前提として、経路選択を行うアルゴリズムを提案する。ここで解説するアルゴリズムは、アプリや経路の特性が固定の場合に使用することが出来る。仮に、アプリや経路の特性が途中で変化する場合は再度本アルゴリズムを動作させる必要があるが、計算量としては  $O(n^2 m)$  程度である。この計算量は、例えば経路数が 10、アプリ数が 100 の場合でも数 msec 程度で計算が完了すると考えられ、たとえば数秒毎に計算してもボトルネックとはならない。

まず、遅延が大きい順に経路を並べ直す。すなわち  $T_1 > T_2 > \dots > T_n$  となるように経路の番号を付ける。その上で、アプリ  $D_1 \dots D_m$  に対して、遅延の大きいものから順に割り当てる。アプリを割り当てた場合、遅延が変化するため、各アプリの遅延制約を満たしているかどうか式(4)に示したモデルを用いて調べ、全てのア

プリの遅延制約を満たしている場合のみ割り当てを行う。このアルゴリズムを図 4 に示した。

```

for (j = 1 .. m){
  for (i = 1 .. n){
    assign(Ri, Dj) //仮にアプリ Dj を経路 Ri に割り当てる
    for (k = 1 to j){ //割り当てた場合に、遅延制約がどうなるか再調査
      if(S(Dk) = Ri && Fk <= f'(Ti, Lk, Wi, Qk})) {
        disassign(Ri, Dj) //遅延制約を満たせないアプリがあれば //割り当てをやめる
      }
    }
  }
  if (S(Dj) = Ri){
    next j //割り当てをやめていなければ、次のアプリの経路を探す
  }
}
}

```

図 4 経路選択アルゴリズム

#### 4. 評価環境

本研究では、評価のためにプロトタイプシステム及び、ベンチマークプログラムを構築した。下記にプロトタイプシステムとベンチマークプログラムの概要を示したのちに、本研究で用いたベンチマークのパラメータについて示す。

##### 4.1. プロトタイプシステム概要

拠点 A から拠点 B に対して、データを送信するモデルを再現するための実験環境を、図 5 のように用意した。本実験の構成に置いては、アプリの通信データを生成するサーバ(送信サーバ)、アプリの通信データに対して経路を割り振るサーバ(経路振り分けサーバ)、アプリの通信データを受け取るサーバ(受信サーバ)の 3 種類のサーバマシンを使用する。各サーバは、Ethernet とネットワークスイッチによってつながれている。各サーバマシン及びネットワークスイッチのスペックについては、表 1 に示した。

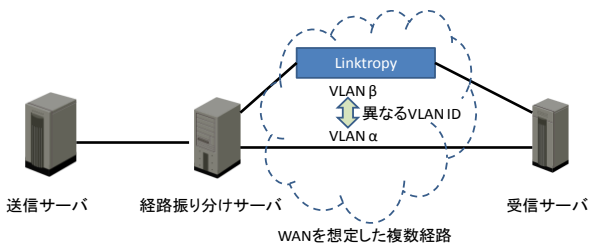


図 5 実験構成図

表 1 実験環境(サーバ及びスイッチのスペック)

CPU	Intel Pentium® Dual E2180 2.0G Hz
Memory	Samsung DDR2 667MHz 2G Byte
OS	CentOS 5.2 (Linux Kernel 2.6.18-92.el5)
NIC	Intel® 82572 EI Gigabit Ethernet Controller
スイッチ	Alaxala® AX2400S

経路振り分けサーバと受信サーバの間には、VLAN を用いて区切られた二つの経路があり、二つの経路は異なる VLAN に所属している。各経路において、経路振り分けサーバと、受信サーバはそれぞれ異なる IP アドレスを持っているため、IP アドレスを指定することが経路指定を行うことと等価である。経路の片方は Apposite Technology 社の Linktropy® を介してつながれている。Linktropy® は、内部にバッファメモリを持つことにより、遅延を発生させる機器である。また、各経路の帯域を制御するために経路振り分けサーバ内に PSPacer[12]を用いた。PSPacer は、VLAN 毎の帯域を厳密に制御することを可能にするソフトウェアである。

送信サーバ - 経路振り分けサーバ間及び、経路振り分けサーバ - 受信サーバ間においては、それぞれ UDP/IP 上に構築した独自プロトコル(データ管理 I/F プロトコル)を用いて通信を行っている。今後の WAN 環境ではより安定した通信を行えることを想定し、輻輳制御やスロースタート等の、TCP ウィンドウ制御による帯域制限の影響を排除するためにこのプロトコルを設計した。またデータ管理 I/F プロトコルは、各データパケットがどのアプリの通信データに所属するかを判定するためのフィールドを持っている。

本実験の構成では、送信サーバ、経路振り分けサーバ、受信サーバはそれぞれ 1 台だけ用意した。しかし、経路振り分けサーバ及び受信サーバを複数用意した場合でも、経路選択アルゴリズムの結果を共有していれば並列化可能である。従って、多数のアプリを動作させる際にも、振り分けサーバ・受信サーバのマシンスペックはボトルネックとならないと考えられる。

##### 4.2. ベンチマーク環境と実験パラメータ

前記したプロトタイプシステム上に、ベンチマークプログラムを構築し、実験を行った。

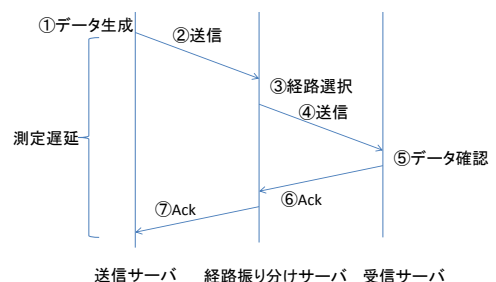


図 6 ベンチマークのシーケンス

図 6 を用いて本研究におけるベンチマークのシーケンスについて解説を行う。本研究上では、①送信サーバがデータを生成し、②経路振り分けサーバに対してデータを送信する。③経路振り分けサーバは、送信サーバから受け取ったデータに基づいて経路を選択し、

④同じデータを受信サーバに送信する。⑤受信サーバは、経路振り分けサーバからデータを受信したことを確認すると、⑥経路振り分けサーバに Ack を返す。⑦ Ack を受け取った経路振り分けサーバは、同様に送信サーバに Ack を返す。

次に、送信サーバがデータ生成をするタイミングについて記す。各アプリは 3.2 節で述べた平均送信間隔というパラメータを持っており、送信サーバは送信間隔の平均が平均送信間隔と同じになるように調整しながら、経路振り分けサーバにデータを送信する。送信サーバは送信間隔が確率分布である指数分布に基づくようにする。

このような確率分布に従った送信間隔にした理由は、周期的なデータ送信を行うと、アプリが経路を共有することによる帯域の減少が、偶然観察されないことがありえてしまうためである。

本研究の実験で用いたアプリを想定したベンチマークの特性を表 2 に示す。本研究では、表 2 に示された 5 種類のアプリを同時に動作させ、各アプリのデータ送信毎の遅延を測定した。

表 2 使用アプリの特性

アプリ名	平均データ長	平均送信間隔	要求遅延
アプリ 1	1 Mbyte	200 msec	1000 msec
アプリ 2	100 kbyte	100 msec	500 msec
アプリ 3	100 kbyte	100 msec	20 msec
アプリ 4	10 kbyte	100 msec	100 msec
アプリ 5	10 kbyte	100 msec	20 msec

本研究の実験で用いた 2 つの経路の経路特性を表 3 に示す。経路 1 及び経路 2 は、それぞれ 100 Mbps の帯域を持つ。伝送遅延は経路 1 が 100 msec なのに対し、経路 2 は 2 msec となるように設定した。

表 3 使用経路の特性

経路名	帯域	伝送遅延
経路 1	100 Mbps	100 msec
経路 2	100 Mbps	2 msec

## 5. 評価・考察

2 章で示した提案方式中の経路選択アルゴリズムを用い、ベンチマーク対象であるアプリ毎に経路を選択させたところ、アプリ毎の経路は表 4 のようになった。また比較対象である従来方式として、表 5 に示したように全ての通信を経路 2 に割り当てるものを用いた。表 4、表 5 には本研究で作成したモデルを用いて平均遅延を予測した値も併記した。

表 4 提案方式による経路割り当て

アプリ名	選択経路	平均遅延予測値
アプリ 1	経路 1	200 msec
アプリ 2	経路 1	115 msec
アプリ 3	経路 2	10 msec
アプリ 4	経路 1	100 msec
アプリ 5	経路 2	2 msec

表 5 従来方式における経路割り当て

アプリ名	選択経路	平均遅延予測値
アプリ 1	経路 2	100 msec
アプリ 2	経路 2	18 msec
アプリ 3	経路 2	18 msec
アプリ 4	経路 2	4 msec
アプリ 5	経路 2	4 msec

実験結果として、以下では本研究の目的である、低遅延を要求するアプリについて示す。5 つのアプリの中から相対的に低遅延(20 msec)を要求しているアプリである、アプリ 3 及びアプリ 5 の遅延時間の分布を図 7、図 8 に示す。また、表 6 に、それぞれの方式での平均遅延と提案方式による平均遅延の低減率について記載した。

図 7 及び図 8 においては、横軸に 5 msec 単位での測定遅延を、縦軸にその測定遅延の範囲で通信できたデータの頻度を示す。

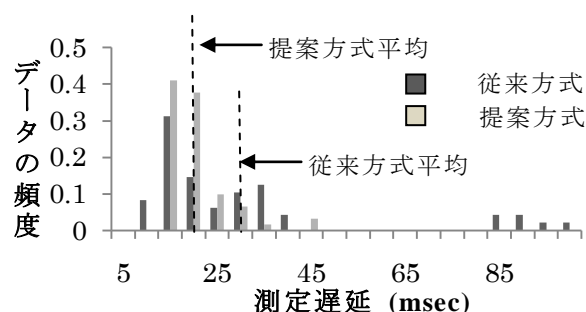


図 7 アプリ 3 の測定遅延分布

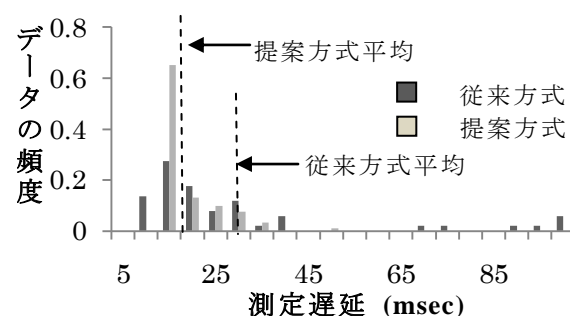


図 8 アプリ 5 の測定遅延分布

表 6 方式毎の平均遅延

		平均遅延	平均遅延低減率
アプリ 3	従来方式	27.4 msec	36%
	提案方式	17.7 msec	
アプリ 5	従来方式	27.5 msec	42%
	提案方式	15.9 msec	

本研究の提案方式を用いた場合、アプリ 3 及びアプリ 5 のみが経路 2 を使用することが良いと判断した。その判断に従った場合、アプリ 3 及びアプリ 5 の平均通信遅延が減少し、元々の要求遅延であった 20 msec を満たしている。

## 6. おわりに

本研究では、複数経路のある WAN 上で、アプリによる低遅延要求に応えることを目的とした経路選択方式を提案し、プロトタイプを実装した。本経路選択方式では、アプリ及び経路の特性を用いることで遅延をモデル化し、そのモデル上で遅延要求に応えるような経路割り当てを行った。その結果、提案方式による経路選択を行うことで、経路選択を行わない場合と比べて遅延が最大で 42% 低減し、結果として低遅延要求を満たしていることを確認した。

今後の課題として、データ管理 I/F プロトコルをより高速化させることと、実環境に近い環境での評価が挙げられる。

また表 4 及び表 5 に記載した平均遅延の予測値と、表 6 に示した方式毎の平均遅延の比較を行うと、経路選択を行う・行わないにかかわらず、平均遅延が予測値よりも大きくなっていることがわかった。これは、データ管理 I/F プロトコル自体が UDP 上に構築されているため、スロースタートや輻輳制御の速度低下が発生しない一方で、NIC の機能を有効に使いきれていないことが原因と考えられる。これは今後のより広帯域ネットワークにおいては処理のボトルネックとなりうる。従って、TCP 上に同様のプロトコルを構築した上でウィンドウ制御を調整する、あるいは UDP 上のプロトコルであってもオフロードエンジンをもった NIC を作成するなどの解決策が考えられる。

実環境に近いベンチマークに関しては、今回のベンチマークではアプリの増減が存在しなかった。時間とともに使用するアプリが増加/減少することが考えられるため、その場合に対応するベンチマークを検討している。

## 謝辞

本研究は、独立行政法人情報通信研究機構の委託研究「ネットワーク仮想化を活用したデータサービスアプリケーション基盤技術に関する研究開発」の研究成果である。

## 文 献

- [1] Cisco, Cisco Visual Networking Index: Forecast and Methodology 2009-2014, June 2010, [http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white\\_paper\\_c11-481360.pdf](http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf)
- [2] 内閣府、平成 21 年度防災白書第 1 部第 2 章 3-1(9) 総合防災情報システムの整備 [http://www.bousai.go.jp/hakusho/h21/bousai2009/html/honbun/1b\\_2s\\_3\\_01\\_9.htm](http://www.bousai.go.jp/hakusho/h21/bousai2009/html/honbun/1b_2s_3_01_9.htm)
- [3] 総務省、「医療分野における ICT の利活用に関する検討会」報告書、平成 18 年 4 月 18 日
- [4] ITU-T Recommendation G.114, “One-way transmission time”, May 2003, <http://itu-int/rec/T-REC-G.114-200305-I/en>
- [5] 文部科学省 高度即時的自身情報伝達網実用化プロジェクト, 平成 19 年度成果報告書, [http://www.bosai.go.jp/kenkyu/sokuji/pages/results\\_h19.htm](http://www.bosai.go.jp/kenkyu/sokuji/pages/results_h19.htm)
- [6] Leonard Kleinrock, Gigabit networks have forced us to deal with the propagation delay due to the finite speed of light, IEEE Communications Magazine, April 1992
- [7] Tom Sheldon, Encyclopedia of Networking & Telecommunications, Delay, Latency and Jitter
- [8] IETF, RFC 5317 Joint Working Team (JWT) Report on MPLS Architectural considerations for a Transport Profile, Feb. 2009
- [9] OpenFlow Consortium, The OpenFlow Switch Consortium, <http://www.openflowswitch.org/>
- [10] Bernard Fonts et. al., Internet traffic engineering by optimizing OSPF weights, Proc. IEEE INFOCOM, vol2, pp.519-528, Mar.2000
- [11] 藪崎仁史, 松原大典, 時間軸を考慮した経路計算アルゴリズムの検討, 新世代ネットワークワークショップ 2009
- [12] Ryosei Takano et al., Efficient Packet Pacing for MPI Programs in a Grid Environment, Cluster2007, 2007