社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

インラインネットワーク計測手法 ImTCP およびその応用手法の実装および性能評価

津川　知朗†　　長谷川　剛††　　村田　正幸†

† 大阪大学大学院情報科学研究科　〒 565–0871 吹田市山田丘 1–5
†† 大阪大学サイバーメディアセンター　〒 560–0043 豊中市待兼山町 1–32
E-mail: †{t-tugawa,murata}@ist.osaka-u.ac.jp, ††hasegawa@cmc.osaka-u.ac.jp

あらまし　我々の研究グループでは，これまでに利用可能帯域を高い精度で継続的に計測したインラインネットワーク計測手法である ImTCP，およびその計測結果を利用するバックグラウンド転送方式である ImTCP-bg の提案を行い，コンピュータ上のシミュレーションによる評価によってその有効性を示した．しかしながら，ネットワーク計測およびその応用手法の評価を行う際には，実ネットワーク上での実装実験が必要不可欠である．本稿では，これらの提案手法の実装を行い，実験ネットワークを用いた実装実験を行う．実装実験を通じて，実ネットワーク上においても ImTCP がシミュレーションによる評価と同様に高い精度で計測を行うことができること，および ImTCP-bg が他のトラヒックに影響を与えずにバックグラウンド転送を行うことができることを示す
キーワード　バックグラウンド転送，インライン計測，輻輳制御，利用可能帯域，TCP

Implementation and evaluation of an inline network measurement algorithm and its application technique

Tomoaki TSUGAWA†, Go HASEGAWA††, and Masayuki MURATA†

† Graduate School of Information Science and Technology, Osaka University　Yamadaoka 1–5, Suita-shi, Osaka 565–0871 Japan
†† Cyber Media Center, Osaka University　Machikaneyama 1–32, Toyonaka-shi, Osaka 560–0043 Japan
E-mail: †{t-tugawa,murata}@ist.osaka-u.ac.jp, ††hasegawa@cmc.osaka-u.ac.jp

Abstract　In our previous studies, we proposed ImTCP, a novel inline network measurement technique which can obtain available bandwidth information continuously, and ImTCP-bg, a new background TCP data transfer mechanism by using measurement results. We investigated the effectiveness of ImTCP and ImTCP-bg through simulation experiments. However, it is important to test the measurement-related mechanisms in actual networks when we evaluate their effectiveness. In this paper, we implement the proposed mechanisms and evaluate them in the experiment network. We investigate the performance through the experiments, and validate the effectiveness of measurement accuracy and interference degree with the other traffic.
Key words　background data transfer, inline network measurement, congestion control, available bandwidth, TCP

1. はじめに

近年のネットワーク速度の飛躍的な向上やインターネット利用者数の爆発的な増加にともなってインターネットが急速に発展していくにつれ，提供されるネットワークサービスも多種多様のものとなってきている．例えば，コンテンツ配信を目的とした Contents Delivery Network（CDN）[1, 2]，ピア同士の直接的な通信を実現する P2P ネットワーク [3, 4]，ネットワーク上で分散計算環境を提供するグリッドネットワーク [5, 6]，IP ネットワーク上に仮想網を構築する IP-VPN [7] などのサービスオーバレイネットワークが挙げられる．これらのネットワークサービスの品質を向上させるためには，ネットワークの基盤となる IP ネットワークの資源状況を把握し，有効に利用することが重要となってくる．特に，ネットワークリンクの帯域に関する情報を得ることによって，様々なネットワークサービスの品質を向上させることができると考えられる．

そこで我々の研究グループでは，これまでエンドホスト間の利用可能帯域を計測するための新たなインラインネットワーク計測手法 ImTCP [8] を提案している．利用可能帯域を計測するための手法はこれまでにも多く提案されているが [9–11]，それ

らの手法の多くは，計測を行う際に多くの計測用パケットを必要とするためにネットワークへ大きな影響を与える，計測に長い時間がかかるなどの問題が存在する．一方，ImTCP は TCP コネクションがデータ転送に用いるデータパケットとそれに対する ACK パケットのみを用いてネットワークの利用可能帯域を計測するため，計測用パケットを必要とすることなく高い精度のアクティブ計測を可能としている．また，非常に短い周期（1–4 RTT）で継続的に計測結果を取得することができるため，ネットワーク状況の変化に素早く追随することができる．

また我々は，この計測結果に基づいた応用手法として，バックグラウンド転送方式 ImTCP-bg [12] を提案している．バックグラウンド転送とは，他のトラヒックに影響を与えずにネットワークの空いている帯域のみを利用して行うデータ転送のことである．バックグラウンド転送が実現されることによって，品質を向上させることのできるネットワークサービスが存在する．例えば，前述した CDN ではユーザからのコンテンツ閲覧要求を受けて行われるデータ転送以外にもバックアップ，キャッシング [13]，プリフェッチ [14, 15] などによって発生するデータ転送を行っている．このとき，バックアップ等のデータ転送をバックグラウンド転送によって行うことによって，バックアップ等のデータ転送中にもユーザからのコンテンツ要求に迅速に対応することが可能となる．ImTCP-bg は，従来まで提案されてきたラウンドトリップ時間を指標として早期にネットワーク輻輳を検知する方式 [16, 17] とは異なり，輻輳を発生させることなく空き帯域を効率的に利用したバックグラウンド転送を行うことができる．

[8, 12] においては，提案手法の有効性を ns-2 [18] を用いたコンピュータ上のシミュレーションによって評価している．その結果, [8] においては，提案した計測手法が少ない数のパケットで高い精度の計測結果を継続的に導出することができることを確認しており, [12] においては，ImTCP の計測結果に基づいたバックグラウンド転送方式が，優先されるべきトラヒックにほとんど影響を与えずにネットワークの空き帯域のみを用いてデータ転送を行うことができることを確認している．しかしながら，ネットワーク計測手法の有効性の評価には，実コンピュータおよび実ネットワークを用いた実装実験が必要不可欠である．

そこで本稿においては, [8] において提案されたインラインネットワーク計測手法 ImTCP および [12] において提案された計測結果に基づいたバックグラウンド転送方式 ImTCP-bg の実装を行い，実験ネットワークを用いた実装実験を行うことにより，これらの提案手法の実ネットワーク上での有効性を評価する．本稿では，ImTCP および ImTCP-bg を FreeBSD 4.10 [19] のカーネルシステムに実装する．

以下，2 章ではインライン計測手法 ImTCP および計測結果に基づくバックグラウンド転送方式 ImTCP-bg のアルゴリズムの説明を行う．3 章では，これらの提案手法を FreeBSD 4.10 のカーネルシステムに実装する際の実装指針を示す．また，その時に発生する問題点を挙げ，その解決方法に対して議論を行う．4 章では，実験ネットワークを用いてこれらの提案方式の評価を行う．最後に，5 章で本稿のまとめと今後の課題を示す．

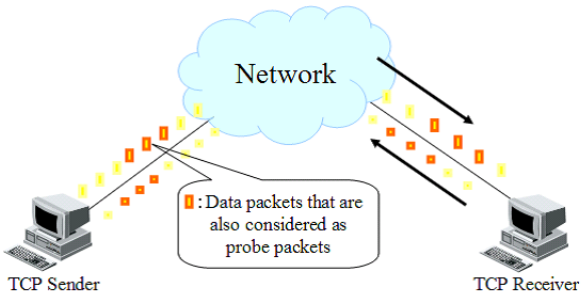

図 1　ImTCP によるインラインネットワーク計測

2. インライン計測手法 ImTCP およびその応用手法 ImTCP-bg のアルゴリズム

本章では，インラインネットワーク計測手法 ImTCP およびその応用手法 ImTCP-bg のアルゴリズムについて，その概要の説明を行う．

2.1 ImTCP アルゴリズム [8]

ImTCP は，送受信エンドホスト間のネットワークパスにおける現在の利用可能帯域を計測する．TCP によるデータ転送においては，送信側が受信側にパケットを送信し，受信側が ACK パケットを返送する．この性質を利用することにより，図 1 に示すように，送信側で設定したデータパケットの送信間隔に対して，その ACK パケットの到着間隔の変化を観察することによって利用可能帯域の計測を行う．

利用可能帯域を計測する際には，現在の利用可能帯域値が含まれていると考えられる帯域の上限と下限を過去の計測結果を利用して設定し，この区間の中から利用可能帯域を探索する（この区間を探索区間と呼ぶ）．探索区間を設定することで，不必要に高いレートでパケットを送出することが避けられるため，ネットワークに与える影響を最小限に抑えることができる．また，探索する帯域が狭くなるため，計測の精度を保ちながら用いるパケット数を減少させることができる．探索区間は過去の計測結果を基に設定するため，ネットワーク状況の変化に伴い利用可能帯域が急激に変化した場合，探索区間内に利用可能帯域が存在しない場合が存在する．ImTCP では，そのような場合においても，数回の計測で新たな利用可能帯域を発見することができる．ImTCP の動作概略を以下に示す．それぞれのステップにおける詳細なアルゴリズムについては，[20] を参照されたい．

（1） Cprobe アルゴリズム [21] に基づいて，初期探索区間を決定する

（2） 探索区間を複数の小区間に分別する

（3） 各小区間に対応する計測ストリームを計測アルゴリズムに基づいて決定されたタイミングで送信し，送信間隔とそれらの ACK パケットの受信間隔の比較を行い，パケット間隔が増加したかどうかを判断する

（4） パケットの送受信結果から，利用可能帯域が含まれていると考えられる小区間を選択する

（5） 選択した小区間に対応する計測ストリームの送受信結果から，利用可能帯域を導出する

（6） 過去の計測結果から 95% 信頼区間を導出する．その区間を次回の探索区間として設定し，(2) へ戻る．

2.2 ImTCP-bg アルゴリズム [12]

ImTCP-bg は，インラインネットワーク計測によって利用可能帯域を計測する ImTCP を基に，計測結果を輻輳ウィンドウサイズの制御に用いることによって，バックグラウンド転送を行う TCP を実現したものである．具体的には，計測結果を基に送信ホ側の TCP が持つ輻輳ウィンドウサイズの上限値を動的に設定する．

まず，インライン計測によって取得されたネットワークの利用可能帯域に対して，平滑化を行う．A_{cur} を最新のインライン計測による計測結果，$\bar{A}$ を平滑化が行われた計測結果とすると，$\bar{A}$ は平滑化パラメータ γ を用いて次のような指数移動平均の式を用いて求める．

$$\bar{A} \leftarrow (1-\gamma) \times \bar{A} + \gamma \times A_{cur} \tag{1}$$

次に，送信側 TCP の最大輻輳ウィンドウサイズ $maxcwnd$ を次式によって決定する．

$$maxcwnd \leftarrow \bar{A} \times RTT_{min} \tag{2}$$

ここで，RTT_{min} はこれまでに観測された RTT の最小値を表す．

ImTCP はそのアルゴリズムの性質上，利用可能帯域の計測を行うことができない場合や計測結果が不正確になる場合が存在する．そのような場合に，その計測結果を輻輳ウィンドウサイズの制御に用いると輻輳ウィンドウサイズが理想的な値よりも大きくなる．そして，その輻輳ウィンドウサイズのままデータ転送を続けるとネットワークで輻輳が発生し，優先されるべきトラヒックにも影響を与える．そこで，輻輳が発生している場合には，最大輻輳ウィンドウサイズは変化させずに輻輳ウィンドウサイズを強制的に減少させることによって輻輳回避を行う．輻輳が発生しているかどうかの判断は，現在の RTT の値およびこれまでに観測された RTT の最小値の比を用いて行う．現在の RTT の値を RTT_{cur}，これまでに観測された RTT の最小値を RTT_{min} とし，次式が真となる場合に輻輳が発生していると判断する．

$$\frac{RTT_{cur}}{RTT_{min}} > \delta \tag{3}$$

ここで，δ はしきい値を表し 1.0 以上の値である．(3) 式が真となり輻輳が発生していると判断されると，現在の輻輳ウィンドウサイズ $cwnd$ を次式を用いて減少させる．

$$cwnd \leftarrow cwnd \times \frac{RTT_{min}}{RTT_{cur}} \tag{4}$$

3. ImTCP および ImTCP-bg の実装

本章では，ImTCP および ImTCP-bg を FreeBSD 4.10 のカーネルシステムに実装する際の実装指針と，その時に発生する問題点について述べる．

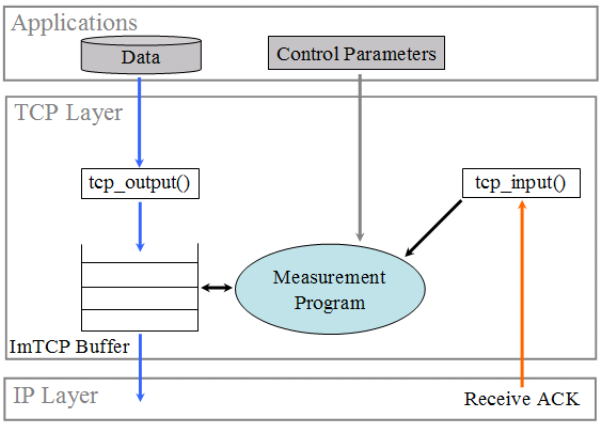

図 2 ImTCP の構成図

3.1 実装指針

図 2 に送信端末のカーネルシステムに実装する ImTCP の構成を示す．通常，アプリケーションから Socket インターフェースを通じて渡されたデータパケットは関数 tcp_output() によって TCP のプロトコル処理を受けた後，関数 ip_output() によって IP のプロトコル処理を受け，ネットワークへ送出される．ImTCP は TCP コネクションの輻輳ウィンドウサイズなどを制御パラメータとして用いるため，TCP 層の下部に実装される必要がある．そのため，計測プログラムは tcp_output() 内に実装する．具体的には，カーネルシステムによって確保されたメモリを用いて ImTCP バッファを作成し，TCP のプロトコル処理が終了したパケットを関数 ip_output() に渡す前に ImTCP バッファへ格納する．格納されたパケットは計測アルゴリズムに基づいたタイミングで 関数 ip_output() へ渡される．このとき，計測プログラムはデータパケットの送信時刻を記録する．

計測プログラムは受信した ACK パケットの処理を行う関数 tcp_input() 内にも必要となる．すなわち，受信端末から返送された ACK パケットは関数 ip_input() によって IP のプロトコル処理を受け，関数 tcp_input() へ渡される．計測プログラムは，ACK パケットの受信時刻を記録し，データパケットの送信時刻と ACK パケットの受信時刻を用いて計測アルゴリズムに基づいて利用可能帯域の推測を行う．

通常，輻輳ウィンドウサイズは関数 ip_input() によって IP プロトコル処理を受けた ACK パケットを関数 tcp_input() に渡し，TCP プロトコル処理を行うときに更新される．そのため，ImTCP-bg の輻輳制御プログラムは tcp_input() 内に実装される必要がある．輻輳制御プログラムは，tcp_input() で輻輳ウィンドウサイズが更新される際に，ImTCP-bg のアルゴリズムに基づいて輻輳ウィンドウサイズおよび最大輻輳ウィンドウサイズを決定する．

3.2 実装における問題点

ImTCP の計測アルゴリズムは，送信するデータパケットの送信間隔を調整し，そのデータパケットに対する ACK パケットの到着間隔を観察することにより利用可能帯域を推測する．そのため，3.1 で述べたように，データパケットは一旦 ImTCP

表 1　実験に使用した PC のスペック

CPU	Intel Pentium 4 3.0GHz
Memory	1,024MB
OS	FreeBSD 4.10

表 2　カーネルプログラムのコンパイルに要した処理時間

HZ	処理時間 [sec]
100	168.20
1,000	170.09
10,000	183.38
20,000	199.78
50,000	277.84
100,000	734.10

バッファへ格納された後，計測アルゴリズムに基づいたタイミングで関数 ip_input() へ渡される．この時，指定されたタイミングでパケットを関数 ip_input() へ渡す機能はカーネルシステムのスケジューリング機構を用いて実現する．しかしながら，カーネルシステムで用いられるタイマ粒度は，一般的にアプリケーションが用いるタイマ粒度よりも粗い [22] ため，計測精度が低下することが考えられる．

FreeBSD システムにおいては，カーネルシステムで用いられるタイマ粒度は HZ と呼ばれるパラメータによって決定される．通常，HZ の値として 100 が用いられており，このときカーネルシステムのタイマ粒度は 10msec となる．しかしながら，この粒度では一般的なパケット（パケットサイズ 1500Byte）を用いた場合，1.2Mbps までの帯域しか計測することができない．さらに，最大計測可能帯域に近づくほど計測粒度は粗くなる（$HZ = 100$ の場合，計測可能帯域は，1.2Mbps，0.6Mbps，0.4Mbps，... の順）．したがって，広帯域ネットワークで利用可能帯域の計測を行うためには HZ の値を大きくする必要がある（例えば，$HZ = 100,000$ の場合では最大計測可能粒度は 1.2Gbps となる）．しかしながら，HZ の値を大きくするとタスク切り替えが頻繁に発生するようになり，その処理のオーバヘッドが原因でカーネルの実行速度に影響を与えるようになる．例えば，表 1 のスペックを持つ PC を用いて，カーネルプログラムのコンパイルを行った場合の，HZ パラメータの値とコンパイルに要した処理時間の関係を表 2 に示す．この表から，HZ の値が大きくなるにつれて処理時間が飛躍的に長くなることが分かる．そのため，HZ の値を決定する際には，これらのことを考慮して決定する必要がある．

4. 性能評価

本章では，3 章に示した方針に基づいて実装した ImTCP およびその応用手法である ImTCP-bg を用いて実験用ネットワーク環境下で実装実験を行い，その実験結果をもとに実ネットワーク上での計測アルゴリズムおよびその計測結果に基づくバックグラウンド転送方式の有効性を評価する．実験ネットワーク環境を図 3 に示す．実験ネットワーク環境は，100Mbps のイーサネットによって構築され，DUMMYNET がインストールされた PC ルータを介してクロストラヒックを発生させるエンドホスト（Traffic generator），利用可能帯域の計測および計測結果を利用するバックグラウンド転送を行う送信ホスト（Sender），それぞれのホストからのパケットを受信する受信ホスト（Receiver）が接続されている．実験ネットワーク環境を構築する PC のスペックを表 3 に示す．送信ホスト（Sender）の HZ パラメータの値は，20,000 に設定する．また，DUMMYNET を用いて遅延時間を発生させ，RTT の最小値を 30msec に設定している．

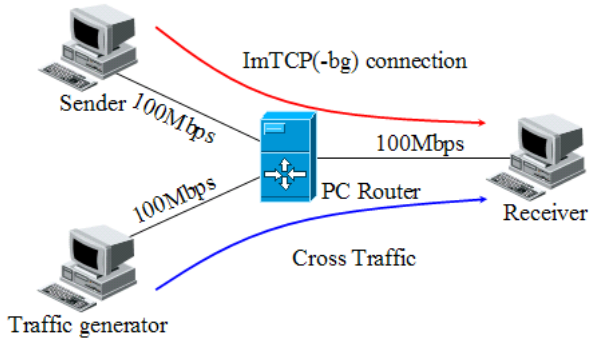

図 3　実験ネットワーク環境

表 4　CPU へかかる負荷の比較

	ImTCP-bg	TCP Reno
平均負荷 [%]	18.62	19.12

4.1 ImTCP の計測精度の評価

本節では，ImTCP の計測精度の評価を行う．評価は，クロストラヒックとして UDP トラヒックと TCP トラヒックを発生させ，それぞれの場合について ImTCP の計測精度を観察することによって行う．

4.1.1 クロストラヒックが UDP トラヒックの場合

まず，クロストラヒックとして UDP トラヒックが発生している場合について，ImTCP の計測精度の評価を行う．図 4 は，発生させるクロストラヒックの量を時間の経過とともに変動させることにより，ボトルネックリンクの実際の利用可能帯域が 0sec から 20sec までは 70Mbps，20sec から 40sec までは 30Mbps，40sec から 60sec までは 40Mbps と変化したときの計測結果，平滑化された計測結果および実際の利用可能帯域の値の変化を示している．この図から，高い精度の計測結果が得られていることが分かる．また，ここで得られた結果は，[8] でのシミュレーションによる性能評価で示された結果とほぼ同じ精度を示している．このことから，[8] で示された計測アルゴリズムが実ネットワーク環境においても有効であることが分かる．また，表 4 は ImTCP および TCP Reno それぞれの場合について，データ転送中に CPU へかかる負荷の平均を示したものである．この表を見ると，CPU へかかる負荷の大きさは ImTCP と TCP Reno で大きな差は見られない．このことから，[8] で提案された計測アルゴリズムは CPU へ大きな負荷を与えずに実現できることが分かる．

4.1.2 クロストラヒックが TCP トラヒックの場合

次に，クロストラヒックとして TCP トラヒックが発生している場合について，4.1.1 と同様に ImTCP の計測精度の評価

表 3　実験ネットワーク環境を構築する PC のスペック

	Sender	Receiver	PC Router	Traffic generator
CPU	Intel Pentium 4 3.0GHz	Intel Pentium 4 3.4GHz	Intel Pentium 4 3.0GHz	Intel Pentium 4 3.4GHz
Memory	1,024MB	1,024MB	1,024MB	1,024MB
OS	FreeBSD 4.10	FedoraCore 4	FreeBSD 4.10	FedoraCore 4

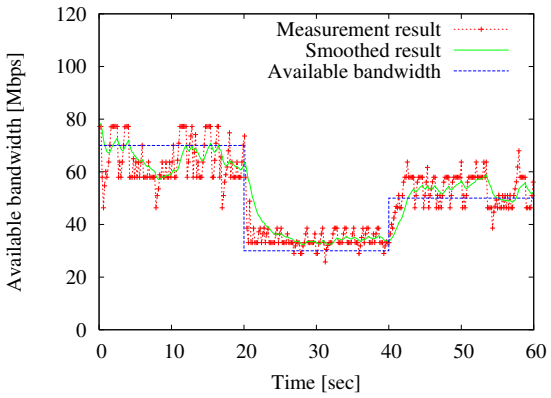

図 4　利用可能帯域および計測結果の変化（クロストラヒックが UDP トラヒックの場合）

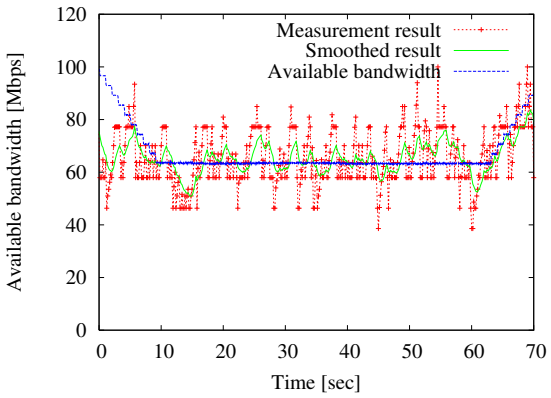

図 5　利用可能帯域および計測結果の変化（クロストラヒックが TCP トラヒックの場合）

を行う．TCP トラヒックは，1 秒おきに合計 10 本の TCP コネクションを確立し，それぞれのコネクションが 60 秒間データ転送を行うことにより発生させる．ただし，TCP コネクションはデータ転送を行う際には，エンドホスト間のネットワークパスの空き帯域を全て使用してデータ転送を行うため，利用可能帯域が存在しない．したがって，実験を行う際にはこの点に留意して行う必要がある．今回の実験では，受信側の広告ウィンドウサイズを用いてデータ転送量の上限値を決定することにより利用可能帯域が存在するネットワーク環境を構築した．図 5 は，経過時間に対する計測結果，平滑化された計測結果および実際の利用可能帯域の変化を示したものである．この図を見ると，ImTCP の計測結果 が 4.1.1 と同程度の精度を示している．このことから，クロストラヒックが TCP トラヒックの場合においても ImTCP は高い精度で計測を行うことができることが分かる．

4.2　ImTCP-bg の評価

本節では，ImTCP-bg の評価を行う．評価は，4.1.2 と同様のネットワーク環境下で ImTCP-bg を用いてバックグラウンド転送を行い，クロストラヒックにどの程度影響を与えるかを観察する．ImTCP-bg の輻輳ウィンドウサイズおよび RTT の変化を表したものを図 6 に示す．この図を見ると，ImTCP-bg が ImTCP の計測結果に基づいて輻輳ウィンドウサイズを制御することによって，RTT をほとんど増加させずにデータ転送を行うことができることが分かる．また，ImTCP の計測結果が実際の利用可能帯域よりも大きくなり輻輳が発生した場合においても，RTT の変化を観察することによって早期に輻輳を検出し，適切に輻輳ウィンドウサイズを減少させていることが分かる．

最後に，ネットワークの空き帯域の利用率およびクロストラヒックへ与える影響の大きさの評価として，ImTCP-bg および TCP Reno のスループットの変化を図 7 に，バックグラウンド転送を行っていない場合，ImTCP-bg を用いてバックグラウンド転送を行った場合および TCP Reno を用いてバックグラウンド転送を行った場合のクロストラヒックのスループットの変化を図 8 に示す．図 7 から，TCP Reno は ImTCP-bg よりも大きなスループットを得られていることが分かる．しかしながら，そのスループットは利用可能帯域を大きく超えており，図 8 から，クロストラヒックのスループットを押し下げていることが分かる．このことから，TCP Reno を用いた場合には他のトラヒックに大きな影響を与えており，他のトラヒックに影響を与えずにネットワークの空き帯域のみを有効に利用するというバックグラウンド転送の性質を満たしていないことが分かる．これに対して図 8 から，ImTCP-bg を用いてバックグラウンド転送を行った場合にはクロストラヒックのスループットがほとんど低下していないことが分かる．また，図 7 を見ると，ImTCP-bg のスループットは利用可能帯域に近い値を示していることが分かる．以上のことから，ImTCP-bg は他のトラヒックに影響を与えずに空き帯域を有効に利用したバックグラウンド転送を行うことができていることが分かる．

5.　おわりに

本稿では，我々の研究グループがこれまでに提案したインラインネットワーク計測手法 ImTCP およびその計測結果に基づいたバックグラウンド転送方式である ImTCP-bg の実装を行った．実験ネットワークを用いた実装実験を行うことによりこれらの提案手法が実ネットワーク上においても，ImTCP の計測精度や ImTCP-bg の他のトラヒックに与える影響度に関して，シミュレーションの評価結果と同等の有効性があることを示した．

今後の課題としては，実験ネットワーク上において ImTCP-bg と他のバックグラウンド転送方式を比較し，ImTCP-bg の

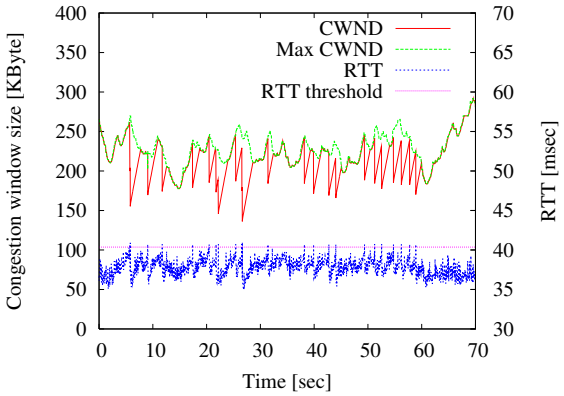

図 6　輻輳ウィンドウサイズおよび RTT の変化

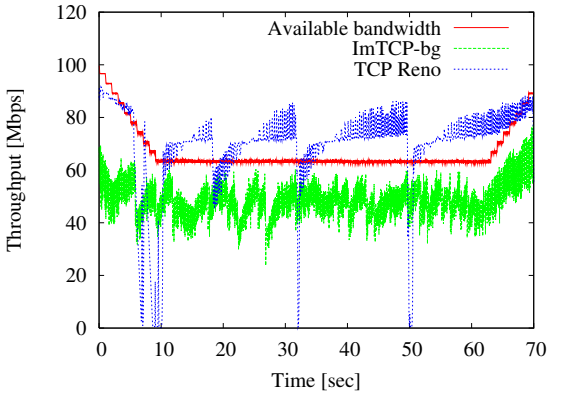

図 7　ImTCP-bg および TCP Reno のスループットの変化

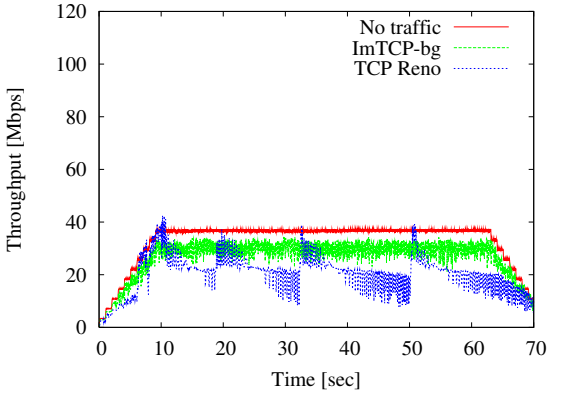

図 8　クロストラヒックのトータルスループットの変化

有効性を評価すること，インターネットを用いた実装実験を行い，これらの提案手法が実ネットワーク上においても有効であることを確認することなどが挙げられる．

文　献

[1] G. Pierre and M. van Steen, “Design and implementation of usercentered content delivery network,” in *Proceedings of the 3rd IEEE Workshop on Internet Applications*, June 2003.

[2] Akamai Home Page. available at `http://www.akamai.com/`.

[3] A. Rao, K. Lakshminarayanan, S. Surana, and I. Stoica, “Load balancing in structured P2P systems,” in *Proceedings of IPTPS 2003*, Feb. 2003.

[4] F. Dabek, B. Zhao, P. Druschel, J. Kubiatowicz, and I. Stoica, “Towards a common API for structured peer-to-peer overlays,” in *Proceedings of IPTPS 2003*, Feb. 2003.

[5] K. Czajkowski, S. Fitzgerald, I. Foster, and C. Kesselman, “Grid information services for distributed resource sharing,” in *Proceedings of the 10th IEEE International Symposium on High-Performance Distributed Computing (HPDC-10)*, Aug. 2001.

[6] Y. Zhao and Y. Hu, “GRESS – a grid replica selection service,” in *Proceedings of the 15th IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS-2003)*, Aug. 2003.

[7] J. Jha and A. Sood, “An architectural framework for management of IP–VPNs,” in *Proceedings of the 3rd Asia-Pacific Network Operations and Management Symposium*, Sept. 1999.

[8] M. L. T. Cao, G. Hasegawa, and M. Murata, “Available bandwidth measurement via TCP connection,” in *Proceedings of IFIP/IEEE MMNS 2004 E2EMON Workshop*, Oct. 2004.

[9] B. Melander, M. Bjorkman, and P. Gunningberg, “A new end-to-end probing and analysis method for estimating bandwidth bottlenecks,” in *Proceedings of IEEE GLOBECOM 2000*, Nov. 2000.

[10] M. Jain and C. Dovrolis, “End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput,” in *Proceedings of ACM SIGCOMM 2002*, Aug. 2002.

[11] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, and L. Cottrell, “pathChirp: Efficient available bandwidth estimation for network paths,” in *Proceedings of NLANR PAM 2003*, Apr. 2003.

[12] T. Tsugawa, G. Hasegawa, and M. Murata, “Background TCP data transfer with inline network measurement,” in *Proceedings of Asia-Pacific Conference on Communications 2005*, Oct. 2005.

[13] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, “Web caching and Zipf-like distributions: Evidence and implications,” in *Proceedings of IEEE INFOCOM 1999*, Mar. 1999.

[14] M. Crovella and P. Barford, “The network effects of prefetching,” in *Proceedings of the IEEE INFOCOM 1998*, Mar. 1998.

[15] A. Venkataramani, P. Yalagandula, R. Kokku, S. Sharif, and M. Dahlin, “The potential costs and benefits of long term prefetching for content distribution,” *Computer Communication Journal*, vol. 25, pp. 367–375, Mar. 2002.

[16] A. Venkataramani, R. Kokku, and M. Dahlin, “TCP Nice: A mechanism for background transfers,” in *Proceedings of OSDI 2002*, Dec. 2002.

[17] A. Kuzmanovic and E. W. Knihtly, “TCP-LP: A distributed algorithm for low priority data transfer,” in *Proceedings of IEEE INFOCOM 2003*, Apr. 2003.

[18] The VINT Project, “UCB/LBNL/VINT network simulator - ns (version 2).” available at `http://www.isi.edu/nsnam/ns/`.

[19] FreeBSD Home Page. available at `http://www.freebsd.org/`.

[20] M. L. T. Cao, G. Hasegawa, and M. Murata, “A new available bandwidth measurement technique for service overlay networks,” in *Proceedings of IFIP/IEEE MMNS 2003 E2EMON Workshop*, Sept. 2003.

[21] R. L. Carter and M. E. Crovella, “Measuring bottleneck link speed in packet-switched networks,” *International Journal on Performance Evaluation*, vol. 27–28, pp. 297–318, Oct. 1996.

[22] M. K. McKusick and G. V. Neville-Neil, *The Design And Implementation Of The FreeBSD Operating System*. Addison-Wesley, 2004.