片道遅延，可用帯域とボトルネック回線容量推定ツールの改良とその評価

2003MT034 片岡 哲哉　　2003MT059 森 真一朗
指導教員　後藤 邦夫

1　はじめに

近年，急速なインターネットの普及に伴い動画像配信サービスなど様々なコンテンツの需要が増加してきた．それに伴い，回線が混雑することに起因する遅延等により，目的通りのサービスを受けることができないおそれがでてきた．そこで，全てのユーザが安定して回線を多く消費するこれらのサービスを利用できるために，事前に利用可能な帯域幅や，経路上で最も狭い回線の容量などを把握できるツールが提案されている [1][2][4].

そのツールの１つである PathTester[4] とは，既存の帯域幅計測ツールで用いられているワンパケット，パケットペア，パケットトレインの 3 つの手法を応用し，短時間で少ない探査パケットを送信し，そこから必要な情報を推定できるツールである．また，両端ホストでの正確な推定値を得るために，NTP サーバと時刻同期をとり，時刻補正をする．

しかし，このツールには下記の２つの問題点がある．

１つ目は，可用帯域の推定では，狭い回線で計測できない点や，推定にかかる無駄な探査パケットが発生してしまう点，２つ目に，短時間で推定するために，一定時間内に到達すると思われる返信探査パケットの待ち時間が適当な値ではない点である．

そこで本研究では，可用帯域の推定におけるパケットトレインの送信レートの範囲を調整することで，測定可能な帯域幅を広げ，また，受信側ホストの受信レートを用いて送信間隔を決定し，さらに，受信待ち時間をネットワーク環境に合わせて動的に決定することで，推定時間の短縮を試みた．上記の手法を組み合わせることで，様々な実ネットワーク上で，広範囲の帯域幅をより短時間で推定できると期待される．

片岡哲哉は主にプログラムの作成を，森真一朗は主にモデルの作成を担当した．

2　PathTester の概要

本節では，本研究での改良点に関する既存 PathTester の手法のみを説明する．

2.1　可用帯域の推定手法

可用帯域とは，経路上で一番遅い (狭い) 回線で使用できる残りの帯域であり，その残りの帯域を全て使いきることで測定できるが，長時間回線を占有するのは迷惑行為となるので，既存 PathTester では，できるだけ短時間で推定する手法 (図 1) がとられている．

探査パケットが分割送信されると計測が不可能となるので，合計約 10k オクテットのパケットトレインの探査パケットを最大長の 1400 オクテットで連続送信し，送信レートと受信レートがほぼ同じになるときの最大送信レートを可用帯域の推定値とする．

ここで，受信レートは，受信側の合計受信オクテット数と受信開始から終了までの時間を用いて計算する．また，送信レートは，送信側の合計送信オクテット数と，送信開始から終了までの時間を用いて計算する．このとき送信に要する時間は，送信間隔が最小の値から 2 のべき乗に比例して最大の値へ推移している．

しかし，この手法では，送信間隔が最小の値から最大の値までのすべての間隔を線形探索で計算しており，無駄な送信が発生してしまう．

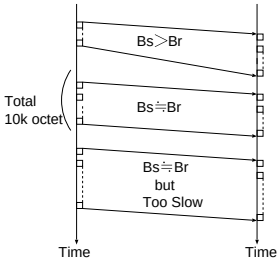

図 1　可用帯域の推定の流れ

2.2　受信待ち時間の推定手法

既存 PathTester では，探査パケットの待ち時間の推定手法は下記の通りである．

パケットペア，パケットトレインにおいて，最初の探査パケットはどのくらいの間隔で届くか分からないため，少し長めに待ち，2 つ目の探査パケットからは，わずかな間隔だけ待つ．

もし限られた時間内に届かなかった場合はあきらめて，片道伝送遅延，可用帯域，ボトルネック回線容量の各推定処理が終わるごとに全て届いたものとみなす．損失したとあきらめた後，遅れて届いた探査パケットは全て破棄する．

しかし，この手法はネットワークの状況を考慮せずに受信待ち時間を決めている．極端に言えば，100kbps の帯域幅の回線でも 100Mbps の回線でも，また，クロストラフィックに起因するルータでの待ち行列遅延が発生しても，受信待ち時間は同じ値に設定されている．

3　改良 PathTester の目標とその解決手法

本節では，改良 PathTester の目標と解決方法を説明する．実現したい目標は以下の 2 点である．

(1) 可用帯域の推定における測定可能な帯域幅の拡大と推定時間の短縮…測定可能な帯域幅を拡大し，さらに

その推定時間を短縮する．

(2) 受信待ち時間の短縮…可用帯域，片道伝送遅延，ボトルネック回線容量の推定時間を短縮する．

3.1 可用帯域推定における測定可能な帯域幅の拡大と推定時間の短縮

既存 PathTester での可用帯域の推定手法では，パケットトレインの送信間隔の最大値が小さすぎるため，500kbps 以下では計測できないという問題点がある．

そこで本研究では，100kbps 以上の回線を測定できることを目標とする．目標値である 100kbps は，大部分の家庭において一般的に使用している回線容量の最低値を参考にして採用した．

送信間隔を大きく設定すれば，帯域幅が小さい回線でも測定できるようになるが，送信間隔を大きく設定しすぎると，推定時間が長くなってしまうため，適当な値を設定する必要がある．エミュレータ GINE[3] で実験し，その適当な値を抽出する．エミュレータの実験環境は図 2 の通りである．

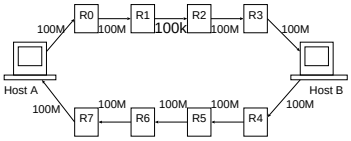

図 2　実験環境モデル

Out 方向のボトルネック回線速度を 100kbps にし，残りの回線速度は全て 100Mbps とする．

また，既存 PathTester では iunit の範囲の数だけ (範囲が 0-15 なら 16 回) パケットトレインを送信しているため，探査パケットの送信が発生しすぎてしまう．

そこで本研究では，受信側ホストでの受信レートを用いて，パケットトレインにおける送信間隔の調整をする．

可用帯域の推定開始時，パケットトレインの探査パケットの送信間隔として，限りなく小さな値 (実際は 10 マイクロ秒．今後 waittime とする) だけ待つことから始め，推定を開始する．

送信側ホストでは，返答されたパケットトレインに含まれる情報をもとに，受信側ホストの受信レート (MbpsR) を計算する．

ここで得られた，受信側ホストでの受信レートと，送信側ホストでの送信レートを比較することで，続いて送信されるパケットトレインの送信間隔 (waittime) を決定する．

図 3 に，waittime の状態遷移図を示す．

本研究では，10 回程度のパケットトレインを送信し，各パケットトレインから得られた，受信側ホストでの受信レートを用いて，平均，分散，標準偏差を計算し，可用帯域の推定値を決定する．

ここで，得られる受信レートのサンプルは，極端にずれたサンプルを除き，正規分布に従うとする．推定値 (e_BW) の範囲は，平均値 (a_BW) に 3 倍した標準偏差 (s_BW) を加減算することで，得られたサンプル群の 99.73% を満たす．その式を式 (1) に示す．

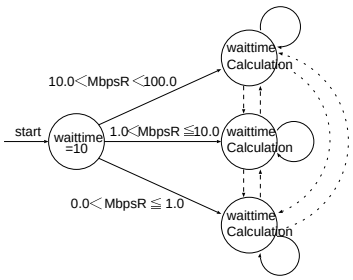

図 3　waittime の状態遷移図

$$\text{a\_BW} - 3 \cdot \text{s\_BW} < \text{e\_BW} < \text{a\_BW} + 3 \cdot \text{s\_BW} \quad (1)$$

3.2 受信待ち時間の短縮

送信された探査パケットが，なんらかの理由でインターネット上で損失した場合，端末は探査パケットが遅延しているのか損失したのかを知る事ができない．永続的に待っている訳にはいかないので，いつまで受信待ちをするか，この時間の設定は重要である．

本研究では，ネットワークの回線状況に応じて探査パケットが往復するのに必要とする時間情報を取得し，その情報を利用することで，動的に受信待ち時間を決定，利用している．

NTP サーバから時刻情報を取得した後，受信待ち時間を設定する工程 (受信待ち設定工程) を作成した．受信待ち設定工程では，パケットサイズが 1400 オクテットの探査パケットを 50 回程度往復させ，それぞれの探査パケットが往復でどれだけ遅延したかを計測し，それぞれのデータを用いて，平均 (ave_Timeout)，分散，標準偏差 (sta_Timeout) を用いて，受信待ち時間 (Timeout) を設定する．その時に用いられる計算式を式 (2) に示す．

$$\text{Timeout} = \text{ave\_Timeout} + 3 \cdot \text{sta\_Timeout} \quad (2)$$

式 (2) で求めた受信待ち時間の値を，各推定工程ではパケットサイズによって変化させ，最後に送信した探査パケットと最初に受信する探査パケットの待ち時間として使用する．

4 実験結果と考察

本節では，それぞれの改良点を加えた PathTester と改良点全てを統合した改良 PathTester が，既存 PathTester より広範囲の可用帯域を短時間で推定できるか確認することを目標とした実験について述べる．

実ネットワーク上で既存 PathTester とそれぞれの改良点を加えた PathTester，また，エミュレータ GINE を用いて，既存ツールと改良 PathTester で比較実験をした．最後に，研究員宅間で実験をし，改良 PathTester の有用性を確認した．

ここで，改良 PathTester の可用帯域推定においては，受信側ホストの受信レートを用いて，パケットトレインの送信間隔を決定するアルゴリズムを採用した．本研究では片道のみ計測しているため，送信間隔の拡大に iunit は使用しない．

4.1 測定可能な可用帯域拡大の実験結果

表 1 は図 2 のエミュレータ環境での実験結果である．

表 1　測定可能な可用帯域拡大の実験結果

iunit	bw(Mbps)		cap(Mbps)		time info(sec)		
	out	in	out	in	p2	p3	total
0-15	0.0	78.8	NA	NA	NA	NA	NA
0-16	0.0	79.2	NA	NA	NA	NA	NA
0-17	0.1	51.2	0.1	727	16	36	52
0-18	0.1	52.5	0.1	848	18	36	54
0-19	0.1	56.6	0.1	880	22	39	61
0-20	NA	NA	NA	NA	NA	NA	NA

(NA: Not available)

表 1 の iunit は，パケットトレインの送信間隔を決定するのに使用されている要素である．既存 PathTester では，パケットトレインにおける送信間隔を，この iunit を 2 のべき乗の数として使用することで変更することができる．

iunit を増加させることにより，パケットトレインにおける個々の探査パケットの送信間隔が大きくなり，それにより，狭い帯域幅も測定可能となる．

表 1 の Times info の p2(phase2)，p3(phase3)，total はそれぞれ，可用帯域の予備調査，可用帯域の本調査，それら 2 つの合計に要した処理時間を示している．

この実験から，送信間隔の範囲を小さくしすぎると (表 1 の 0-15 ，0-16)，狭い回線では測定不能となり，反対に送信間隔の範囲を大きくしすぎても (表 1 の 0-20)，測定不能となることが分かる．

よって 100kbps 以上の回線で正確な測定結果が得られ，かつ測定時間が最小になる iunit の範囲は，合計時間が最小な 0-17 である．

4.2 可用帯域推定時間の短縮の実験結果

エミュレータで動作確認をした後，研究員宅間で実験をする．実験環境は，片岡宅，森宅の ISP がどちらも DION である．また，片岡宅の回線は，ADSL 回線 (上り：1Mbps，下り：24Mbps)，森宅の回線が ADSL 回線 (上り：1Mbps，下り:12Mbps) である．以後，実ネットワークでの実験はすべて上記の環境とする．

表 2 に実験結果を示す．表 2 の ex，im はそれぞれ，既存 PathTester，改良 PathTester を示す (今後同様に表記)．time は可用帯域の推定にかかった時間 (秒) である．なお，片岡宅から森宅への Hops は上り下りともに 6 であり，森宅から片岡宅への Hops は上り下りともに 3 である．表 2 より，既存 PathTester の推定値と比較しても同程度な可用帯域の推定値が得られ，かつ推定時間の短縮が実現できたことが分かる．

表 2　可用帯域推定時間の短縮の実験結果

Cont(Mbps)		BW(Mbps)		Dly(msec)		time(sec)	
Out	In	ex	im	ex	im	ex	im
ADSL							
1	24	0.90	0.96	18.6	16.9	11	2
		0.98	0.97	17.8	16.3	11	2
1	12	0.91	0.89	9.9	8.3	11	2
		0.90	0.89	9.7	7.9	11	2

4.3 受信待ち時間短縮の実験結果

エミュレータで動作確認をした後，研究員宅間で実験をする．

表 3 に実験結果を示す．表 3 の time は，片道伝送遅延，可用帯域，ボトルネック回線容量の推定にかかった時間である．なお，片岡宅から森宅への Hops は上り下りともに 6 であり，森宅から片岡宅への Hops は上り下りともに 3 である．

表 3　受信待ち時間短縮の実験結果

Cont(Mbps)		BW(Mbps)		Dly(msec)		time(sec)	
Out	In	ex	im	ex	im	ex	im
ADSL							
1	24	0.99	0.99	20.2	48.0	92	20
		0.98	0.92	20.5	46.3	92	21
1	12	0.89	0.90	11.3	87.5	92	20
		0.60	0.54	11.3	84.9	91	21

表 3 より，可用帯域は，既存 PathTester と比較しても同程度な推定値が得られ，かつ各推定にかかる時間の短縮が実現できたことが分かる．しかし，片道伝送遅延の推定値が過大評価となってしまった．これは，受信待ち時間に，パケットサイズに比例した値のみではなく，両端ホスト間での物理的距離も考慮していないことが原因だと考えられる．

4.4 既存ツールとの比較

改良 PathTester が既存ツールと比べて優れている点を示すため，エミュレータ GINE が動作しているルータをはさんで HostA，HostB 間で比較実験をする．

比較に使用したツールは, Pathload である．Pathload はパケットトレイン推定法によって，可用帯域を推定する．実験で使用した PC を表 4 に示す．

表 4　使用した PC の環境

PC	CPU	RAM	NIC
Host A	Celeron(500MHz)	512MB	epro100
Host B	Celeron(500MHz)	128MB	epro100
Host C	Celeron(1200MHz)	256MB	e100
Router	Celeron(2.53GHz)	512MB	e100

4.5 既存ツールとの比較実験

エミュレータ GINE を用いて図 4 のような実験環境モデルを再現し、Out 方向のボトルネック回線速度を 5Mbps とし，残りの回線速度を全て 100Mbps とした．

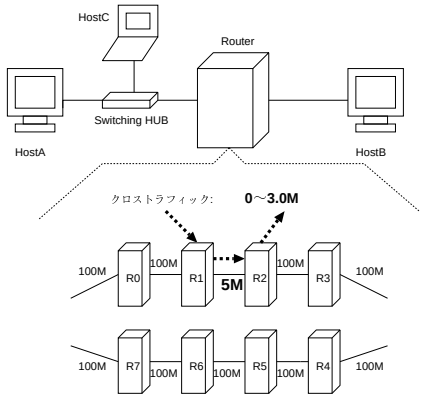

図 4　実験環境モデル

iperf を用いて HostC からクロストラフィックを挿入し，0.5Mbps から 3.0 Mbps までレートを増加させ，UDP パケットを流した．

実験結果を表 5 に示す．

表 5　既存ツールとの比較の実験結果

Cross	Pathload		PathTester	
(Mbps)	BW(Mbps)	time(sec)	BW	time
0	2.68	34.4	4.48	69.2
0.5	1.83	76.0	3.54	69.1
1.5	1.84	75.6	2.65	69.6
3.0	0.87	129.2	1.36	69.4

表 5 より，Pathload は，クロストラフィックが投入されなければ，改良 PathTester より短時間で推定できるが，クロストラフィックが増加するにつれて，推定時間も増加する．また，推定値も若干精度に欠ける．

改良 PathTester は Pathload と比較すれば，推定値の精度が高く，かつ短時間で推定できることが確認できる．しかし，クロストラフィックが増加するにつれて若干改良 PathTester も精度に欠ける．

両ツールで推定値の精度が若干欠けている原因として，エミュレータ GINE では，ルータ内のキューに貯められる容量が小さすぎるため，帯域を制限した場合，キューの容量を越えるレートでパケットが到着すると，ロスが多発してしまうことが考えられる．

4.6 実ネットワーク上での実験

改良 PathTester が実ネットワーク上でも動作することを確認するため，研究員宅間で実験をした．

表 6 に改良 PathTester の実験結果を示す．time は，片道伝送遅延，可用帯域，ボトルネック回線容量の推定にかかった時間である．なお，片岡宅から森宅への Hops は上り下りともに 6 であり，森宅から片岡宅への Hops は上り下りともに 3 である．

表 6　研究員宅間での改良 PathTester の実験結果

Cont(Mbps)		BW(Mbps)		Dly(msec)		time(sec)	
Out	In	ex	im	ex	im	ex	im
ADSL							
1	24	0.98	0.98	20.6	13.6	92	15
		0.98	0.98	17.8	13.3	92	15
1	12	0.91	0.88	11.3	76.7	92	15
		0.89	0.89	9.7	66.2	92	15

表 6 より，可用帯域は，既存 PathTester と比較しても同程度な推定値が得られ，かつ推定時間の短縮が実現できたことが分かる．しかし，片道伝送遅延に関しては，受信待ち時間の短縮で与えられた値が適当ではないため，この実験においても精度に欠ける．

5 おわりに

本研究では，広範囲の可用帯域を短時間で推定できるツールを提案し，実験した．実験ネットワーク環境，実ネットワーク環境において，短時間で可用帯域が推定できることを確認した．課題として下記の 3 点がある．

- ボトルネック回線容量の推定における精度を向上させ，探査パケットサイズによる増加傾向を利用した階層化クラスタ分析で抽出された推定値の理論的裏付けを取ること．
- 片道伝送遅延の推定工程において，受信待ち時間をパケットサイズのみに比例した値を利用するのではなく，両ホスト間の物理的距離も考慮に入れた値に変更すること．
- 可用帯域を推定する際，Out 方向のみを考慮しているため，今後は In 方向も同様のアルゴリズムを使って推定すること．

参考文献

[1] Chen, L.-J., Sun, T., Yang, G., Sanadidi, M. and Gerla, M.: CapProbe: A Simple and Accurate Capacity Estimation Tool, Proc. of ACM SIGCOMM2004, Portland (2004). (http://www.cs.ucla.edu/NRL/CapProbe/).

[2] Dovrols, C., Jain, M. and Prasad, R.: Pathload a measurement tool for end-to-end available bandwidth (accessed Jun. 2005). (http://www.cc.gatech.edu/fac/Constantions. Dovrolis/pathload.html).

[3] Ihara, A., Murase, S. and Goto, K.: IPv4/v6 Network Emulator using Divert Socket., *Proc. of 11th International Conference on Systems Engineering (ICSE2006)*., pp. 159–166.

[4] 吉田秀考：インターネットにおける可用帯域幅の推定と伝送遅延の評価，修士論文 (Apr. 2005).