特別研究報告

題目

インライン計測に基づいた TCP 輻輳制御方式の実ネットワークにおける性能評価

指導教員
中野 博隆 教授

報告者
児玉 瑞穂

平成 19 年 2 月 20 日

大阪大学 基礎工学部 情報科学科

平成 18 年度 特別研究報告

インライン計測に基づいた TCP 輻輳制御方式の実ネットワークにおける性能評価

児玉 瑞穂

内容梗概

現在のインターネットアプリケーションが標準的に使用しているトランスポート層プロトコルである TCP において，TCP Reno と呼ばれるバージョンが現在最も普及している. 近年インターネット環境の多様性，複雑性が増大してきたことにより，TCP Reno には多くの問題点があることが明らかになっている. これらの問題の多くは，TCP がネットワークの輻輳の有無を判断するのにパケット廃棄の発生のみを指標としていること，およびウィンドウサイズの増加量と減少量を決定するパラメータがネットワーク環境に関係なく常に一定であるということに起因する.

我々の研究グループでは，これまでとは全く異なる輻輳制御方式である TCP Symbiosis を提案している. 従来の手法である TCP Reno やその改善手法はネットワークの輻輳の有無を判断するのにパケット廃棄の発生を指標としていたが，本手法はインラインネットワーク計測によって取得したネットワークパスの利用可能帯域に関する情報を利用する．そして，数理生態学において生物の個体数の変化を表すモデルである，ロジスティック増殖モデルおよびロトカ・ヴォルテラ競争モデルを適用したウィンドウ制御アルゴリズムによって輻輳制御を行う.

提案方式の評価はこれまでコンピュータシミュレーションによって行われてきたが，ネットワークの計測結果を利用する本提案手法は，実ネットワーク環境における実験を通じた評価が不可欠である. そこで本報告では，実験室ネットワークおよびインターネット公衆回線を用いたデータ転送実験を通じて，TCP Symbiosis の性能評価を行った. その結果，本手法を用いることにより，大阪-米国西海岸インターネット環境において，TCP Reno に比べて最大 104%のスループット向上を達成できることが明らかとなった．また，本手法が周期的なパケット廃棄を必要としない効率的なデータ転送を可能にすること，およびネットワークの帯域遅延積に対する高いスケーラビリティを持つことを示す.

主な用語

輻輳制御，インライン計測，利用可能帯域，ロジスティック増殖モデル，ロトカ・ヴォルテラ競争モデル

目 次

図 目 次

表 目 次

1　はじめに

Transmission Control Protocol (TCP) [1] は，現在のインターネットアプリケーションが標準的に使用するトランスポート層プロトコルである．TCP は 1970 年代に設計され，それを規定する最初の Request for Comments (RFC) は 1981 年に発行された [2]. そして，その後のインターネットの発展にともない改良が繰り返し行われている (たとえば，[3-5]). TCP は様々な機能を持っているが，その中で最も重要なのは輻輳制御方式 [1] である．この機能により，TCP はネットワーク内で競合する複数のコネクション間でネットワーク帯域を公平に分配するために，データ転送速度を調節している．TCP はウィンドウサイズ，すなわち，確認応答無しに一度に送出できるデータ転送量を増減させることによってデータ転送速度の調節を行う．

現在，TCP は Reno と呼ばれるバージョンが最も普及している．TCP Reno のウィンドウサイズ制御アルゴリズムは，図 1 に示すように，パケット廃棄の発生を検出するまでウィンドウサイズを線形的に増加させ，パケット廃棄の発生を検出するとウィンドウサイズを半減させるというものである．この制御方式はその増減方法から Additive Increase Multiplicative Decrease (AIMD) 方式と呼ばれている．この方式を用いることで，ネットワーク輻輳に関する情報が各コネクションへ同時に伝わるのであれば，複数の TCP コネクションが独立してウィンドウサイズを制御しても，それらの間でネットワークの帯域を有効にかつ公平に利用できることが知られている [6].

しかし，近年インターネット環境の多様性，複雑性が増大してきたことにより，TCP Reno には多くの問題点があることが明らかになってきている [7-11]．これらの問題の多くは TCP がパケットの廃棄発生のみをネットワーク輻輳の指標として用いていること，およびウィンドウサイズの増加量と減少量を決定するパラメータがネットワーク環境に関係なく常に一定であるということに起因する．例えば，ネットワーク帯域や遅延が大きい環境において，TCP Reno はリンク帯域を使い切ることが出来ないために，スループットを低下させてしまうことが挙げられる [11]．この問題は，ウィンドウサイズの増加量を決定するパラメータが小さく (1 ラウンドトリップ時間 (RTT) あたり 1 パケット)，ウィンドウサイズの減少量を決定するパラメータが大きい (パケット廃棄発生時に半減させる) ことに起因する．この問題に対する解決法は [11-16] などにおいて提案されている. [11, 12] では，TCP Reno のパケット廃棄の発生のみをネットワーク輻輳と判断する基本機構を引き継ぎ，ウィンドウサイズの増減幅を決定するパラメータをネットワーク環境に応じて静的あるいは動的に調節することでスループットの改善を行っている．しかし，これらの方法は共存するコネクションとの公平性を維持することができない，および周期的にパケット廃棄が発生するという問題点がある．[13, 14] では，各パケットの RTT を監視し，その増大をネットワーク輻輳の初期段階の

指標として利用する手法を提案している．この手法は，単独で用いられる場合にはスループット，公平性および収束速度などの面で優れていることが明らかになっている．しかし，TCP Reno や前者の改善手法と混在した環境においては，この手法を用いるコネクションのスループットが低下するという問題が指摘されている [15, 16]．[17, 18] ではネットワーク帯域が使いきれていないときにはウィンドウサイズを TCP Reno よりも大きく増加させ，輻輳発生時には，TCP Reno と同じ増加幅で輻輳ウィンドウサイズを増加させるという手法が提案されている．

TCP Reno および既存の改善手法の特徴の 1 つとして，送受信端末間のパスの利用可能帯域を知るための効率的な方法を持たないことが挙げられる．ウィンドウサイズは 1 RTT の間に送出可能なパケット数であるので，ある TCP コネクションにとって最適なウィンドウサイズは送受信端末間のネットワークパスにおいて現在利用可能な空き帯域 (利用可能帯域) と RTT の積で表される．TCP Reno はデータ転送速度を利用可能帯域まで到達させるためにウィンドウサイズを調節する機構をもっているため，ある意味では利用可能帯域を計測しているということが出来る．しかしながら，その方法はパケット廃棄が発生するまでウィンドウサイズを増加させるという単純なものであるために効率的ではない．このことは，現在のインターネットで主に用いられている TCP Reno を含め，パケット廃棄の発生をネットワーク輻輳の指標として用いる TCP の改善手法が，周期的なパケット廃棄の発生を避けることができないという問題の原因でもある．すなわち，TCP が何らかの手法を用いて，送受信端末間のパスの帯域に関する情報をすばやく，高い精度で取得することが出来れば，ウィンドウサイズの制御により効率の良い手法を用いることが可能となる．

ネットワークパスの帯域情報を計測するための手法はこれまでにも数多く提案されてきた [19-23]. しかしこれらの手法は，大量の計測パケットを必要としたり，新たな計測結果を取得するまでに多くの時間を必要とするため，TCP のウィンドウサイズ制御に直接利用することは出来ない．これらの問題点を持たず，TCP のデータ転送と併用可能な計測を行う手法として，我々の研究グループではインラインネットワーク計測 [24, 25] と呼ばれる手法を提案している．この手法は送受信端末間のパスの物理帯域および利用可能帯域を計測するために，TCP コネクションがデータ転送に用いるデータパケットおよび ACK パケットのみを用いるため，計測用パケットを必要としない．また非常に短い間隔 (1-4 RTT) で計測結果を継続的に取得できるため，ネットワーク状況の変化にすばやく追随することができる．また，TCP の輻輳制御アルゴリズムを変更するのではなく，送信側端末において TCP から IP へパケットが渡される部分に計測機構が組み込まれるため，任意の TCP の輻輳制御方式と組み合わせて用いることが可能である．

我々の研究グループではさらに，上記の利用可能帯域のインラインネットワーク計測手法を用いた TCP Symbiosis [26] という新たな TCP の輻輳制御方式を提案している．TCP Symbiosis

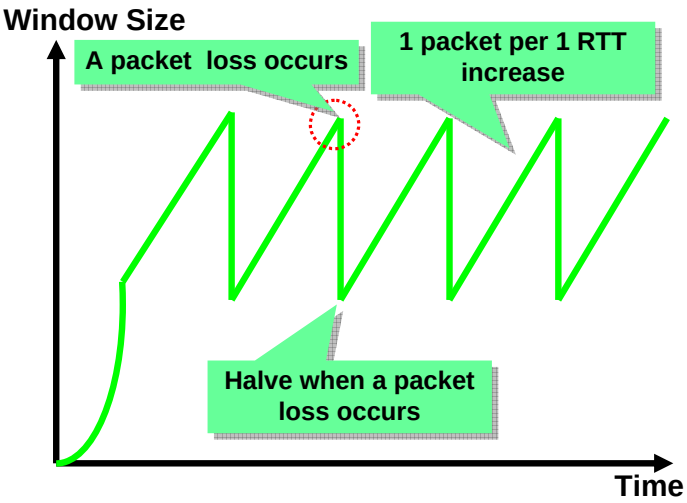


図 1: TCP Reno のウィンドウ制御

のアルゴリズムには，数理生態学において生物の個体数の変化を表すモデルとして有名なロジスティック増殖モデル，ロトカ・ヴォルテラ競争モデルが用いられている．これらのモデルを TCP の輻輳制御へ応用するために，生物の個体数をデータ転送速度に，個体数の上限値である環境容量をボトルネックリンク帯域に，種間の競争を同一リンク上の複数コネクションの競合にそれぞれ適用している．

提案方式の評価はこれまでコンピュータシミュレーションによって行われてきたが，ネットワークの計測結果を利用する本提案手法は，実ネットワーク環境における実験を通じた評価が不可欠である．そこで，本報告では，実験室ネットワークおよびインターネット公衆回線を用いたデータ転送実験を通じて，TCP Symbiosis の性能評価を行う．実験室ネットワークにおいては，RTT を変化させてデータ転送実験を行うことにより，TCP Symbiosis の基本的な性能に関する評価を行う．インターネット公衆回線を用いた実験においては，帯域遅延積の小さい環境である大阪-東京間のネットワーク，および帯域遅延積の大きい環境である大阪-米国カリフォルニア州間のネットワークにおいてデータ転送実験を行う．これらの実験を通して，TCP Symbiosis が周期的なパケット廃棄を必要としない効率的なデータ転送を可能にすること，およびネットワークの帯域遅延積に対する高いスケーラビリティを持ち，実ネットワーク環境においても性能を発揮できることを示す．

本報告の構成は以下のとおりである．2 章ではインラインネットワーク計測手法および TCP Symbiosis の説明を行う．3 章では実験室ネットワーク，および 4 章ではインターネット上での TCP Symbiosis の性能評価を行う．5 章で本報告のまとめと今後の課題を示す．

2　関連研究

2.1　インラインネットワーク計測

インラインネットワーク計測 [24, 25] (図 2 参照) とは，TCP コネクションがデータ転送に用いるデータパケットおよび ACK パケットのみを用いてエンドホスト間のネットワークパスの帯域情報を計測する技術である．この手法では，従来の手法とは異なり計測用のパケットを必要としないので，ネットワークに余計な計測負荷を掛けない．また，アクティブな TCP コネクションのデータ転送を利用することにより，送信端末の修正のみで計測手法を実装することができるという利点も存在する．我々のグループでは，この計測概念に基づく計測手法として ImTCP を提案している．この手法では，非常に短い周期 (1-4RTT) で継続的に利用可能帯域値を取得することができるため，ネットワークの状況の変化にすばやく追随することができる．

利用可能帯域を計測する際には，現在の利用可能帯域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用いて設定し，この区間の中から利用可能帯域を検索する (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与えることが出来る．また，計測に必要なパケット数を大幅に削減することも可能となる. 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出来る．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，[24, 25] を参照されたい.

1. Cprobe アルゴリズム [27] 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類する
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否かを確認する
4. 送出・受信結果から，利用可能帯域が含まれると考えられる小区間を選択する
5. 選択小区間に対応する計測ストリームの送出・受信結果から，利用可能帯域を算出する
6. 探索区間の再計算を行い，2. へ戻る

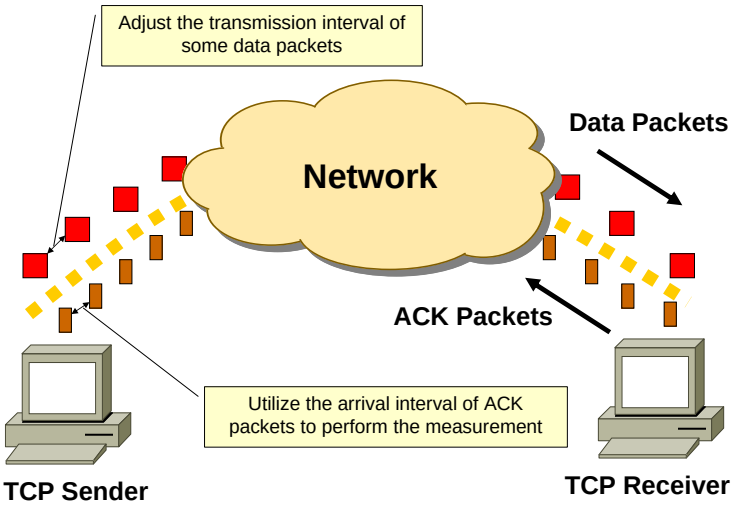


図 2: インラインネットワーク計測手法

また，ImTCP は，利用可能帯域の計測のために送出するストリームを決定するための下記パラメータを持つ．

- ストリーム数: 上記の 2. において探索区間を分割する小区間数に相当する
- パケット数: 1 つのストリームを構成するパケットの個数

3 章において，これらのパラメータが ImTCP の計測精度に与える影響に関する性能評価を行う．

2.2 TCP Symbiosis

TCP Symbiosis は，インラインネットワーク計測手法を用いることによって送受信端末間の物理帯域および利用可能帯域を取得し，数理生態学において生物の個体数の変化を表すモデルを適用したウィンドウサイズ制御アルゴリズムを用いて輻輳制御を行う．

一般に，ある特定の環境下において，特性 (環境容量，増殖率，および競争相手種の存在による増殖低下率) の等しい生物の個体数は収束し，収束する個体数は等しい．また，環境の変化が発生すると，各々の生物の個体数は直ちに変動する．したがって，生物の個体数の変化をデータ転送に適用すると，公平性および安定性，環境の変化に対する追随性などの利点が得られることが期待できる．

本章では，TCP Symbiosis のウィンドウサイズ制御アルゴリズム，輻輳制御アルゴリズムおよびその特性について述べる．

2.2.1 ウィンドウサイズ制御アルゴリズム

ロジスティック増殖モデルは，ある限られた領域の中で生息している一種の生物の個体数 N の変化を表している．一般的に，領域中の個体数が多くなるにつれてその増殖速度は大きくなる．しかし，自然界では環境や資源について様々な制約があるために，個体数の上限となる環境容量 K が存在する．これらの影響を考慮し，時間の経過にともなう生物の個体数の増殖過程は次式で表される [28].

$$\frac{d}{dt}N = \epsilon\left(1 - \frac{N}{K}\right)N \tag{1}$$

ここで，ϵ は種の内的自然増殖率を表す $(\epsilon > 0)$.

次に示すロトカ・ヴォルテラ競争モデルは，種内の競争による影響だけでなく種間の競争による影響をも考慮した，生物の個体数の変化を表すモデルである．種の個体数は式 (1) を拡張して次式で表される [28].

$$\frac{d}{dt}N_1 = \epsilon\left(1 - \frac{N_1 + \gamma_{12}N_2}{K_1}\right)N_1 \tag{2}$$

$$\frac{d}{dt}N_2 = \epsilon\left(1 - \frac{N_2 + \gamma_{21}N_1}{K_2}\right)N_2 \tag{3}$$

ここで，N_1，K_i，ϵ_i はそれぞれ種 i の個体数，環境容量，および内的自然増殖率を表す．γ_{ij} は，競争相手種 j の存在による種 i の増殖率低下を表すパラメータである．ここで，両方の種が同一の環境下に存在し，さらに同じ特性を持つと仮定する．すなわち，$K = K_1 = K_2$，$\epsilon = \epsilon_1 = \epsilon_2$，および $\gamma = \gamma_{12} = \gamma_{21}$ とする．このとき，両種が競争関係を持ちながらも片方の種が絶滅せず，共生を実現するための条件が $0 < \gamma < 1$ であることが知られている [28]. さらに，式 (2) , (3) を n 種の生物の個体数変化を表すように拡張すると，

$$\frac{d}{dt}N_i = \epsilon\left(1 - \frac{N_i + \gamma\displaystyle\sum_{i\neq j}^{n} N_j}{K}\right)N_i \tag{4}$$

となる．

ここで，生物の個体数を表したモデルを TCP の輻輳制御に適用するため，N_i を TCP コネクション i のデータ転送速度，K を物理帯域とみなす．このとき，式 (4) 中の $\displaystyle\sum_{i\neq j}^{n} N_j$ は他の

コネクションのデータ転送速度の合計値に相当するが，TCP コネクションはこの情報を直接知ることは出来ない．そのため，提案方式においては物理帯域 K と利用可能帯域 A_i を用いて $\sum_{i \neq j}^{n} N_j \sim (K - A_i)$ と近似する．これにより，式 (4) は次のように書き換えられる．

$$\frac{d}{dt}N_i = \epsilon \left(1 - \frac{N_i + \gamma(K - A_i)}{K}\right) N_i \tag{5}$$

TCP の輻輳制御は，ウィンドウサイズを制御することによって行われるため，データ転送速度で表された式 (5) をウィンドウサイズで表された式へ書き換える必要がある．ここで，ウィンドウサイズ w_i は，データ転送速度 N_i および TCP コネクションの RTT の最小値 τ_i を用いて $w_i = N_i\tau_i$ とする．これにより，式 (5) は次のように書き換えられる．

$$\frac{d}{dt}w_i = \epsilon \left(1 - \frac{w_i + \gamma(K - A_i)\tau_i}{K\tau_i}\right) w_i \tag{6}$$

最後に，微分方程式で表された式 (6) を積分することにより，次式のように $w_i(t)$ を導出することができる．

$$w_i(t) = \frac{w_i(0)\tau_i f_i(t)\left\{K - \gamma(K - A_i)\right\}}{w_i(0)\left(f_i(t) - 1\right) + \tau_i\left\{K - \gamma(K - A_i)\right\}} \tag{7}$$

ただし，

$$f_i(t) = e^{\epsilon t\left\{1 - \gamma\left(1 - \frac{A_i}{K}\right)\right\}} \tag{8}$$

であり，$w_i(0)$ は時刻 0 におけるウィンドウサイズとする．式 (7) を用いて，ある時刻を基準 $(t = 0)$ とし，その時のウィンドウサイズを $w_i(0)$ に代入することで，任意の時刻におけるウィンドウサイズ $w_i(t)$ を得ることができる．

式 (8) には e^x の計算が含まれているが，一般的に TCP が実装される OS のカーネルにおいては指数計算などの複雑な演算を行うことは大きな負荷となる．そこで，式 (8) をテイラー展開を用いて簡易な式に近似する．一般に，e^x を $x = a$ の周りで第 4 次の項までテイラー展開したもので近似すると，次式のようになる．

$$e^x \sim e^a \sum_{k=0}^{4} \frac{1}{k!}(x - a)^k$$

ここでは a は x の整数部分 (例えば，$0 \leq x < 1$ においては $a = 0$) とし，計算の必要となる e^a などは事前に計算結果をテーブルに保持する．以上のような近似計算を用いることで，式 (8) を小さい負荷で計算することができると考えられる．

2.2.2 輻輳制御アルゴリズム

TCP Symbiosis の輻輳制御アルゴリズムは，従来の TCP Reno に ImTCP のインライン計測機能を組み込んだものを元に拡張したものである．提案方式は，計測によって帯域の情報

が得られるまではTCP Renoと同じウィンドウサイズ制御を行う．また，重複ACKパケットの受信によってパケット廃棄を検出した場合にはTCP Renoと同様にウィンドウサイズを半減させる．タイムアウトによってパケット廃棄を検出した場合には，TCP Renoと同様にウィンドウサイズを1 [packet] に設定しSlow-Startフェーズを開始し，それまでに得られた全ての計測結果を破棄する．

ImTCPによってネットワークパスの帯域情報を取得できれば，前述したウィンドウサイズ制御アルゴリズムを用いて，TCP Renoと同様にACKを受信するたびに以下のようにウィンドウサイズの調節を行う．j 個目のACKを受信した時刻を t_j とすると，j 個目のACKを受信したときのウィンドウサイズは，式(7)において，時刻0を $j-1$ 個目のACKを受信した時刻 t_{j-1} とし，$t = t_j - t_{j-1}$ を代入することによって求める．

2.2.3 パラメータの設定

TCP Symbiosisは2つの制御パラメータ，ϵ と γ を持つ. γ はボトルネックリンクを共有する他のコネクションの影響を考慮する度合いを表し，各コネクションの物理帯域 (K_i) の値に関わらずウィンドウサイズが正の値に収束するためには $0 < \gamma < 1$ である必要があることがわかっている [28]．式(9)および(10)より，γ の値を設定する際には，収束速度と収束時にボトルネックリンクに蓄積されるパケット量に関するトレードオフを考慮する必要があることがわかる．すなわち，γ を小さくすると，収束速度が早くなる反面，収束時にボトルネックリンクに蓄積されるパケット量が大きくなる．したがって，ボトルネックリンクのバッファサイズを考慮して γ を設定する必要があると考えられる．

一方 ϵ は，式(6)に示されるように，収束速度を決定するパラメータである．一般的に，式(1)を離散化した場合，ϵ が2より大きくなると個体数が振動し続けて収束しないことが知られている [28]．これに対しTCP Symbiosisにおいては，式(6)を積分した式(7)を用い，振動が発生しないような離散化を行っているため，この点に関しては ϵ の値に制限は発生しない．すなわち，ϵ を大きくすればするほど収束速度が速くなる．しかし，特に広帯域・高遅延ネットワークにおいては，ϵ を大きくすると1つのACKパケットを受信した際のウィンドウサイズの増加量が大きくなり，多くのパケットをバースト的に送出しネットワークに影響を与えることが考えられる．

なお，3, 4章における実験においては，$\gamma = 0.9$，$\epsilon = 1.95$ を用いている．

2.2.4 特性

スケーラビリティおよび収束速度

物理帯域 K および利用可能帯域 A が一定であると仮定すると，TCP Symbiosis を用いることによってウィンドウサイズはある値に収束する．その値 w^* は式 (6) を用いて $dw/dt = 0$ とすることで以下のように得ることができる．

$$w^* = \{(1-\gamma)K + \gamma\}A \tag{9}$$

このとき，ウィンドウサイズが任意の値 w_0 から $\rho w^* (0 < \rho < 1, w_0 < \rho w^*)$ へ増加するのにかかる時間 T_{proposed} は，式 (9) より，

$$\begin{aligned} T_{\mathrm{proposed}} =& \frac{1}{\epsilon\{1-\gamma(1-\frac{A}{K})\}} \log\left(\frac{\rho}{1-\rho}\frac{w^*-w_0}{w_0}\right) \\ <& \frac{1}{\epsilon(1-\gamma)} \log\left(\frac{\rho}{1-\rho}\frac{w^*-w_0}{w_0}\right) \end{aligned} \tag{10}$$

となる ($0 \leq A \leq K$ より)．一方 TCP Reno においてウィンドウサイズが w_0 から w^* へ増加するのにかかる時間 T_{reno} は，

$$T_{\mathrm{reno}} = (w^* - w_0)\overline{\tau} = [\{(1-\gamma)K + \gamma A\}\tau - w_0]\overline{\tau} \tag{11}$$

となる．ここで，$\overline{\tau}$ は平均の RTT とする．また，Delayed ACK オプションは無効であるとし，輻輳回避フェーズでウィンドウサイズが増加した場合の結果を示している．

式 (11) から，TCP Reno においてはウィンドウサイズの増加にかかる時間が，物理帯域 K および遅延時間 τ の増加に対して線形的に大きくなることがわかる．すなわち，帯域遅延積が大きくなるにつれて帯域遅延積を使い切るまでの時間が線形的に大きくなることを示している．一方，TCP Symbiosis においては，式 (11) からウィンドウサイズの増加にかかる時間が帯域遅延積の増加に対して対数的に大きくなる．このことから，TCP Symbiosis は TCP Reno よりも帯域遅延積の増加に対して高いスケーラビリティを持つことがわかる．

周期的なパケット廃棄の回避

ネットワーク内に n 本の TCP コネクションが存在する場合に，収束後のウィンドウサイズの合計値は以下の式で求められる．

$$\sum_{i=1}^{n} w_i = \frac{n}{1+(n-1)\gamma} K\tau$$

これは，n が増加するとウィンドウサイズの合計が増加することを示している．ここで，

$$\lim_{n\to\infty} \sum_{i=1}^{n} w_i = K\tau/\gamma$$

すなわち，ボトルネックリンクのバッファサイズが十分に大きい場合には，コネクション数に関わらずパケット廃棄は発生しないことを示している．このことは，従来の TCP Reno お

よび AIMD 方式を用いる改善手法がそのアルゴリズムの性質上バッファサイズをいくら大きくしても周期的なパケット廃棄の発生を避けられないことと対照的である．

コネクション間の公平性

TCP コネクションの収束後のウィンドウサイズは式 (9) によって独立に求められる．よって物理帯域 K が同じであるならば，各コネクションのウィンドウサイズが同じ値となるのは明らかである．

3　実験室ネットワークにおける性能評価

本章では，実験室ネットワークにおいてデータ転送実験を行い，TCP Symbiosis の性能評価を行う．

3.1　実験環境

実験室ネットワーク環境は以下のとおりである．

- 本実験のネットワークは Dummynet [29] という帯域，伝播遅延時間，パケット損失率などの制御を行うことが出来るツールを用いて構成する．実験ネットワークを図 3 に，送受信端末の性能を表 1 にそれぞれ示す．本実験では帯域幅を 100 Mbps，パケット損失率を 0 %に設定する．
- 送受信端末間には iperf [30] によるトラヒックを流す．
- 受信側端末には Delayed ACK オプション [31] を無効にするツールを用いる．
- TCP スループットが送受信端末のソケットバッファサイズに制限を受けることを避けるために，送信側端末のソケット送信バッファ，受信側端末のソケット受信バッファの値を帯域遅延積より十分大きな値に設定する．
- 1 秒間に発生するシステムのタイマー割り込み回数 (=HZ) は 20000 であり，計測の最小粒度は 50 μs となる．

3.2　ImTCP の計測結果が TCP Symbiosis に与える影響

TCP Symbiosis は ImTCP が計測した利用可能帯域の値を用いてウィンドウサイズの増減を決定する．そこで，クロストラヒックとして UDP トラヒックを流し，計測ストリーム数およびストリームあたりのパケット数を変化させて ImTCP の精度を変化させ，ImTCP の計測結果が TCP Symbiosis に与える影響を調べた．実験した結果を図 4 に示す．ImTCP の計測結果およびその指数移動平均，TCP Symbiosis を動作させたときによる輻輳ウィンドウの変化を表示している．

まず，パラメータであるストリーム数およびパケット数を変更したときの，ImTCP の計測精度の評価を行う．ImTCP の計測結果が最も優れているのは，ストリーム数を 10，パケット数を 20 としたとき (図 4 (d)) である．精度がもっとも高く，また利用可能帯域が 30 Mbps のときの計測結果が最も安定するのが早い．また，図 4 (a)〜(c) の場合よりも計測に必要と

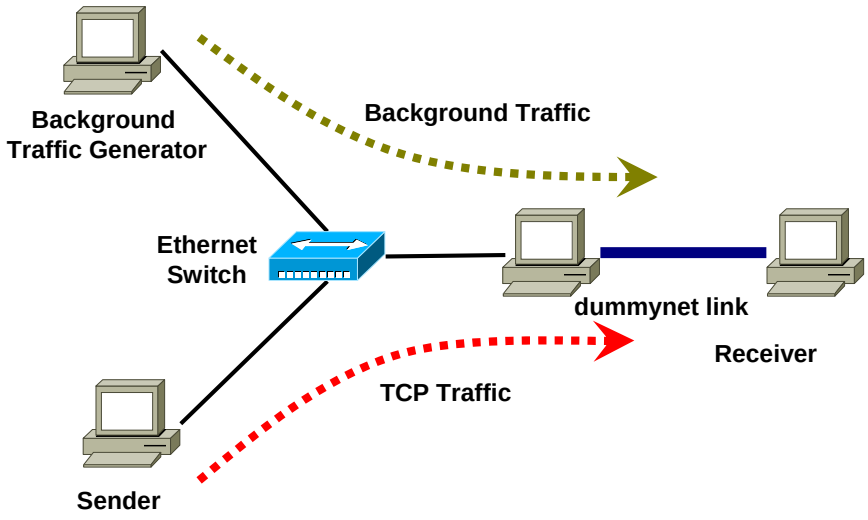


図 3: 実験室ネットワーク

するパケット数は多いが，利用可能帯域が 30 Mbps から 50 Mbps に変化に追随する性能についてもそれほど変化はない．ストリーム数を 10，パケット数を 30 としたとき (図 4 (e)) については計測頻度が低いために，利用可能帯域が 80 Mbps から 30 Mbps に変化する場合の追随性が劣化している．また，ストリーム数を 15，パケット数を 20 としたとき (図 4 (f)) については計測結果の精度が低い．これは，ストリーム数が多いため，粒度に対して探索区間が細かくなるため，利用可能帯域が含まれると考えられる探索区間を選ぶことができないことに起因すると考えられる．また，計測頻度が低いために，利用可能帯域の変化に対して追随することができていない．

次に ImTCP の計測結果に TCP Symbiosis がどのように影響を受けるか考察する．利用可能帯域が 30 Mbps となるときに注目すると，ImTCP の計測結果が 30 Mbps より大きい場合には，周期的にパケット廃棄が発生し，ImTCP の計測結果が 30 Mbps に近づくと，パケット廃棄の発生はなくなりウィンドウサイズは安定する．また，ImTCP の計測精度が最も高い，ストリーム数を 10，パケット数を 20 とした場合においては，周期的なパケット廃棄が発生する時間が最も短い．また，ストリーム数を 15，パケット数を 20 としたとき (図 4 (f)) の 0-40 sec の区間において利用可能帯域の値が 80 Mbps であるのに対し，ImTCP の計測結果の値が 60 Mbps 程度になっている．この計測結果に影響を受けて，TCP Symbiosis のウィンド

表 1: 実験室ネットワーク環境を構築する端末の性能

	送信側端末 (計測トラヒック用)	送信側端末 (クロストラヒック用)	受信側端末
CPU	Intel Pentium 4 1.90GHz	Intel Pentium 4 1.70GHz	Intel Xeon 2.80GHz
Memory	1,024MB	256MB	2,048MB
OS	Fedora Core 5	Fedora Core 5	Fedora Core 5

表 2: 実験における経過時間にともなうクロストラヒックの変化 (実験室ネットワーク)

t（時間）	$0 \leq t < 40$	$40 \leq t < 80$	$80 \leq t < 120$
クロストラヒックの帯域 [Mbps]	20	70	50
予測される利用可能帯域 [Mbps]	80	30	50

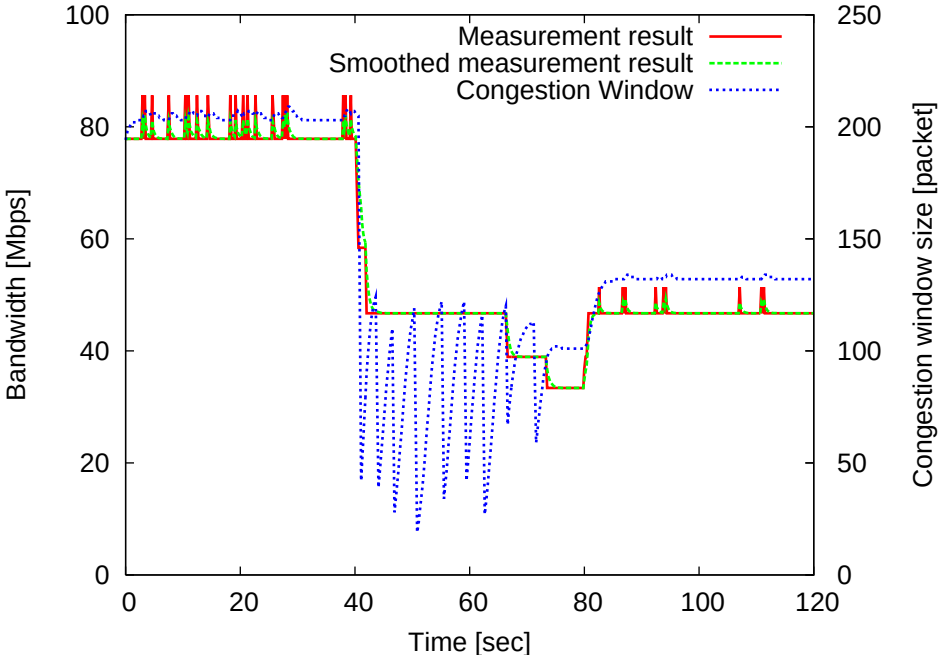
ウサイズも低くなるため，スループットが低下し，空き帯域を有効に使うことができない．

以上のことより，TCP Symbiosis のスループットは ImTCP の計測精度に大きく依存するということがいえる．また，この環境における TCP Symbiosis の実験においては，ImTCP は計測精度が最も高い計測ストリーム数を 10，ストリームあたりのパケット数を 20 にするのが適切であると考えられる．

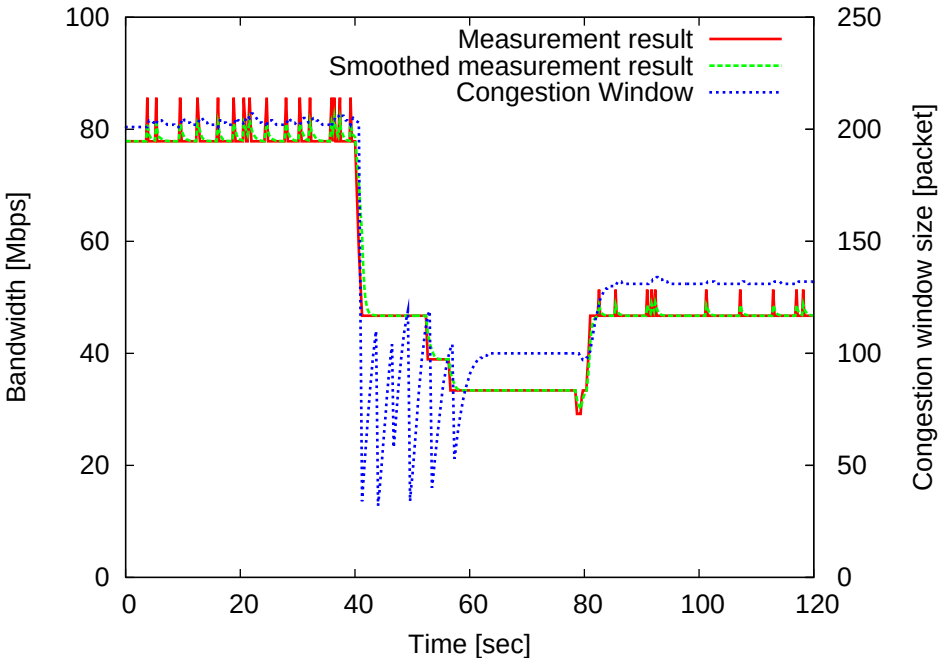
3.3 帯域遅延積に対するスケーラビリティ

シミュレーションによる性能評価の結果，TCP Symbiosis は帯域遅延積に対して高いスケーラビリティをもつことが示されている．実装実験においても TCP Symbiosis が帯域遅延積に対して高いスケーラビリティを示すために，Dummynet で RTT を変化させて TCP Reno とのスループット比較を行った．本節における実験では，クロストラヒックとして UDP トラヒックを前節と同じように流し，TCP Symbiosis および TCP Reno のスループットを受信側で tcpdump [32] を用いて監視する．実験においては，Dummynet により送受信端末間の RTT を 30，60，100 および 150 ms と変化させた．ImTCP の計測ストリーム数は 10，1 ストリームあたりのパケット数は 20 とする．

実験結果を図 6 に示す．また，ImTCP のストリーム数を 10，1 ストリームあたりのパケット数を 20 としたときの ImTCP の 1 秒あたり計測頻度を図 5 に示す．RTT が 30 ms のとき (図 6 (a)) にはスループットは安定しているものの，Reno に比べて優位性が少ない．しかし，RTT が大きくなるにつれてウィンドウサイズの増加にかかる時間が TCP Reno は大きく増

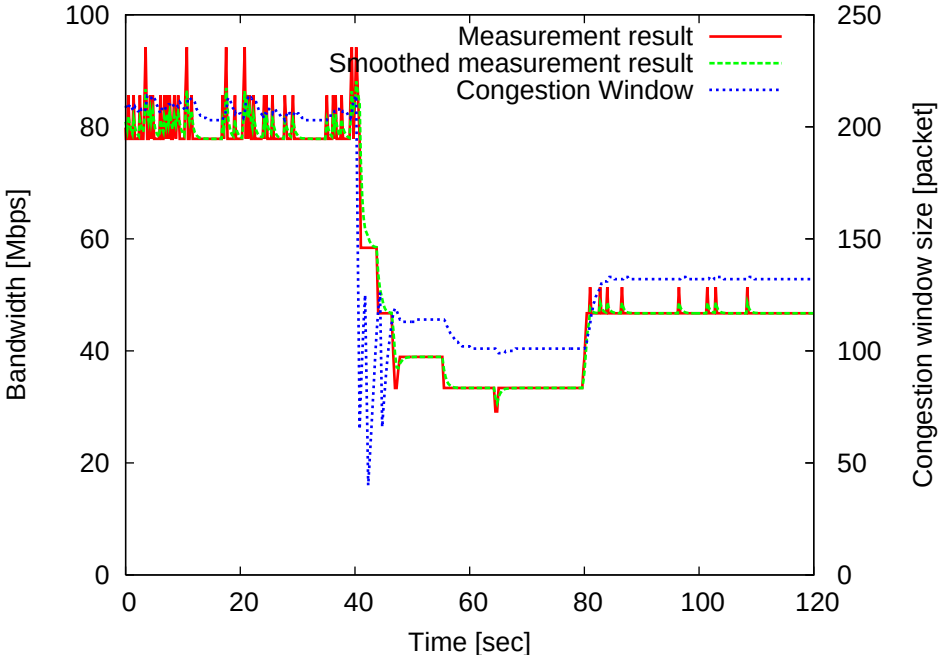


(a) ストリーム数 5，パケット数 10

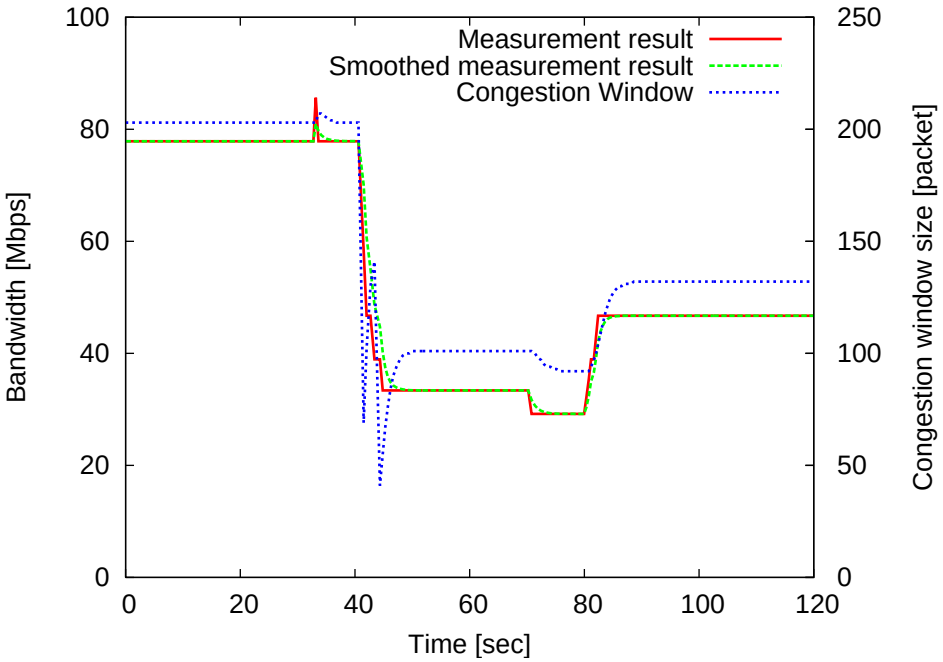


(b) ストリーム数 10，パケット数 10

図 4: ImTCP のパラメータ設定の影響

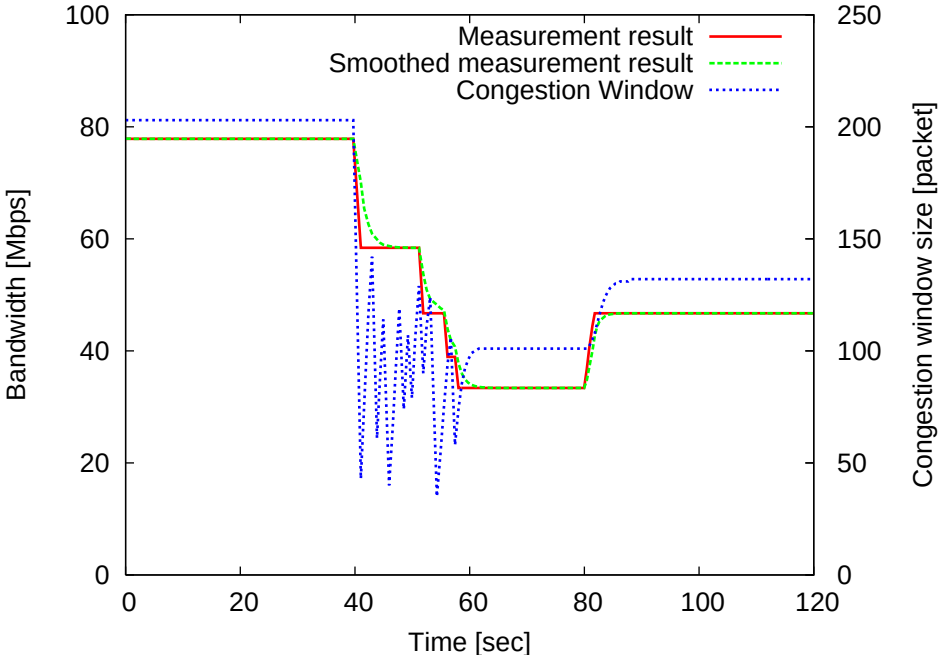


(c) ストリーム数 5，パケット数 20

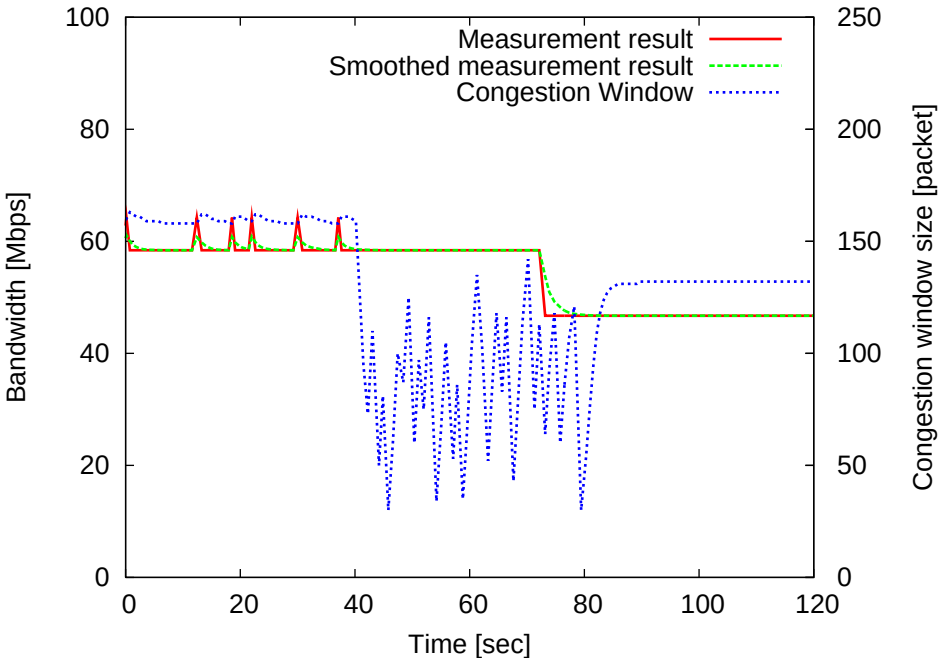


(d) ストリーム数 10，パケット数 20

図 4: ImTCP のパラメータ設定の影響



(e) ストリーム数 10，パケット数 30



(f) ストリーム数 15，パケット数 20

図 4: ImTCP のパラメータ設定の影響

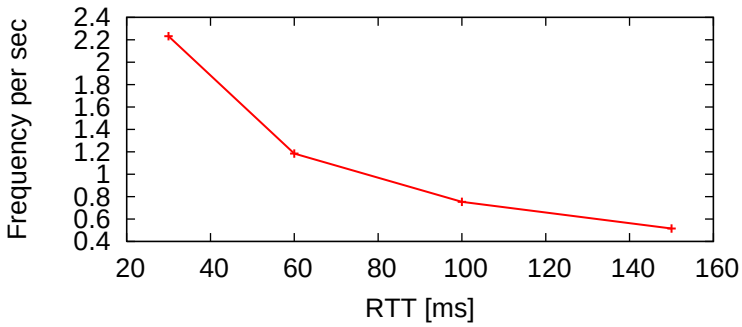
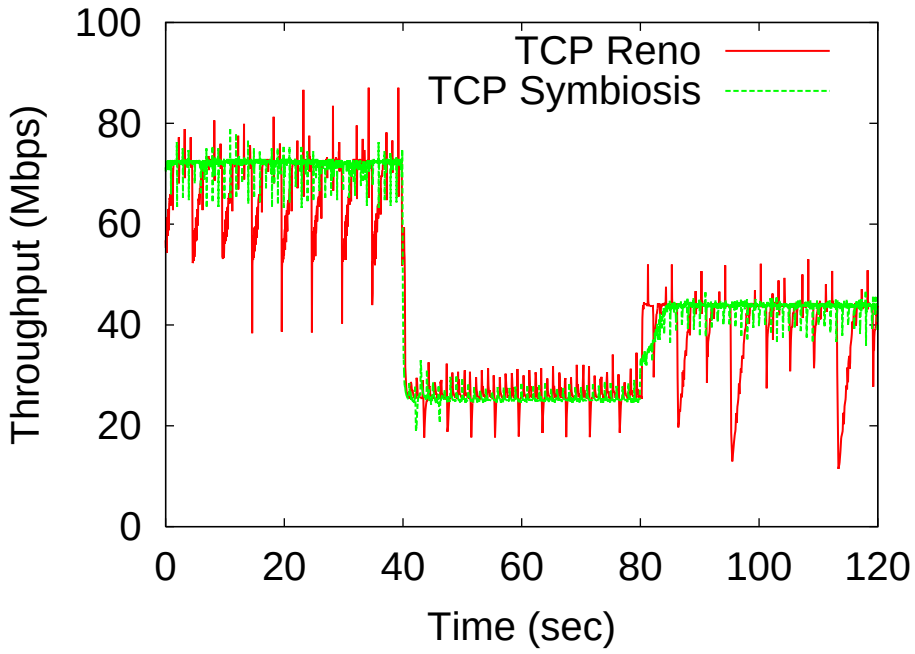


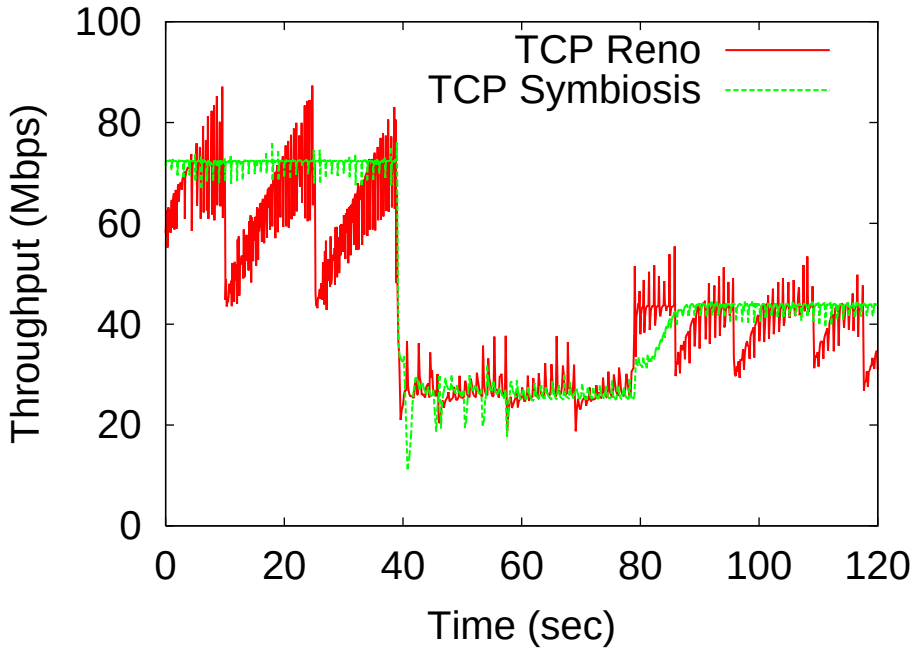
図 5: 異なる RTT での ImTCP の計測頻度

えるのに対して，Symbiosis においては，それほど差異はない．これは，帯域遅延積の増加に対してウィンドウサイズの増加にかかる時間が TCP Reno は線形的に増えるのに対して，TCP Symbiosis は対数的に増加するからである (3.3 参照) .

また，RTT が増加するにつれて ImTCP の計測頻度が低くなり (図 5) ，計測精度も低下する．そのため，RTT が 100, 150 ms の場合 (図 6 (c), (d)) において，利用可能帯域が 30 Mbps となる (40〜60sec) とき，ImTCP の計測が正しくなるまでに時間がかかるためにパケット廃棄が発生している．しかし，ImTCP の計測精度が低下しているにもかかわらず，TCP Symbiosis のスループットが TCP Reno のスループットよりも上回っていることがわかる．したがって，帯域遅延積の大きい環境においては，ImTCP の計測精度が低下しても，TCP Symbiosis の有効性は大きいといえる．

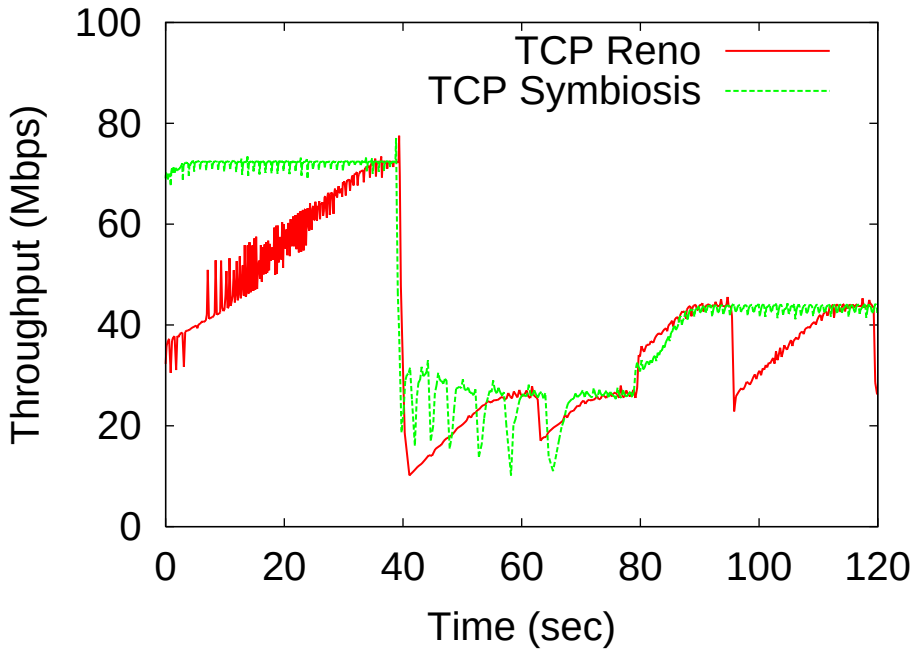


(a) RTT 30 ms

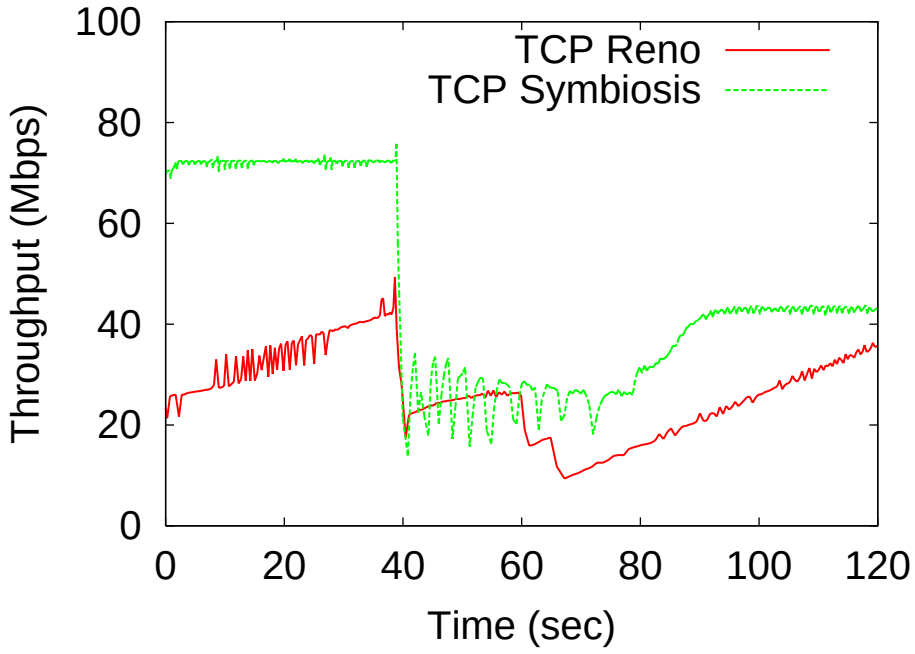


(b) RTT 60 ms

図 6: 異なる RTT での TCP Symbiosis と TCP Reno とのスループット比較



(c) RTT 100 ms



(d) RTT 150 ms

図 6: 異なる RTT での TCP Symbiosis と TCP Reno とのスループット比較

4　インターネット環境における性能評価

本章では，インターネット公衆回線を用いてデータ転送実験を行い，TCP Symbiosis の性能評価を行う. 実験ネットワークとして，帯域遅延積の小さい環境である大阪大学 (大阪府) と NEC (東京都) 間，および帯域遅延積の大きい環境である大阪大学と University of California, Los Angeles; UCLA (米国カリフォルニア州) 間でそれぞれ実験を行った．実験ネットワークは図 7 のように構成し，送信側端末およびクロストラヒック発生用の端末を大阪大学に，受信側端末を他方に設置する．Sender より，TCP Symbiosis あるいは TCP Reno のトラヒックを，Background Traffic Generator よりクロストラヒックを受信側に流す.

実験環境は以下のとおりである.

- 送受信端末間には iperf [30] によるトラヒックを流す.
- 受信側は Delayed ACK オプションを無効にするツールを用いる.
- TCP スループットが送受信端末のソケットバッファサイズに制限を受けることを避けるために，送信側端末のソケット送信バッファ，受信側端末のソケット受信バッファの値を帯域遅延積より十分大きな値に設定する.
- 1 秒間に発生するシステムのタイマー割り込み回数 (=HZ) は 20000 であり，計測の最小粒度は 50 μs となる.

4.1　帯域遅延積の小さい環境における実験

4.1.1　大阪-東京間のネットワーク環境

送受信端末およびクロストラヒック発生用端末の端末のスペックを表 3 に示す.
また，大阪-東京間のネットワークの性質として以下のことがわかっている.

- ping により，最小の RTT は 16 ms となっていることから，送受信端末間の往復伝播遅延時間が 16 ms である.
- 帯域遅延積の小さい環境にするため, ボトルネックリンク帯域は 40 Mbps に制限された環境において実験を行っている.

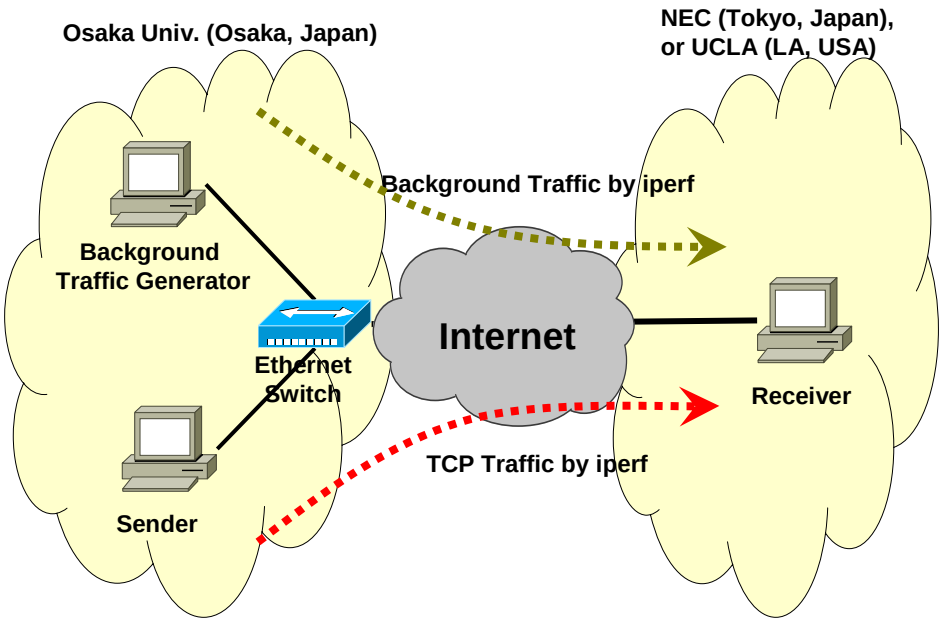


図 7: インターネット公衆回線における実験環境

表 3: 大阪-東京間の実験環境を構築する端末の性能

	送信側端末 (実験トラヒック用)	送信側端末 (クロストラヒック用)	受信側端末
CPU	Intel Pentium 4 3.40 GHz	Intel Xeon 3.60 GHz	Intel Xeon 2.66 GHz
Memory	1,024 MB	2,048 MB	1,024 MB
OS	Fedora Core 5	Linux	Linux

表 4: 実験における時間経過にともなうクロストラヒックの変化 (大阪-東京間)

t（時間）	$0 \leq t < 20$	$20 \leq t < 40$	$40 \leq t < 60$	$60 \leq t < 80$	$80 \leq t < 100$
クロストラヒックの帯域 [Mbps]	0	20	30	10	0
予測される利用可能帯域 [Mbps]	40	20	10	30	40

4.1.2 TCP Symbiosis の性能評価

クロストラヒックとして UDP トラヒックを表 4 のように流し，ImTCP の計測結果および TCP Symbiosis の制御による輻輳ウィンドウサイズの変動を確認した．また，スループットを受信側で tcpdump を用いて監視する．実験結果を図 8 に示す．また，同様の実験を TCP Reno で行った場合に監視たスループットとの比較を図 9 に示す.

まず，ImTCP の精度について考察する．図 8 より，20-60 sec において，ImTCP は計測結果が安定しており，精度も高い．これは，20-60 sec の利用可能帯域が小さいことに起因する．ImTCP は粒度の関係で計測値の値が小さいほど計測できる値が細かくなるため，20-60 sec においては計測結果が安定しやすい．反対に，0-20 sec, 60-80 sec においては 20-60 sec の区間に比べて計測できる値が粗いため，計測結果が安定していない．また，実際の利用可能帯域の値に比べて小さい計測結果が得られている.

次に，TCP Symbiosis の性能評価を行う．実験環境はボトルネックリンク帯域が 40 Mbps, 最小 RTT が 16 ms という帯域遅延積の小さい環境である. この環境においては，TCP Reno を用いた場合においてもリンク帯域を十分に使い切ることができることが知られている．一方，TCP Symbiosis は ImTCP の計測結果が正確な場合 (20-60 sec) には TCP Reno と同程度のスループットを発揮でき，リンク帯域を使いきれているということができる．しかし，ImTCP の計測結果が実際の利用可能帯域より小さい場合 (0-20 sec, 60-100 sec) には TCP Reno に比べてスループットが劣ってしまう.

したがって，帯域遅延積が小さい環境においては ImTCP の計測精度が低下すると，TCP Symbiosis のスループットも低下するということがいえる.

4.2 帯域遅延積の大きい環境における実験

4.2.1 大阪-米国カリフォルニア間のネットワーク環境

送受信端末およびクロストラヒック発生用端末の端末のスペックを表 5 に示す.

また，大阪-米国カリフォルニア間ネットワークの性質として以下のことがわかっている.

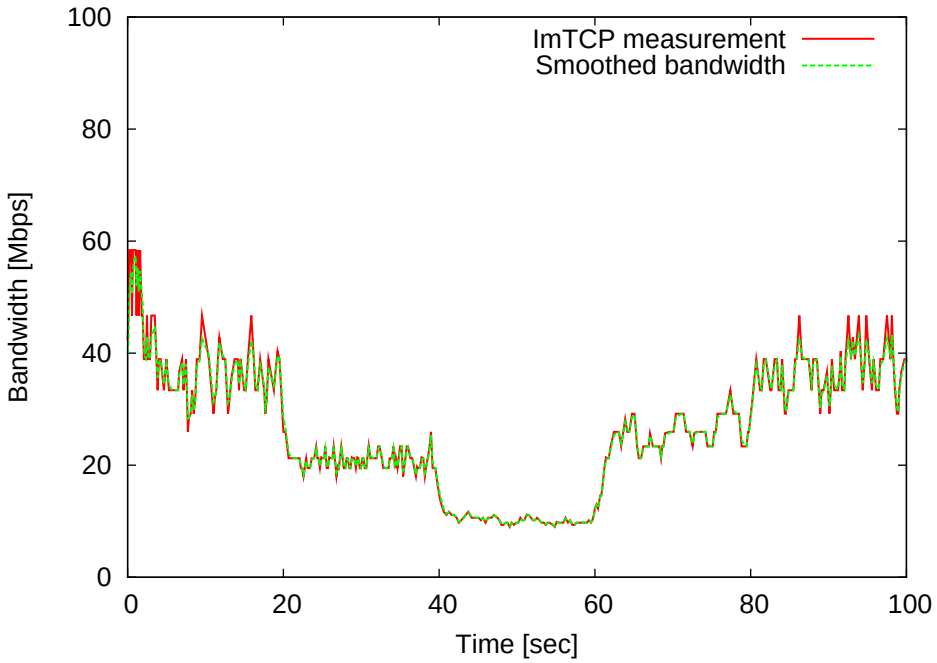


図 8: ImTCP の計測精度 (大阪-東京間)

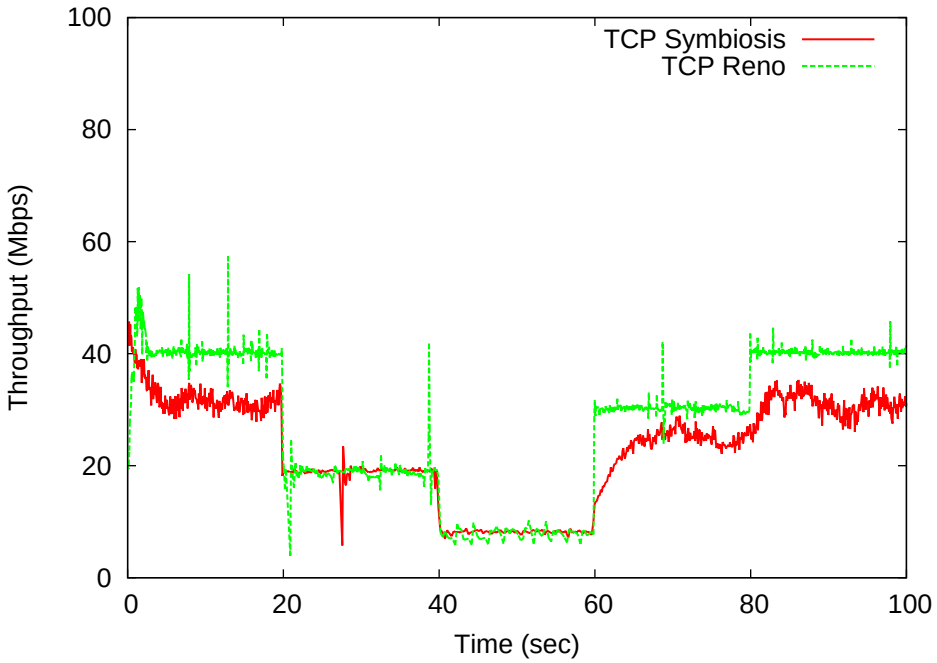


図 9: TCP Symbiosis と TCP Reno とのスループット比較 (大阪-東京間)

表 5: 大阪-米国カリフォルニア間の実験環境を構築する端末の性能

	送信側端末 (実験トラヒック用)	送信側端末 (クロストラヒック用)	受信側端末
CPU	Intel Pentium 4 3.40 GHz	Intel Xeon 3.60 GHz	Intel Xeon 3.06 GHz
Memory	1,024 MB	2,048 MB	1,024 MB
OS	Fedora Core 5	Fedora Core 5	Fedora Core 6

表 6: 実験における時間経過にともなうクロストラヒックの変化 (大阪-米国カリフォルニア間)

t（時間）	$0 \leq t < 20$	$20 \leq t < 40$	$40 \leq t < 60$
クロストラヒックの帯域 [Mbps]	15	65	45
予測される利用可能帯域 [Mbps]	80	30	50

- ping により，最小の RTT は 118 ms となっていることから，送受信端末間の往復伝播遅延時間が 118 ms である．
- 最小の計測粒度が 50 μs であることより，ImTCP の計測できる最大値は 120 Mbps である．また，本ネットワーク環境においては，100 Mbps を超えるスループットを得ようとした場合に，ネットワーク内において帯域制限を受けることがわかっている．ボトルネックリンク帯域が 100 Mbps になるようにネットワークを構成している．

4.2.2 TCP Symbiosis の性能評価

クロストラヒックとして UDP トラヒックを表 6 のように流し，ImTCP の計測結果および TCP Symbiosis の制御による輻輳ウィンドウの変動を確認した．また，スループットを受信側で tcpdump を用いて監視する. 実験結果を図 10 に示す. また，同様の実験を TCP Reno で行った場合に得たスループットとの比較を図 11 に示す.

まず，図 10 より，ImTCP の計測精度について考察する．実験室ネットワークの実験より，RTT が大きくなると，計測頻度が低くなり，また精度も下がることがわかっている．この実験環境では，RTT が 118 ms と大きいために計測頻度が低くなり，ImTCP の計測精度が低下している．しかし，TCP Symbiosis のスループットに着目すると，TCP Reno より高いスループットでデータ転送を行っていることがわかる．これは，ImTCP の計測誤差が大きい 20-60 sec の間でパケット廃棄が周期的に発生しているものの，ウィンドウサイズの増加が TCP Reno に比べて大きいためである．

表 7: 利用可能帯域が 80Mbps の時のスループット，平均および標準偏差

回数	1 回目	2 回目	3 回目	4 回目	5 回目	平均	標準偏差
TCP Symbiosis	78.1	78.1	78.3	78.3	78.3	78.22	0.22
TCP Reno	49.6	49.1	44.5	38.8	45.8	45.56	8.70

したがって，帯域遅延積の大きい環境においては，ImTCP の誤差があるとしても，TCP Symbiosis は TCP Reno に比べて高いスループットでデータ転送を行うことができるということができる．

4.2.3 利用可能帯域を固定した場合のスループットの向上率

最後に，TCP Symbiosis のスループットが TCP Reno に比べてどの程度向上するかを検証した．利用可能帯域の値が 80 Mbps となるようにクロストラヒックを 20 Mbps を流し，TCP Reno あるいは TCP Symbiosis のコネクション 1 本を用いてデータ転送を行ったときのスループットを 1 分間監視するという試行を 5 回行った．結果として，表 7 に各方式の各試行で得られたスループット値およびその平均と標準偏差を示す．表から，TCP Symbiosis が TCP Reno の約 2 倍のスループットを極めて安定的に獲得していることがわかる．また，そのうちの各方式の 1 回目の試行のスループット変化を時間ごとにプロットしたものを図 12 に示す．

TCP Reno のスループット変化を見ると，データ転送開始直後にスループットが低下していることがわかる．これは，データ転送開始直後のスロースタートフェーズが原因である．また，同じ現象が TCP Symbiosis にも見られる．これは，TCP Symbiosis は ImTCP の計測結果が得られるまで TCP Reno と同じように動作するので，スロースタートによって大きくウィンドウサイズが増加するためである．しかし，ImTCP の計測結果が得られると，TCP Symbiosis は計測結果を用いてウィンドウサイズを更新するため，TCP Reno に比べると早い段階でウィンドウサイズが増加に転じる．さらにその後はすばやくウィンドウサイズを大きくし，10 秒未満でウィンドウサイズがほぼ理想値に収束し，高いスループットが得られている．一方 TCP Reno を用いた場合，ウィンドウサイズの増加速度が小さいため，1 分間という時間では空き帯域を十分使いきれるほどウィンドウサイズが増加しない．表 7 に示した，TCP Reno のスループットの標準偏差が大きいのは，パケット廃棄が発生するタイミングによって，ウィンドウサイズが大きく変化するためである．

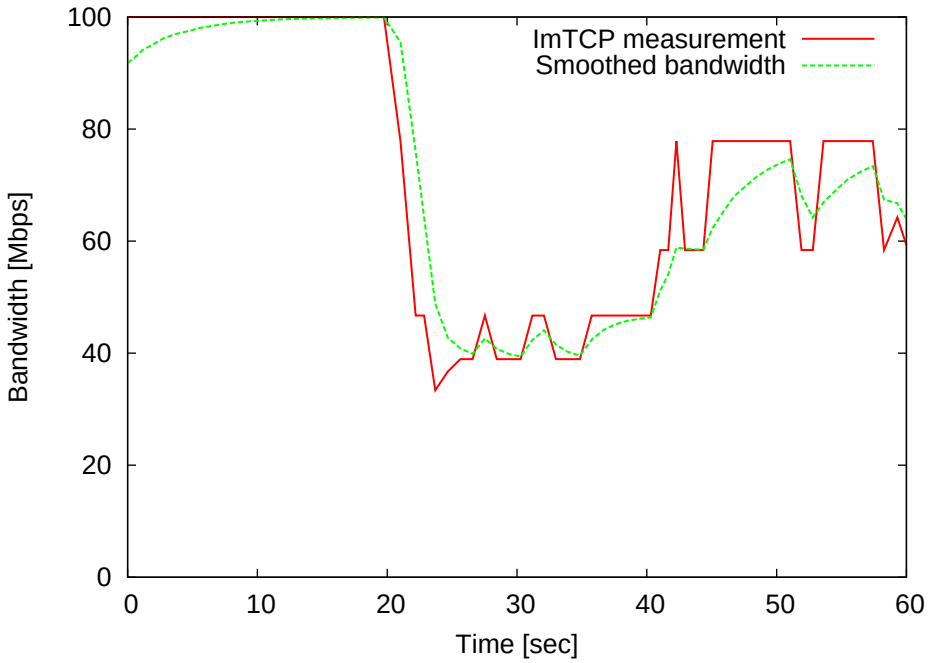


図 10: ImTCP の計測精度 (大阪-米国カリフォルニア間)

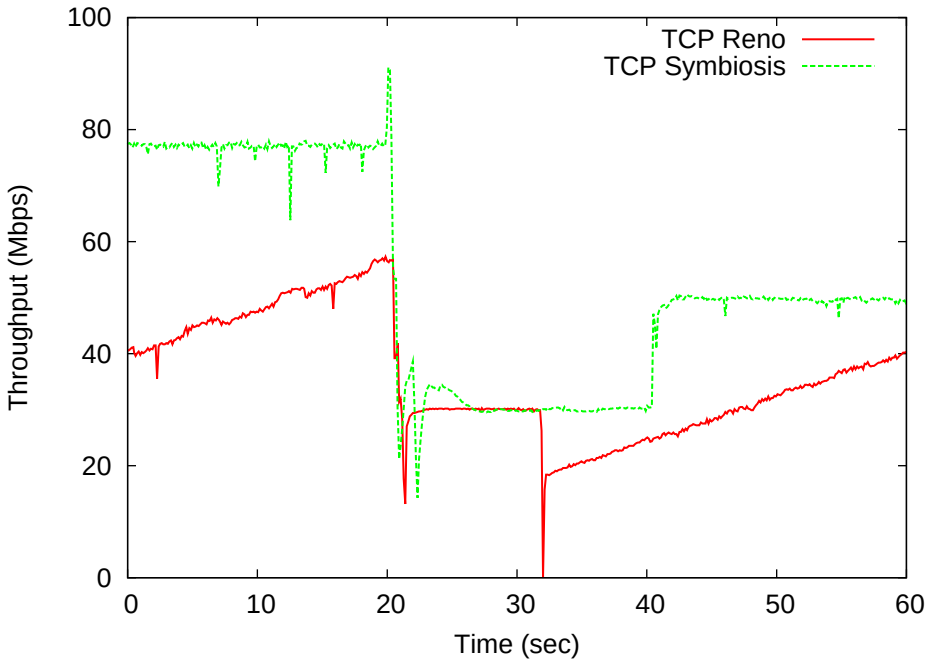


図 11: TCP Symbiosis と TCP Reno とのスループット比較 (大阪-米国カリフォルニア間)

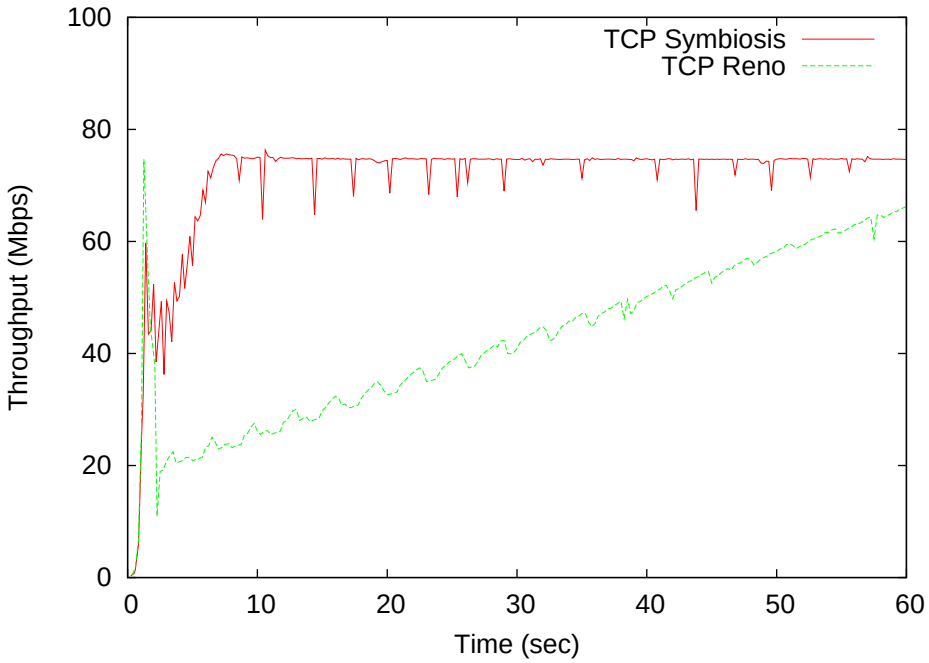


図 12: 利用可能帯域が 80 Mbps であるときの TCP Symbiosis と TCP Reno とのスループット比較 (大阪-米国カリフォルニア間)

5　おわりに

本報告では，TCP の輻輳制御方式である TCP Symbiosis の性能評価を実験室ネットワークおよびインターネット公衆回線上での実験を通して行った．その結果，TCP Symbiosis は帯域遅延積の大きいネットワークにおいて ImTCP の粒度による計測誤差が生じたとしても，既存の方式である TCP Reno より高いスループットが得られることがわかった．また，ImTCP の計測結果が正確な限りは周期的なパケット廃棄を避けることができ，帯域遅延積の小さいネットワークにおいても TCP Reno と同程度のスループットを得ることを示した．

本報告により，TCP Symbiosis の制御が ImTCP の計測結果に強く依存しており，その精度次第ではスループットが低下してしまうという問題が明らかとなった．ImTCP の計測誤差の原因は粒度が関係しており，計測制度の向上は難しいと考えられる．そこで，今後の課題として，ImTCP の計測結果の効率的なフィルタリングにより TCP Symbiosis のスループット向上を目指したい．

また，本手法と [17, 18] の手法とのスループット比較についても行いたい．

謝辞

本報告を終えるにあたり，御指導，御教授を頂きました中野博隆教授，村田正幸教授に深く感謝致します．本報告の作成にあたり，終始熱心に指導および助言をして頂きました長谷川剛助教授に深く感謝致します．的確な助言を頂きました大阪大学サイバーメディアセンター笹部昌弘助手に心から感謝致します．また，TCP Symbiosis の実験評価に関し，様々な助言を頂きました NEC システムプラットフォーム研究所の村瀬勉様，下西英之様および浜崇之様に心から感謝致します．並びに，実験および本報告の執筆するにあたり，大阪大学の津川知朗氏および山根木果奈氏に的確な助言を頂きました．ここに記して謝意を示します．最後に，日頃から相談に答えて頂きました中野研究室および村田研究室の皆様方に心より御礼申し上げます．

参考文献

[1] W. R. Stevens and G. R. Wright, *TCP/IP Illustrated Volume 2 : The Implementation (Addison-Wesley Professional Computing Series)*. Addison-Wesley Pub (Sd), 1994.

[2] J. B. Postel, "Transmission control protocol," *Request for Comments 793*, Sept. 1981.

[3] V. Jacobson, R. Bradon, and D.Borman, "TCP extensions for high performance," *Requenst for Comments 1323*, May 1992.

[4] M. Allman, H. Balakrishman, and S. Floyd, "Enhancing TCP's loss recovery using limited transmit," *Request for Comments 3042*, Jan. 2001.

[5] E. Blanton, M. Allman, and L. W. K. Fall, "A conservatic selective acknowledgement (SACK) - based loss recovery algorithm for TCP," *Request for Comments 3517*, Apr. 2003.

[6] D. M. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Journal of Computer Networks and ISDN Systems*, pp. 1–14, June 1989.

[7] S. Shenker, L. Zhang, and D. D. Clark, "Some observations on the dynamics of a congestion control algorithm," *ACM Computer Communication Review*, vol. 20, pp. 30–39, Oct. 1990.

[8] J. C. Hoe, "Improving the start-up behavior of a convergestion control scheme of TCP," *ACM SIGCOMM Computer Communication Review*, vol. 26, pp. 270–280, Oct. 1996.

[9] L. Guo and I. Matta, "The war between mice and elephants," *Technical Report BU-CS-2001-005*, May 2001.

[10] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla, "The impact of multihop wireless channel on TCP throughput and loss," in *Proceedings of IEEE INFOCOM 2003*, Apr. 2003.

[11] S. Floyd, "HighSpeed TCP for large congestion windows," *Request for Comments 3649*, Dec. 2003.

[12] T. Kelly, "Scalable TCP: Improving perfomance in highspeed wide area networks," in *Proceedings of PFLDnet 2003*, Feb. 2003.

[13] C. Jin, D. X. Wei, and S. H. Low, "FAST TCP: motivation, architecture, algorithms, performance," in *Proceedings of IEEE INFOCOM 2004*, Mar. 2004.

[14] L. S. Barkmo and L. L. Peterson, "TCP Vegas: End to end congestion avoidance on a global Internet," *IEEE Jounal on Selected Areas in Communications*, vol. 13, pp. 1465–1480, Mar. 1995.

[15] J. Mo, R. J. La, V. Anantharam, and J. Walrand, "Analysis and comparison of TCP Reno and Vegas," in *Proceedings of IEEE INFOCOM '99*, 1999.

[16] G. Hasegawa, K. Kurata and M. Murata, "Analysis and improvement of fairness between TCP Reno and Vegas for deployment of TCP Vegas to the Internet," in *Proceedings of IEEE ICNP 2000*, Nov. 2000.

[17] K. T. J. Song, Q. Zhang, and M. Sridharan, "Compound TCP: A scalable and TCP-friendly congestion control for high-speed networks," in *Proceedings of PFLDnet 2006*, Feb. 2006.

[18] H. Shimonishi, T. Hama, and T. Murase, "TCP-Adaptive Reno: Improving efficiency-friendliness tradeoffs of TCP congestion control algrithm," in *Proceedings of PFLDnet 2006*, Feb. 2006.

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

[20] 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.

[21] 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.

[22] R. L. Carter and M. E. Crovella, *Measuring bottleneck link speed in packet-switched networks*. Tech. Rep. BU-CS-96-006, Boston University Computer Science Department, Mar. 1996.

[23] C. Dovrolis, P. Ramanathan, and D. Moore, "What do packet dispersion techniques measure?," in *Proceedings of IEEE INFOCOM 2001*, Apr. 2001.

[24] Cao Le Thanh Man, Go Hasegawa, Masayuki Murata, “ImTCP: TCP with an inline measurement mechanism for available bandwidth,” *Computer Communications Journal special issue of Monitoring and Measurements of IP Networks*, vol. 29, Issue 10, June 2006.

[25] Cao Le Thanh Man, Go Hasegawa, Masayuki Murata, “A simultaneous inline measurement mechanism for capacity and available bandwidth of end-to-end network path,” *IEICE Transactions on Communications*, vol. E89-B, pp. 2469–2479, Sept. 2006.

[26] Go Hasegawa and Masayuki Murata, “TCP Symbiosis: Congestion control mechanisms of TCP based on Lotka-Volterra competition model,” in *Proceedings of Workshop on interdisciplinary systems approach in performance evaluation and design of computer & communications systems (Inter-Perf 2006)*, Oct. 2006.

[27] R. L. Carter and M. E. Crovella, *Measuring bottleneck link speed in packet-switched networks*. Tech. Rep. BU-CS-96-006, Boston University Computer Science Department, Mar. 1996.

[28] J. D. Murray, *Mathematical Biology I: An Introduction*. Springer Verlag Published, 2002.

[29] L. Rizzo, “Dummynet.” available from `http://info.iet.unipi.it/~luigi/ip_dummynet/`.

[30] NARANR, “NLANR/DAST: Iperf 1.7.0 - The TCP/UDP Bandwidth Measurement Tool.” available from `http://dast.nlanr.net/Projects/Iperf/`.

[31] B. R., “Requirement for internet hosts – communication layers,” *Request for Comments 1122*, Oct. 1989.

[32] L. N. R. Group, “TCPDUMP public respository.” available from `http://www.tcpdump.org/`.