透過的クライアント監視による ネットワークシステム性能評価手法

斉藤 裕樹　　中所 武司

明治大学大学院 理工学研究科 基礎理工学専攻 情報科学系

{hiro-s,chusho}@cs.meiji.ac.jp

Performance Evaluation of Network Systems by Client Observation

Hiroki SAITO and Takeshi CHUSHO

Computer Science Course, Major in Science,
Graduate School of Science and Technology, Meiji University

概要

TCP/IP インターネット技術の普及に伴い，サーバへのアクセス速度や応答性能といったユーザの利便性を確保するための性能面でのネットワークサービスの品質が求められている．

本論文では，ユーザビリティの観点でのネットワークシステムの性能評価方式を提案する．この方式は，クライアントアプリケーションの監視を行う透過的監視層を導入し，クライアントとサーバの間でやり取りされる要求-応答単位の通信に注目することで性能を測定するものである．これにより稼働状態のシステムの性能を正確に測定できることを実験により示し，実装に基づき有効性を確認する．

1　はじめに

インターネットの展開とその技術を用いた企業，大学などのネットワークの増加と規模の拡大に伴い，データ転送が遅い，転送が中断されるなどのネットワークシステムの性能面での問題が顕在化している．スループットの保証や応答速度の改善といった，ネットワークサービスの性能を確保するネットワーク性能管理技術が望まれている [1, 2].

従来のネットワーク性能管理は，端末間の到達性の保証やリンクの性能確保を目的とし，主にネットワーク層の機器の管理を中心としたものであった．しかし，インターネットの主要プロトコルである TCP の性能はネットワーク層固有の特性に左右されることが知られている [3, 4] ．さらに，ネットワークシステムの性能は，帯域，ルータの処理能力，端末間のスループット，サーバシステムやクライアントシステムの性能などの多くの要因があり，ネットワーク管理者には技術的な知識，勘や経験などが必要とされてきた．アプリケーション利用時のアクセス速度，応答速度，データ転送速度といったユーザの直接感じるネットワークサービスの性能を，勘や経験に頼らず客観的に評価できるシステムが望まれている．

本論文では，ユーザの立場での性能を確保するため，監視層でクライアントアプリケーションの振る舞いを監視し性能管理を行う方式を提案する．

2 ネットワークシステムの性能評価

2.1 性能評価に求められる要件

ユーザの視点でのネットワークシステムの性能とはクライアントでの性能，すなわち終点アプリケーションでの性能である．本報告では終点アプリケーションの性能を評価するための汎用的な枠組みについて検討する．

このようなネットワークシステム性能評価に必要とされる要件を以下に示す．

1. **終点アプリケーションでのユーザの感じるスループットや応答性能を測定できること．**
   ネットワークシステムの性能向上の目標は，終点アプリケーションでの性能を高めることである．システムの全体的な性能は途中の経路の状況と相関があるとは限らない．したがって，途中の経路に注目するのではなく，終点アプリケーション自体の性能で評価する必要がある．
2. **通信経路のデータリンクの種類によらず測定できること．**
   インターネットでは Ethernet, FDDI, ATM, ISDN, 専用線, フレームリレーなど様々なデータリンクが利用されている．TCP/IP 技術はこれらデータリンクの上位層のプロトコルの集合である．したがって，ネットワークシステムの性能測定はデータリンクの種類に関わらず行えなければならない．
3. **特定のアプリケーションに依存しないこと．**
   インターネットでは多くのアプリケーションが利用されているが，性能評価システムは，特定のアプリケーションやアプリケーションプロトコルを前提としたものではなく，これらの性能を共通の方法で測定できる必要がある．
4. **既存のアプリケーションへの変更を必要としないこと．**
   現在稼働しているシステムやアプリケーションソフトウェアに変更を加えて測定機能を付加することはコストが大きく，また変更が容易でないアプリケーションも多い．このため，これらに変更を加えることなく，最小限の設定変更のみで測定が行える必要がある．
5. **実際に稼働しているシステムに対して測定が行えること．**
   計算機上のシミュレーションは，全ての性能要因をシミュレートすることが難しく，ベンチマークによる測定では実際の稼働状態の再現が難しく，結果はベンチマーク環境に限定されたものである．したがって，利用されているシステムの稼働状態での測定が行えることが望ましい．
6. **ネットワークの通信性能だけではなくサーバシステムやクライアントシステムの性能が測定できること．**
   ネットワークシステムの性能には，サーバやクライアントの計算機自体の性能も影響をおよぼすため，これらの性能にも注目しなければならない．
7. **複数のクライアントでの性能を一元的に管理できること．**
   ネットワーク上のサーバは多くのクライアントから利用されるため，異なったネットワーク環境にある複数クライアントの性能を測定し，性能情報を収集し一括管理が行えるシステムが必要である．

2.2 従来の性能測定手法

これまで，ネットワークの性能評価を目的として様々な試みが行われてきた．以下に，従来の性能評価方式とその問題点を示す．

1. **サーバのアクセス記録から解析**
   アクセス記録から，アクセス数，転送データ量やリクエストの処理時間を統計的に計算する方法である．しかし，記録がアプリケーション固有の形式であり，アプリケーションごとに解析プログラムが個別に必要となる．また，アクセス記録に性能に関する記録を行わないものやアクセス記録を行わないアプリケーションも多い．
   さらに，アクセス記録から得られる内容はサーバ自体の性能であり，クライアントの性能やネットワーク回線に関する性能ではないため，実際にユーザが感じる性能とは隔たりがある．
2. **ネットワーク利用率や遅延時間の測定**

SNMP[5, 6] を用いたネットワーク機器の遠隔監視として広く用いられている方法である．中継ノードやリンクの利用率を測定するため，特定のアプリケーションプロトコルに依存しない．しかし，インターネットの主要プロトコル TCP の性能は，利用率や遅延に関わらずネットワーク層の特性によって悪化することが知られている．

このように，ネットワークシステムの性能は，利用率や遅延だけではなく，ネットワーク層の特性，端末間のスループット，サーバシステムやクライアントシステムの性能などに大きく左右される．

3. **ベンチマーク**

   これまで SPECweb[7], WebStone, ttcp や DBS [8] などのいくつかのベンチマークツールが開発されてきた．

   これらは正確な性能計測が行える反面，実際のユーザの振る舞いや運用形態に適したベンチマーク条件を設定することが困難である．また，アプリケーション性能を測定するためにはベンチマークツールをアプリケーションごとに用意する必要がある．

4. **パケット収集**

   パケット収集のためのアナライザは特定のデータリンクにしか対応しておらず，また snoop, tcpdump などのツールはトラヒック共有型のネットワークでしか利用できない．

5. **ユーザへのヒアリング**

   実際にユーザへシステムの使用感をたずねる方法である．評価基準があいまいであり，技術的な指針となりにくい．

以上のことから，異なる環境下でも稼働状態のネットワークシステムに対して測定を可能とするような性能評価システムが必要だと考えられる．

3 監視層による性能評価

3.1 クライアントアプリケーションの監視による性能評価

本報告では，クライアントアプリケーション監視のための透過的監視層を用いたネットワークシステムの性能評価方式を提案する．この方式の目的は，実際に稼働しているネットワークシステムのユーザから見た性能を正確に把握することによって，性能面でのユーザビリティの確保のための汎用的な枠組みを提供することである．

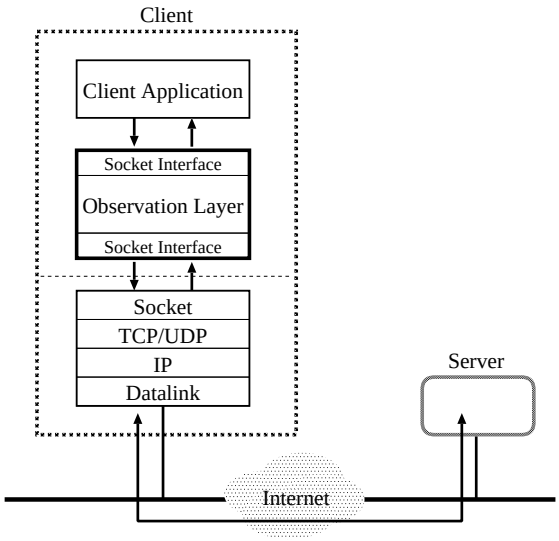

図 1: 監視層による測定の動作原理

提案方式はクライアントアプリケーションの挙動を監視することによって性能測定を行うものである．動作原理を図 1 に示す．クライアントアプリケーションからネットワーク上のサーバへの通信は，すべて OS 内のネットワーク機能のシステムコールを利用して行うため，システムコールの内容を監視するレイヤを新たにを導入することにより，クライアントアプリケーションのネットワーク利用状況を把握することができる．

監視層は，クライアントアプリケーションの挙動を直接監視することができるので，ユーザの感じる性能に近い値を計測することが可能である．また，監視層を BSD Socket, TLI, Winsock といった OS のネットワーク機能と同一のインタフェースとすることにより，アプリケーションから透過的に扱えるのでアプリケーションの改造を必要としない．さらに，複数クライアントの監視を行うために監視層の計測を遠隔操作できるようなアーキテクチャとする．

これらにより 2.1 節の要求を全て満たすことができる．具体的には，監視層は稼働中のクライアントア

プリケーションを監視するものなので，終点アプリケーション性能を計測することが可能であり，標準的なシステムコールのインタフェースを用いるため既存のアプリケーションへの適用が容易であり，またこれらのシステムコールはデータリンク非依存である．以上によって 1 から 6 までの機能を実現できる．さらに，遠隔監視用の管理マネージャによって 7 が実現できると考えられる．

3.2 要求-応答単位の通信モデル

ここで，監視層による性能測定方式が対象とするクライアント-サーバ間の通信モデルについて述べる．HTTP, SMTP, POP3 をはじめとする多くのインターネットプロトコルでは，サーバとクライアントの通信モデルとして，要求と応答を単位とした以下の手順に従っている．

1. コネクションの確立
   (`connect()` システムコール)
2. リクエスト 1 の送信
   (`write()` システムコール)
3. リクエスト 1 の処理結果の受信
   (`read()` システムコール)
4. リクエスト 2 の送信
5. リクエスト 2 の処理結果の受信
   ……
6. リクエスト n の送信
7. リクエスト n の処理結果の受信
8. コネクションの切断
   (`close()` システムコール)

このような通信モデルを「要求-応答単位の通信モデル」と呼ぶ．

HTTP[9] を例にとり，要求-応答単位の通信モデルにおける通信フローを図 2 に示す．クライアントは，サーバに GET, PUT などのリクエストを送信し，次にコンテンツ情報を受けとる．なお，HTTP1.1 における Keep-alive 機能を用いた場合もリクエストと応答の対が増えるのみでモデルは共通である．

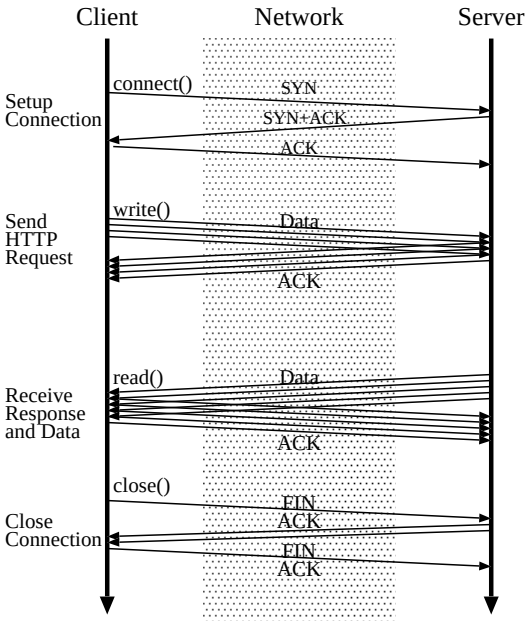

図 2: HTTP の通信フロー

3.3 性能指標

監視層では，クライアントと OS 間のシステムコールによって様々な性能指標を得ることができる．要求-応答単位の通信モデルを基にした性能指標を図 3 に示す．監視層での計測内容は T_1 から T_7 の時間および送受信されたデータ量である．これにより測定可能な性能指標を以下に示す．

コネクション継続時間 T_c (T_1-T_7 間)
: `connect()` システムコールの開始から `close()` システムコールの終了までに要した時間．処理開始から処理終了までの合計時間．この値は，ユーザの感じる処理時間に最も近いと考えられる．

コネクション確立時間 T_e (T_1-T_2 間)
: TCP コネクションを確立するため `connect()` システムコールに要した時間．ネットワークの RTT (Round Trip Time) と端末の TCP/IP 機能の性能を示す．

リクエスト送信時間 T_s (T_3-T_4 間)
: リクエストデータを送信するための `write()` システムコールに要した時間．リクエストデー

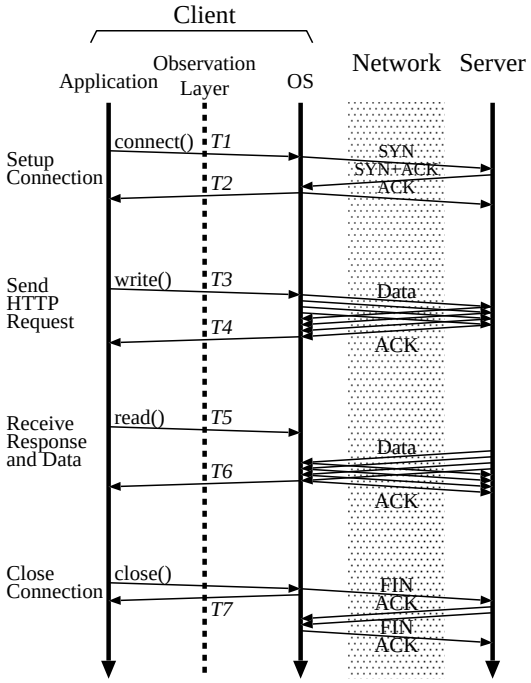

図 3: 監視層で計測される性能指標

タの送信に要した時間.

リクエスト処理および応答受信時間 T_r (T_5-T_6 **間**)
リクエストに対する処理結果を受信するための `read()` システムコールに要した時間．サーバがリクエストの処理に要した時間とデータ転送に要した時間の合計である.

リクエストデータ送信量 D_s
リクエストデータを送信するための `write()` システムコールでサーバに送られたデータの大きさ.

応答データ受信量 D_r
応答データを受信するための `read()` システムコールでサーバから送られてきたデータの大きさ.

リクエストデータ送信レート R_s (D_s/T_s)
クライアントからサーバに送られたリクエストデータの送信レート.

応答データ受信レート R_r (D_r/T_r)
サーバからクライアントに送られた応答データの受信レート.

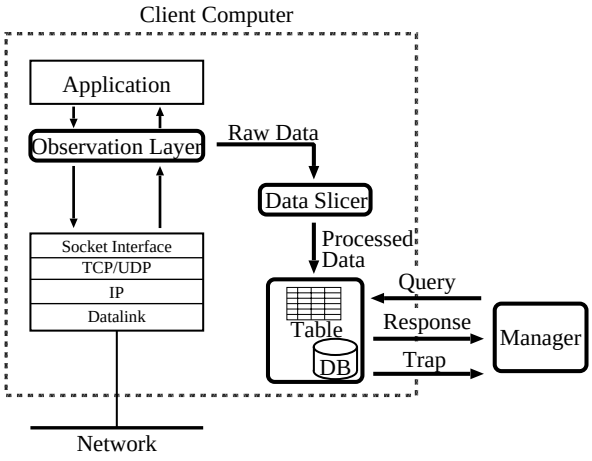

図 4: システム構成

全データ処理レート R_a ($(D_s + D_r)/T_c$)
通信に要した全ての時間に対する全転送データ量の割合．扱うデータ量に応じたユーザの感じる性能の指標となる.

4 監視層による性能評価システム

4.1 システム構成

監視層を用いた性能評価システムの構成を図 4 に示し，各部分について説明する.

Observation Layer アプリケーションと OS 間のやりとりを監視し，通信時間や転送データ量を測定する部分

Data Slicer 指定した条件によって測定データのフィルタリングを行い加工する部分

Database 加工済み測定結果を格納する部分

Manager データの収集，可視化，分析を行う管理マネージャ

4.2 動作手順

監視層のコネクション管理部分でネットワーク利用に関するシステムコールが監視され，コネクションの状態，T_1〜T_7 までの各タイムラインの時刻，転送データ量が記録される．次にデータスライサによって，条件に合わせたフィルタリングが行われ不要な

データを削除し，さらに統計分析などの数学フィルタの処理が行われ，計測結果がデータベースに格納される．

管理マネージャは計測対象システムから計測結果を受け取り，データの可視化，他クライアントの計測結果の比較などを行うことができる．将来，問題分析機能，過去のデータの蓄積機能，性能問題発生時に警告を行う機能など，より高度な管理機能を実現する予定である．

4.3 データ構造

性能測定を行うクライアントシステム内に計測データを格納するために一種のデータベースが用意される．現在の実装ではデータベースは 2 次元の表形式である．まず，監視アプリケーションごとにマトリクスが用意される．マトリクスは (通信先, ポート, T, D, R) を時系列に並べたものである．T, D, R は，監視層によって得られた 1 回のコネクションごとの性能指標 (各タイムラインでの時間，データ転送量，転送レート) を並べたベクトルである．

将来は通信先や時間帯別の比較など多観点での性能評価を容易にするため，任意の n 次元の表に加工できるようにデータスライサとデータベースを拡張することを検討している．

5 実装

クライアント側の監視システムの実装は ANSI C を用い Sun Solaris 上で行った．監視層は Socket インタフェースを持つ代理システムコールからなる共有ライブラリとして実装されている．動的リンク機能を利用し，アプリケーション実行時に共有ライブラリのプリロードを用い通常のシステムコールを監視層の代理システムコールで上書きすることにより，クライアントアプリケーションの変更や再コンパイルすることなく計測を可能にしている．また，このような動的リンク機能は主要な UNIX プラットフォームや Windows にも備わっているため，他のプラットフォームへの対応は容易であると考えられる．また，たとえこのような機能が備わっていないシステムでもアプリケーションを再コンパイルするのみでソースコードを変更する必要はない．

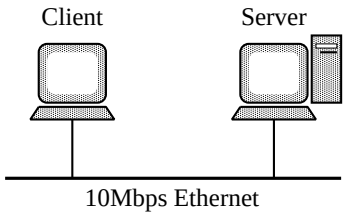

図 5: オーバヘッドの評価: 実験ネットワーク

アプリケーションが Socket インタフェースにアクセスすると，監視層の計測用の代理 Socket インタフェースの関数が呼び出される．代理 Socket インタフェースは，Socket の生成，コネクションの確立，切断，中断およびデータの送受信を解析し，T_1 より T_7 の各タイムラインの記録および転送データ量をカウントした上で，実際の OS 内の Socket インタフェースのシステムコールを起動する．現在は，コネクション単位に計測結果がデータスライサに渡され，逐次データベースに記録しているが，計測における CPU 負荷の低減のため，データスライサ以降の処理をバッチ的に行うことを検討している．

マネージャは現在シェルスクリプトおよび C で記述しており，データの可視化機能として Gnuplot へのインタフェースが実装されている．現在，性能測定対象の計算機と管理マネージャ間の通信は独自プロトコルであるが，将来 SNMP などの標準プロトコルに対応する予定である．

6 実験ネットワークへの適用評価

6.1 オーバヘッドの評価

本システムでは，データを転送するたびに時刻と転送量を記録している．この処理は CPU の負荷となり測定結果に影響をおよぼす可能性がある．負荷に対するスケーラビリティの評価のため，同条件におけるクライアントでのスループットをアプリケーション単体の場合と監視層を介した場合で比較し，測定のオーバヘッドを検証した．

図 5 に示すような 10Mbps の Ethernet で結ばれた HTTP サーバとクライアント間で表 1 に示す設定でデータ転送を行った．クライアントの ApacheBench

表 1: オーバヘッドの評価: 実験環境

サーバ	UltraSPARC 270MHz (Solaris2.6)
クライアント	PentiumII 350MHz (Solaris2.6)
サーバプログラム	Apache 1.3.4
クライアントプログラム	ApacheBench
データサイズ	102400byte

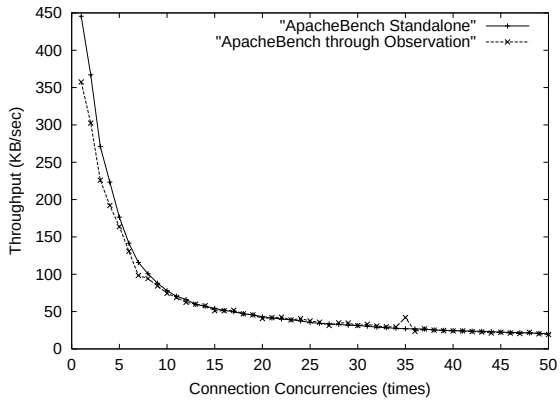

図 6: オーバヘッドの評価: 計測結果

にて，複数コネクションによるデータ転送を同時に実行し，1 コネクションあたりの平均のスループットを測定した結果を図 6 に示す.

結果から，実験環境では時間あたりのデータ転送量が最も大きくなる場合に最大 15%程度のオーバヘッドが生じている．しかし，大量のデータ転送を伴わない通常のインターネットで用いられるようなアプリケーションでは，オーバヘッドによる影響は実用上問題ないものと考えられる.

6.2 サーバ上のアクセス記録との比較

監視層でのコネクション継続時間 T_c とサーバのアクセス記録から分かるサーバ上での処理時間の差を比較するため，実験システム内での負荷環境下における性能計測実験を行った.

実験環境を図 7 に示す．表 2 の設定で同時コネクション数を徐々に増加させ，それぞれの方法で計測された負荷に対する処理時間の変化を計測した．計測結果を図 8 に示す.

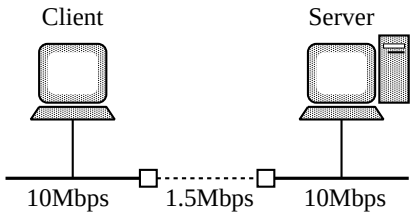

図 7: サーバアクセス記録との比較: 実験ネットワーク

表 2: サーバアクセス記録との比較: 実験環境

サーバ	UltraSPARC 270MHz (Solaris2.6)
クライアント	PentiumII 350MHz (Solaris2.6)
クライアントプログラム	ApacheBench
サーバプログラム	Apache 1.3.4
データサイズ	102400byte

サーバのアクセス記録は監視層と比較して最大約 28 秒，平均約 4.5 秒短い応答時間が計測された．これには次の原因が考えられる.

監視層で計測される T_c の値は，コネクションを確立する時間，HTTP での通信時間，およびコネクションを切断する時間の合計である．コネクションを確立・切断するためには少なくとも 3RTT の時間を要する．しかし，サーバのアクセス記録ではこれらの時間を計測することができない.

また，サーバアプリケーションからは OS のバッファリングによりデータが完全に送信しきれていない場合でも転送終了と見なされてしまう.

これらのため，サーバのアクセス記録では本来の所要時間より少ない値が計測されたと考えられる.

6.3 ボトルネックリンク

サーバとクライアントの間の通信容量の違いがアプリケーションの性能にどのような影響をおよぼす

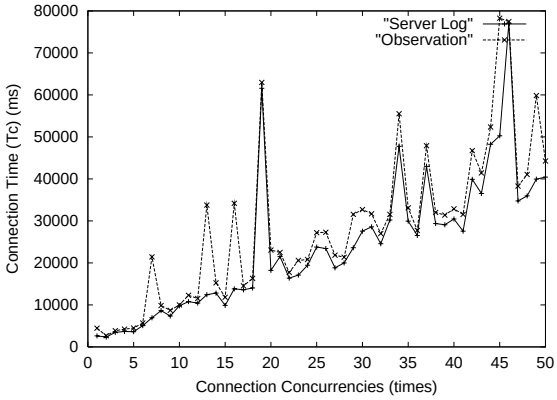

図 8: サーバアクセス記録との比較: 計測結果

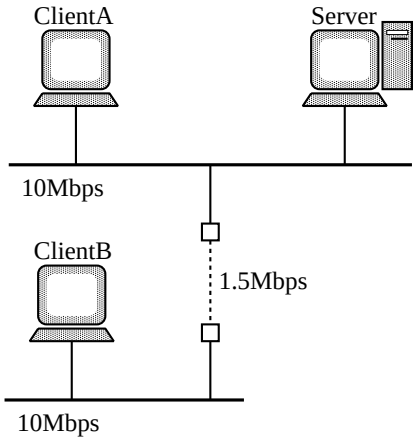

図 9: ボトルネックリンク: 実験ネットワーク

のかを比較するため，図 9 のような実験環境でボトルネックリンクの有無による性能の違いを計測した．

ネットワークの通信速度はボトルネックリンクを 1.5Mbps とし，それ以外の部分を 10Mbps とした．前節と同様に表 2 の設定で同時コネクション数を徐々に増加させ負荷に対する処理時間の変化を計測した．通常のネットワークに接続されているクライアント A とボトルネックリンクの先にあるクライアント B で計測されたコネクション継続時間 T_c の比較を図 10 に示す．

結果から，単純なネットワークトポロジーにおいてネットワーク帯域が性能に与える影響を明確に示しているといえる．

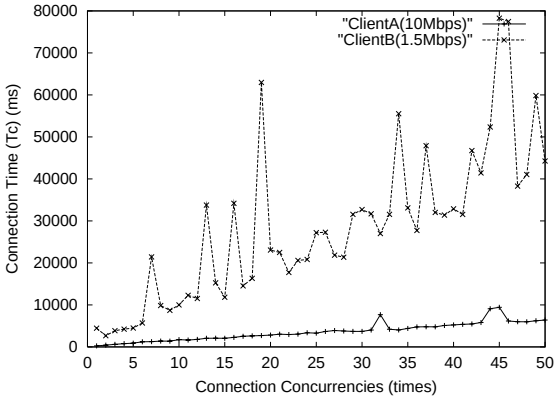

図 10: ボトルネックリンク: 計測結果

7 おわりに

本報告では，クライアントとサーバの間でやり取りされる要求-応答単位の通信に注目し，クライアントアプリケーションの性能の監視を行う監視層を導入することによって，ネットワークアプリケーションの性能測定方式を提案した．

この方式の有効性を確認するため，既存の性能評価の方法と比較し機能面で優れていることを示した．また，測定処理が与えるオーバヘッドの影響は小さなものであることを実験によって確認した．また，実際の運用形態に近い実験ネットワークで，これまで正確に把握することが難しかった稼働状態のシステムの性能を明確に測定できることを示した．

今後は，負荷やネットワークの規模に対するスケーラビリティの評価および，測定結果のより精密な分析手法や多クライアントの遠隔集中管理機構について検討を進めていく．

謝辞

本論文の作成に際して貴重なご意見をいただきました，奈良先端科学技術大学院大学情報科学センターの砂原秀樹助教授に感謝いたします．

参考文献

[1] 斉藤 裕樹, 中所 武司: イベント駆動型管理戦略に基づくネットワーク性能管理手法の提案, 情報

処理学会 マルチメディア通信と分散処理研究会報告 99-DPS-91, pp.97-102 (1999).

[2] 斉藤 裕樹, 中所 武司: 透過的モニタリング層によるネットワーク性能管理方式の提案, 情報処理学会 第 59 回全国大会講演論文集 (3) pp.471-472 (1999).

[3] Mark Allman, Chris Hayes, Hans Kruse, Shaen Osterman: TCP Performance over Satellites Links, Proc. of the 5th International Conference on Telecommunications Systems (1997).

[4] V. Jacobson, R. Braden, D. Borman: TCP Extentions for High Performance, RFC1323 (1992).

[5] Leinwand, A.: Accomplishing Performance Management with SNMP, Proc. of INET'93 (1993).

[6] D. Harrington, R. Presuhn, B. Wijnen: An Architecture for Describing SNMP Management Frameworks, RFC2271 (1998).

[7] SPECweb99 Benchmark, `http://www.spec.org/osg/web99/` (1999).

[8] 村山 公保, 門林 雄基, 山口 英: TCP 性能評価システム DBS の構築, コンピュータソフトウェア, Vol.15, No.2, pp.24-37 (1998).

[9] R. Fielding, J. Getty, J.Mogul, H. Frystyk and T. Berners-Lee: Hyper Text Transfer Protocol - HTTP/1.1, RFC2068 (1997).

[10] Apache HTTP server project, `http://www.apache.org/` (1999).