複数の CUBIC TCP コネクションにおける RTT 公平性の改善

神津智樹[†1]　秋山友理愛[†1]　　山口実靖[†2]

TCP は現在のインターネットにおいて標準的に用いられているトランスポート層のプロトコルである．高遅延環境下での通信性能には，OS の TCP 実装の動作が大きな影響を与える．多くの OS に実装されてきた古典的な TCP アルゴリズムである TCP Reno ではネットワーク帯域を十分に使い切ることができないという問題が指摘されている．そのため，Linux OS には高遅延環境でも高い性能を提供できる CUBIC TCP が実装されている．

本研究では，CUBIC TCP の輻輳ウィンドウ回復時間に着目し，RTT が異なる環境における CUBIC TCP コネクション間の公平性の評価を行い，その課題を示す．そして，RTT により K (CUBIC TCP における輻輳ウィンドウ回復時間) を調整し RTT 公平性を改善する手法を紹介し，複数コネクション環境における本手法の評価結果を示す．そして多コネクション環境にて十分な公平性を実現できないことを示し，その理由と改善手法に関する考察を示す．

Improving RTT Fairness on CUBIC TCP of Multiple Connections

TOMOKI KOZU[†1]　YURIA AKIYAMA[†1]　　SANEYASU YAMAGUCHI[†2]

1. はじめに

TCP は現在のインターネットにおいて標準的に用いられているトランスポート層プロトコルであり，その実装にはネットワークの輻輳制御が組み込まれている．従来の TCP 実装では輻輳制御アルゴリズムとして TCP Reno[1]が使用されてきた．しかし，古典的な TCP アルゴリズムである TCP Reno では高遅延・広帯域のネットワーク環境においてネットワーク帯域を十分に使い切ることができない問題が指摘されており[2][3][4]，BIC TCP[5]，CUBIC TCP[6]，Compound TCP[7]などの TCP Reno に変わる新しい高速 TCP が多数提案されている．Linux OS には CUBIC TCP と呼ばれる高速 TCP が実装されている．CUBIC TCP では，RTT 公平性が改善されたと主張されているが，まだ不十分であることが確認されている[8][9]．また，TCP 公平性に関する研究としてシミュレーションに基づく研究が行われてきている[10][11]が，実際の OS に搭載されている TCP 実装を用いての研究は十分に行われておらず，実 OS の実 TCP 実装を用いての評価や考察も行う必要があると考えられる．

本稿では，遅延時間の異なる CUBIC TCP コネクションが混在する環境において性能と公平性の評価を行う．そして，RTT を考慮して CUBIC TCP の輻輳ウィンドウ回復時間調整をし，RTT 公平性を向上する手法[12]を紹介する．続いて，本手法を複数コネクション環境において評価し，多コネクションかつ RTT 差の大きい環境にて十分な公平性を実現できないことを示す．最後に，本手法が多コネクション環境にて十分な公平性を実現できない理由とその改善方法について考察する．

†1 工学院大学大学院　工学研究科　電気・電子工学専攻
†2 工学院大学　工学部　情報通信工学科

2. 関連研究

2.1 TCP 輻輳制御アルゴリズム

過剰なパケットを送出しネットワークの輻輳を招くことを避けるために，TCP 実装には輻輳制御アルゴリズムが搭載されている． TCP によるパケット送出量はこの輻輳制御アルゴリズムにより制限されており，TCP の通信性能はこのアルゴリズムに大きな影響を受ける．OS により様々な TCP が実装されおり，輻輳制御の仕方が異なる．

TCP の輻輳制御手法は主に，ロスベース手法，遅延ベース手法，両者を組み合わせたハイブリッド型の手法に分類することができる．

ロスベース手法はパケットロスの検出に基づき輻輳ウィンドウを制御する手法である．通常時は確認応答を受信するたびに輻輳ウィンドウを増加させ，パケットロス検出時に輻輳ウィンドウを大幅に減少させる．従来の TCP である TCP Reno がロスベース手法であり，代表的なロスベース手法の高速 TCP に BIC TCP[5]や，CUBIC TCP[6]がある．

遅延ベースの輻輳制御アルゴリズムは，RTT の増減にあわせて輻輳ウィンドウを変化させる手法である．ロスベース手法の様に輻輳が発生してから速度を減少させるのではなく，RTT の増加からネットワークの混雑状況を推定し，輻輳が発生する前に速度を減少させるため安定した通信速度が期待できる．欠点としてロスベース手法とネットワークを共有したときに得られる性能が低くなってしまうということが指摘されている[13]．代表的な遅延ベース手法に TCP Vegas[14]がある．

ハイブリッド型手法はロスベースと遅延ベースの両者を組み合わせた手法であり，代表的なハイブリッド型手法の TCP に Windows Vista 以降の Windows 系 OS に標準で搭載

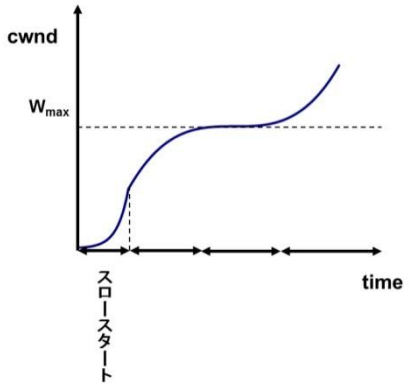

図 1. CUBIC TCP の輻輳ウィンドウの推移

されている Compound TCP[7]がある．

本研究では，サーバ OS として多く使われている Linux OS の標準 TCP である CUBIC TCP に焦点を当て考察を行う．

2.2 CUBIC TCP

CUBIC TCP[6]は，BIC TCP のスケーラビリティを維持しながら，TCP-Fairness，RTT-Fairness，制御手法の複雑さを改善した高速 TCP である．CUBIC TCP は Linux2.6.19 以降で標準の TCP として搭載されている．

CUBIC TCP では，BIC TCP のバイナリーサーチを用いて利用可能帯域を検索するアルゴリズムを式(1), (2)のような 3 次関数を用いた制御によって実現している．その輻輳ウィンドウの推移は図 1 のようになる．

$$cwnd = C(t-K)^3 + W_{max} \quad (1)$$

$$K = \sqrt[3]{\frac{W_{max}\beta}{C}} \quad (2)$$

ここで，$cwnd$ は輻輳ウィンドウサイズ，t はパケットロス検出時からの経過時間，W_{max} はパケットロス検出時の輻輳ウィンドウサイズ，C は増加幅を決めるパラメータ，β はパケットロス検出時のウィンドウサイズ減少幅を表している．通常，C には 0.4 が，β には 0.2 が用いられている．

CUBIC TCP では，上記のようにパケットロス検出時からの経過時間を用いて輻輳ウィンドウの値を定めている．これは，RTT の影響を強く受ける Ack の受信を輻輳ウィンドウサイズの増加処理から排していることを意味し，これにより RTT-Fairness が向上することが期待できる．加えて，BIC TCP の低遅延環境で輻輳ウィンドウサイズを急速に成長させすぎる問題もこれにより解決している．

また，TCP Reno を用いた場合に得られる輻輳ウィンドウサイズを式(3)により計算し，現在のウィンドウサイズが計算値よりも小さい場合はその計算値を輻輳ウィンドウサイズとして採用している．これにより，TCP-Fairness の向上を目指している．

$$cwnd = W_{max}(1-\beta) + 3\frac{\beta}{2-\beta}\frac{t}{RTT} \quad (3)$$

また，ボルトネックを共有するフローが帯域を公平に分け合うまでの収束時間を短縮するため，パケットロスを検出した時の輻輳ウィンドウサイズ値が前回ロスを検出した時の値を下回っている場合，新たな W_{max} は次式のように設定される．

$$W_{max} = cwnd \times (2-\beta)/2 \quad (4)$$

以上のようなしくみにより，CUBIC TCP は高いスケーラビリティ，RTT-Fairness，TCP-Fairness を目指している．

しかし，CUBIC TCP のウィンドウ制御は式(2)の K に依存しているため，以下に示すように RTT 公平性が低くなることも考えられる．K は，式(1)における 3 次関数の変曲点までの時間であり，t=K において，輻輳ウィンドウサイズが前回のパケットロス検出時の値と等しくなる．よって K が小さいほど短い時間で輻輳ウィンドウサイズが回復することとなり，高い性能が得られやすくなる．一般に RTT が大きい通信ほど，より大きな輻輳ウィンドウサイズが必要となる．しかし，式(2)より，前回のパケットロス時の輻輳ウィンドウサイズが大きいほど K の値が大きくなり，高い性能が得られづらいことが分かる．よって，RTT が大きい通信ほど輻輳ウィンドウサイズが大きくなり，結果 K の値が大きくなり通信性能が低くなりやすいと予想される．これは，RTT 公平性を低下させる原因になると考えられる．

2.3 TCP の公平性の研究

TCP の公平性向上に関して様々な研究が行われている．RTT 公平性に関する論文として以下の様な研究が行われている．Floyd らは，RTT 公平性達成のために，$a \times r^2$ の一定のウィンドウ増加(constant rate)アルゴリズムを提案している[15][16]．Henerson らは，一定の増加(constant rate)が劇的に RTT 公平性を改善することを示した[17]．Marfia らは，TCP の RTT 公平性が ACK ベースのメカニズムに起因すると示唆して，TCP Libra と呼ばれる新しい TCP アルゴリズムを提案した[18]．小倉らは RTT 公平性を改善するための HRF TCP という名の新しい TCP アルゴリズムを提案した[19]．CUBIC TCP の RTT 公平性についての研究として鈴川らの文献[20]がある．これによると，CUBIC TCP はウィンドウサイズの小さなフローが大きなフローから帯域を獲得することが難しいとされており，CUBIC TCP の輻輳ウィンドウが収束するのに時間がかかるとされている．鈴川らは，パケットロス時の減少幅である β と増加幅を決める c の調整を行い，CUBIC TCP の輻輳ウィンドウが収束するまでの時間を短くする手法を提案している．

異なる TCP 同士の公平性向上に関して以下の様な研究が行われている．逸身らはルータ内のキュー長を観測し，キュー長が大きく変動した際にパケットを廃棄することで公平性を改善させる手法を提案している [10]．また，長谷川らは RED を用いる公平性改善の研究として，バックボーンルータにおける RED の動的閾値制御方式を提案している[11]．当該研究では，RED のパラメータを輻輳の状況に応じて動的に変化させている．文献[21]では，ブリッジを

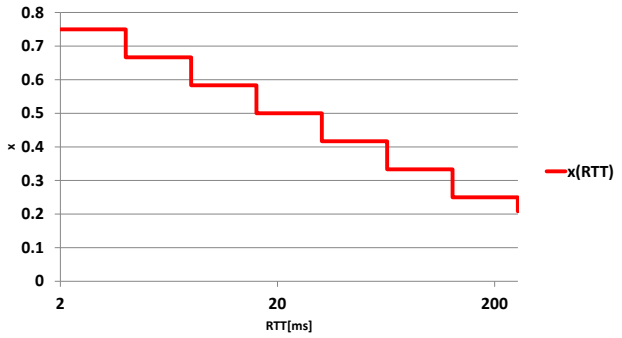

図 2　文献[8][9]に示された x(RTT)

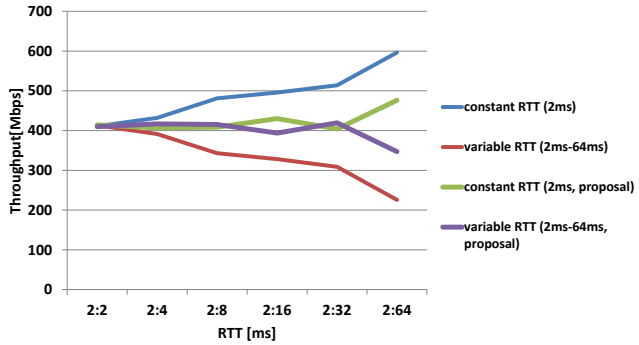

図 3　文献[8][9]に示された評価結果

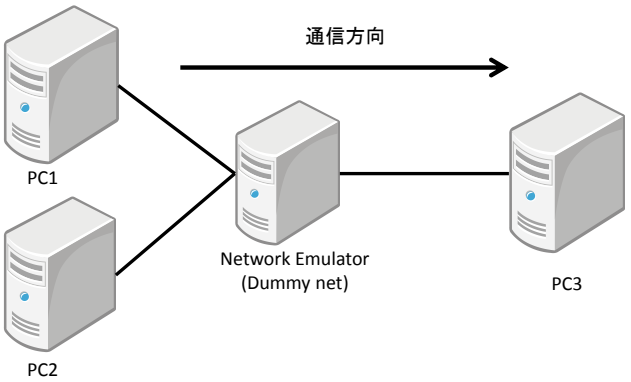

図 4 実験のネットワーク構成

表 1　計算機使用

CPU	Intel CelronG540, 2.4[GHz]
メモリ	2[GB]
OS	(PC1, PC2,) Linux 2.6.32.27 (PC3) Linux 2.6.35.6 (Network Emulator) FreeBSD 8.2

通過するパケットを観察し，帯域を多く使用しているコネクションの推定を行い，優先的にパケットを破棄する手法が提案されており，その実装を用いた評価結果が示されている．

2.4 既存手法（経験則に基づく K の調整）

また，文献[8][9]では，異なる RTT 同士の CUBIC TCP が競合すると，RTT が高いコネクションのスループットが低くなり RTT 公平性が低くなることが示されており，CUBIC TCP の K を調整する手法により公平性が改善できることが示されている．当該手法では，K を次式のように調整し，RTT が大きいコネクションの輻輳ウィンドウのサイズが短い時間で回復できるように制御している．式 5 の $x(RTT)$は図 2 の通りである．図 3 に，初期状態の CUBIC TCP と当該手法適用時の CUBIC TCP の性能を示す. 初期状態では，RTT 公平性が低いが，当該手法により RTT 公平性を大きく改善できることがわかる．ただし，当該手法では，K をユーザが経験則に基づき，調整する必要があり，この自動化が需要な課題となっている．

$$K = \sqrt[3]{\frac{W_{max}\beta}{C}} \times x(RTT) \qquad (5)$$

2.5 既存手法（RTT の 3 乗根に基づく K の調整）

前節の手法はユーザの経験則に基づく最適化が必要であり，これを改善した手法として式(6)のように RTT の 3 乗根により K を調整する手法[][]がある．

$$K = \sqrt[3]{\frac{W_{max}\beta}{C}} \times \frac{1}{\sqrt[3]{RTT}} \qquad (6)$$

本手法の詳細は 4 章にて述べる．

3. CUBIC TCP の RTT 公平性

3.1 CUBIC TCP の RTT 公平性の評価

本章では，実ネットワーク上で RTT 公平性（遅延時間の異なる通信間の性能公平性）についての評価を行う．図 4 のネットワークを構築し，2 つの CUBIC TCP コネクションが混在する環境における通信速度を netperf[22]を用いて測定した．PC1，PC2，PC3，ネットワークエミュレータの使用は表 1 の通りである．ネットワークエミュレータは人工的にネットワーク遅延時間を発生させる装置であり，FreeBSD Dummynet を用いて構築した．ネットワーク機器は全て 1Gigabit Ethernet に対応している．

通信は，PC1-PC3 間と，PC2-PC3 間で同時に行い，PC1 と PC2 を送信者，PC3 を受信者とした．PC1-PC3 間と PC2-PC3 間で netperf の接続を確立し，同時通信時の通信速度を測定した．両コネクションの通信は同時に行われ，両者はネットワークエミュレータから PC3 までのネットワークを共有している．ネットワークエミュレータにより PC1-PC3 間の RTT を 2ms から 64ms に，PC2-PC3 間の RTT を 2ms から 128ms に変動させて測定を行った．それぞれの受信ウィンドウサイズは 16MB に設定した．

実験結果を図 5 から 8 に示す．横軸の値はネットワークエミュレータにより設定した PC2-PC3 間の往復遅延時間である．左の縦軸はスループット，右側の縦軸は Fairness Index[23]である．Fairness Index は 1 に近いほど RTT 公平性が高いことを示している．図 5 から図 8 の結果より，RTT が小さい通信が RTT の大きい通信の性能を上回り，CUBIC TCP の RTT 公平性が低いことが確認できた．

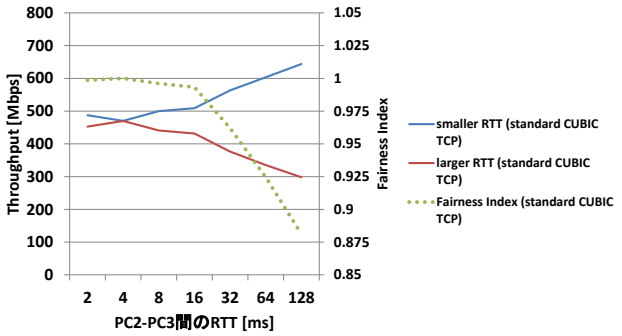

図 5　評価結果（PC1-PC3 間の RTT=2ms）

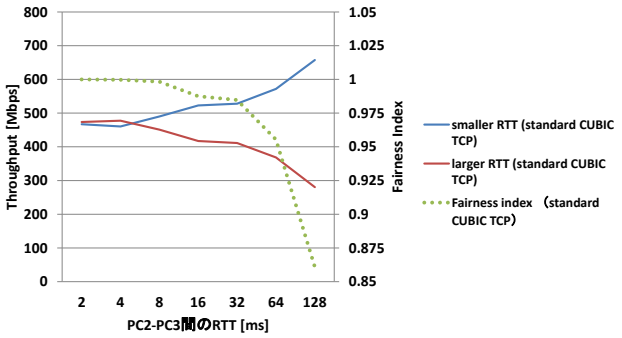

図 6　評価結果（PC1-PC3 間の RTT=4ms）

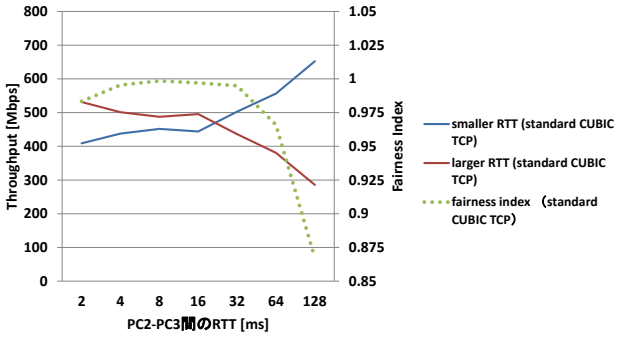

図 7　評価結果（PC1-PC3 間の RTT=16ms）

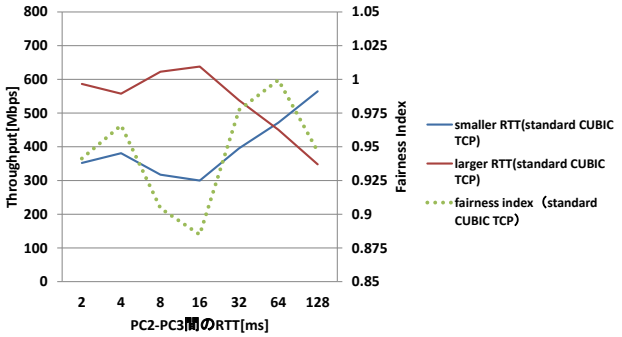

図 8　評価結果（PC1-PC3 間の RTT=64ms）

3.2　不公平の原因

通常のネットワークでは，ルータは Tail Drop を使用しており，全ての接続が同時に輻輳やパケットロスを引き起こす．これはグローバルシンクロナイゼーションと呼ばれている．図 9 に異なる K を持つ 2 つのコネクション接続時の輻輳ウィンドウの推移を示し，は大きな輻輳ウィンドウを持ち，　は小さい輻輳ウィンドウを持つ．図 9 では timeA

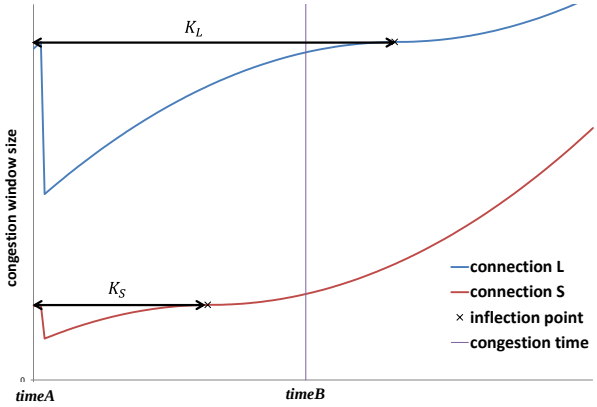

図 9　輻輳ウィンドウの推移

と timeB で輻輳とパケットロスが生じる．2.2 章で述べたように，K の決定式は　の値に依存する．そのため，輻輳ウィンドウの大きいコネクションは K が大きくなり，輻輳ウィンドウの回復に時間がかかる．次の輻輳が起きるのは，前回のパケットロス時の輻輳ウィンドウの合計と等しくなる timeB である．conn(S)は輻輳ウィンドウが増加しており，conn(L)は輻輳ウィンドウが減少している．すなわち，輻輳ウィンドウの大きなコネクションは，大きな K を持ち，他のコネクションより回復が遅れ，結果として輻輳ウィンドウが減少する．そのため，大きな RTT のコネクションは大きな輻輳ウィンドウ値が必要であるが，大きな輻輳ウィンドウ値を保つことができないこととなる．

4.　RTT の 3 乗根に基づく K の調整

本章で RTT を考慮して K を調整し，RTT 公平性を向上させる手法を紹介する．K の決定式を式(6)に示すように改善する．

$$K = \sqrt[3]{\frac{W_{max}\beta}{C}} \times \frac{1}{\sqrt[3]{RTT}} \qquad (6)$$

2 章で述べたとおり，CUBIC TCP の性能はパケットロスから輻輳ウィンドウが回復するまでの時間 K に依存しており，t=K となると輻輳ウィンドウが以前(パケットロス検出時)の値まで回復し，更に使える帯域を探し輻輳ウィンドウサイズを急激に上昇させる．K が小さいと，CUBIC TCP の輻輳ウィンドウサイズは短い時間で回復する．本手法では RTT が大きな通信ほど K が小さくなるように改変しており，これにより公平性が向上すると期待される．

文献[8][9]では，K の調整を経験則に基づいて行っていたが，本手法では上の RTT の式により統一的に扱う．

一般に，十分な性能を得るためには BDP(Bandwidth Delay Product)[2]の輻輳ウィンドウサイズが必要であるとされている．すなわち，必要な輻輳ウィンドウサイズは RTT に比例する．よっては RTT に比例すると予想される．もし，W_{max} が RTT に比例していれば，式(6)の K は一定値となり，全てのコネクションにとって公平となる．もし，W_{max}が比例

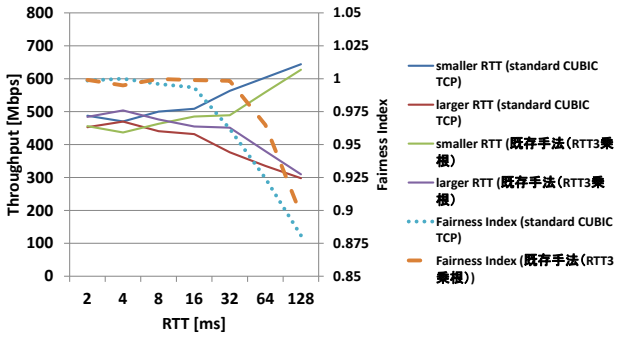

図 10　評価結果（PC1-PC3 間の RTT=2ms）

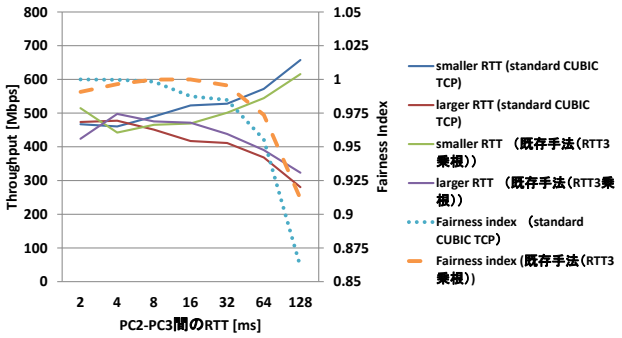

図 11　評価結果（PC1-PC3 間の RTT=4ms）

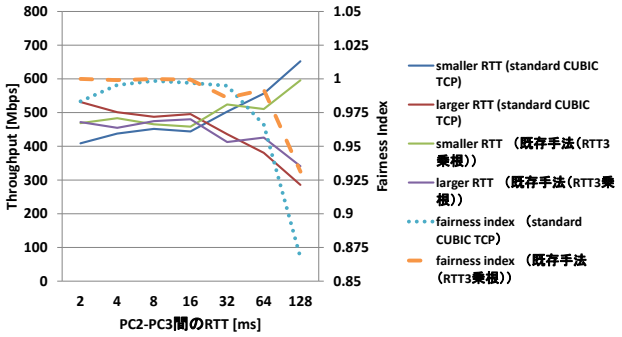

図 12　評価結果（PC1-PC3 間の RTT=16ms）

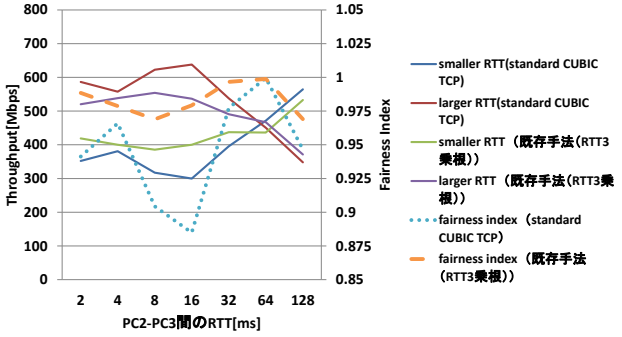

図 13　評価結果（PC1-PC3 間の RTT=64ms）

未満であった場合（すなわち，RTT が大きなコネクションの輻輳ウィンドウが期待よりも小さい場合）は，そのコネクションの K が上記の一定値未満となり，輻輳ウィンドウの回復にて他のコネクションより有利となり，結果，このコネクションの不利な状況は改善されると期待される．

5. RTT の 3 乗根に基づく手法の評価

本章にて，RTT の 3 乗根に基づく手法の性能評価を行い，その有効性について考察する．

5.1 性能評価

3 章の評価と同様に，図 4 の実験環境にて PC1-PC3 間と，PC2-PC3 間で netperf のコネクションを同時に確立し，通信

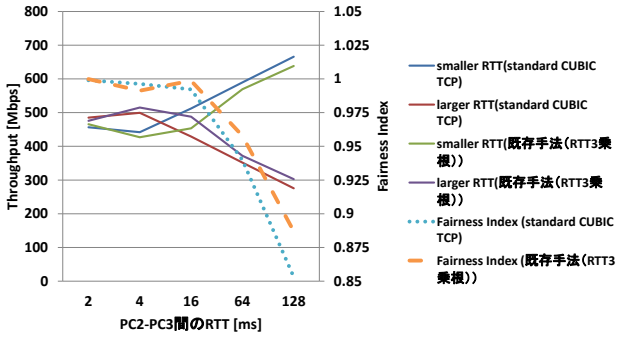

図 14　評価結果（PC1-PC3 間の RTT=2ms, 4vs4 コネクション）

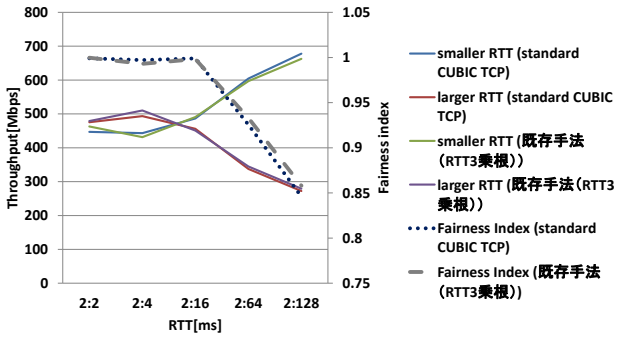

図 15　評価結果（PC1-PC3 間の RTT=2ms, 16vs16 コネクション）

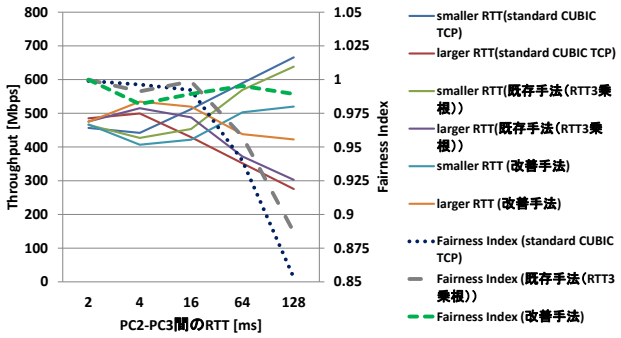

図 16　評価結果（PC1-PC3 間の RTT=2ms, 4vs4 コネクション，改善手法）

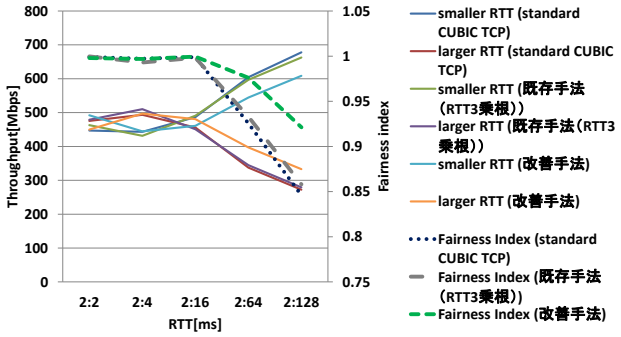

図 17　評価結果（PC1-PC3 間の RTT=2ms, 16vs16 コネクション，改善手法）

速度を測定した.

単一コネクション同士(1 コネクション vs 1 コネクション)の環境における評価結果を図 10 から図 13 に示す. 図より，本手法は CUBIC TCP の RTT 公平性を大きく改善していることが確認できる.

次に，複数コネクション同士(4 vs 4，16 vs 16)の環境における公平性の評価を図 14 と図 15 に示す．図より，RTT 差が小さい環境では複数コネクション同士の競合通信環境でも高い公平性を達成できていることが分かるが，コネクション数が多くかつ RTT 差が大きい環境においては本手法により得られる RTT 公平性は通常の CUBIC TCP で得られる公平性に近く，十分な公平度でないことが分かる．特にコネクション数が多い 16 vs 16 において大きな公平性の劣化を確認できる.

5.2 改善手法

前節の評価にて, 多コネクション環境にて RTT の 3 乗根に基づく手法では十分な公平性を実現できない例があることが確認された．図 14，15 より，RTT が大きなコネクションの性能が RTT が小さなコネクションの性能より低くなっていることが分かる．本手法は 3.2 節の考察に基づき，輻輳ウィンドウが十分でない大きな RTT のコネクションの K を小さくして，競合において優位な状況にし，結果としてそのコネクションの輻輳ウィンドウが増加すること期待している．しかし，前節の一部の環境(コネクション数 16 vs 16, RTT 2ms vs128ms など)では K の制御により与えた大きな RTT のコネクションへの優位性が十分ではなく, 大きな RTT のコネクションの性能が大きく下回る結果となっている.

そこで，RTT が大きな通信の優位性をさらに強めるために，K を以下の式により定める手法について考察する.

$$K = a \times \sqrt[3]{\frac{Wmax\,\beta}{c}} \times \frac{1}{\sqrt[3]{RTT}} \qquad (7)$$

この手法を実装し, 評価した結果を図？から図？に示す. 図より，RTT が大きなコネクションにさらに強い優位性を与えることにより，コネクション差が大きい環境における公平度を大幅に改善できることが分かる．同時に，RTT 差が大きくない環境では, 大きな RTT のコネクションに過度な優位性を与えており, 大きな RTT のコネクションの性能が小さな RTT の通信の性能を上回り, 公平性の劣化が起きていることも確認できる.

7. おわりに

本稿では Linux の標準 TCP 実装である CUBIC TCP の RTT 公平性に着目し，その評価と既存の RTT 公平性向上手法の紹介を行った．そして，既存手法の課題を示し，既存手法の改善方法について考察した.

今後は，他の TCP アルゴリズムとの TCP 公平性についての評価を行っていく予定である.

謝辞

本研究は JSPS 科研費 24300034，25280022，26730040 の助成を受けたものである.

文　献

[1] W. Richard Stevents, "TCSlow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," IETF RFC 2581, 1997.

[2] D.Katabi, M.Handley, and C.Rohrs, “Congestion control for high bandwidth-delay product networks,” in Proceedings of ACM SIG-COMM 2002,Aug.2002.

[3] 大浦亮, 山口実靖, "実機を用いた高速 TCP の公平性の評価", FIT2011 第 10 回情報科学技術フォーラム, RL-003, Sep. 2011

[4] Ryo Oura, Saneyasu Yamaguchi, “Fairness Comparisons Among Modern TCP Implementations,”The 6th International Workshopon Telecommunication Networking, Applications and Systems (TeNAS 2012), Mar. 2012.

[5] L. Xu, K. Harfoush and I. Rhee, “Binary Increase Congestion Control for Fast Long-Distance Networks,” Proc. IEEE Info COM 2004, March 2004

[6] Injong Rhee and LisongXu “CUBIC: A New TCP-Friendly High-Speed TCP Variant,” Proc. Workshop on Protocols for Fast Long Distance Networks, 2005.

[7] Kun Tan, Jingmin Song, Qian Zhang, and MurariSridharan,” A Compound TCP Approach for High-speed and Long Distance Networks” Proc.IEEE Info COM 2005,July 2005.

[8] 神津智樹，大浦亮，秋山友理愛，山口実靖，"CUBIC TCP の RTT 公平性の改善"，情報処理学会 第 75 回全国大会 3X-1 ，2013

[9] Tomoki Kozu, Yuria Akiyama and Saneyasu Yamaguchi, "Improving RTT Fairness on CUBIC TCP", The First International Symposium on Computing and Networking

[10] 逸身勇人，山本幹，“CUBIC と Compound TCP 間の公平性改善手法の提案，”電子情報通信学会信学技報 vol. 110, no. 372, NS2010-160，pp. 103-108

[11] 長谷川剛，板谷夏樹，村田正幸，“バックボーンルータにおける RED の動的閾値制御方式，”電子情報通信学会信学技報　NS2001-11

[12] 神津智樹，秋山友理愛，山口実靖，“輻輳ウィンドウ回復時間調整による CUBIC TCP の RTT 公平性の改善”，電子情報通信学会　進学情報，vol.113，no.472，NS2013-193，pp.97-102，2014.

[13] Jeonghoon Mo, Richard J. La, VenkatAnantharam, and Jean Walrand, “Analysis and comparison of TCP reno and vegas”, in Proceedings of IEEE INFOCOM’99, March 1999.

[14] L. S. Brakmo and L. L. Peterson, “TCP Vegas: End to End Congestion Avoidance on a Global Internet”, IEEE Journal on Selected Areas in Communication, Vol.13, No.8, pp.1465-1480, October 1995.

[15] Floyd, Sally, “Connections with multiple congested gateways in packet-switched networks part 1: one-way traffic,” SIGCOMM Comput. Commun. Rev., Vol. 21, No. 5, pp. 30-47, 1991

[16] S. Floyd and V. Jacobson, "On Traffic Phase Effects in Packet-Switched Gateways," Computer Communication Review (ACM), Vol. 21, No. 2, 1991.

[17] Henderson, T.R.; Sahouria, E.; McCanne, S.; Katz, R.H., "On improving the fairness of TCP congestion avoidance," Global Telecommunications Conference, 1998. GLOBECOM 1998. The Bridge to Global Integration. IEEE , vol.1, no., pp.539,544 vol.1, 1998

[18] G. Marfia, C. E. Palazzi, G. Pau, M. Gerla M. Y. Sanadidi, and M. Roccetti, "Balancing video on demand flows over links with heterogeneous delays," Proceedings of the 3rd international conference on Mobile multimedia communications, pp. 1-6, 2007

[19] Kazumine OGURA, Yohei NEMOTO, Zhou SU, and Jiro KATTO, “A New TCP Congestion Control Supporting RTT-Fairness,” IEICE TRANSACTIONS on Information and Systems Vol.E95-D No.2 pp.523-531, 2012

[20] 鈴川 竜司 , 安達 直世 “CUBIC-TCP におけるフロー間帯域の公平性に対する改善手法の提案” 電子情報通信学会 信学技報 NS2008-108

[21] 秋山友理愛，大浦亮，神津智樹，山口実靖，“実機と実 TCP 実装を用いた TCP 公平性の評価”電子情報通信学会　信学情報 NS2012-149，pp.49-54，2012

[22] netperf homepage, http://www.netperf.org/netperf/

[23] D.Chiu and R.Jain，“Analysis of the increase and decrease algorithms for congestion avoidance in computer networks，” Computer Networks and ISDN Systems, Volume 17, Issue 1, pp. 1-14, 1989.