L-033

CUBIC TCP の輻輳ウィンドウ回復時間の制御による TCP 公平性の改善
Improvement of TCP Fairness with Controlling Congestion Windows Size Recovery Time

神津 智樹† 山口 実靖‡
Tomoki Kozu† Saneyasu YAMAGUCHI‡

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．現在まで多くの OS に実装されてきた典型的な TCP Reno ではネットワーク帯域を使い切れないという問題が指摘されている[1]．そのため，TCP Reno に変わる多くの TCP が提案されてきた．しかし，これらの新しい TCP 間の公平性の問題が指摘されている[2]．特に RTT の差がある時に TCP 間の公平性の問題が大きい．

本稿では， Linux の標準 TCP 実装である CUBIC TCP の輻輳ウィンドウの回復までの時間に着目し，RTT が異なる環境における TCP 間の性能公平性について考察する．

2. 輻輳制御アルゴリズム

TCP はネットワークの輻輳状況に応じて輻輳ウィンドウサイズを自動的に増減させ，ネットワークへ送出するデータ量を調節することにより輻輳制御を行う．典型的な輻輳制御アルゴリズムとして TCP Reno があり，それを改良した新しいアルゴリズムとして CUBIC TCP などがある．

2.1 TCP Reno

TCP Reno の輻輳制御手法は，スロースタートフェイズおよび輻輳回避フェイズで構成され，それぞれにおいて輻輳ウィンドウサイズの増加速度が異なる．スロースタートフェイズにおいては，1 つの ACK パケットを受信するごとに輻輳ウィンドウサイズを 1 パケット増加させる．一方，輻輳回避フェイズにおいては，1 つの ACK パケットを受信するごとに輻輳ウィンドウサイズをその逆数分だけ増加させる．すなわち，TCP Reno の輻輳ウィンドウサイズを *cwnd* とすると，そのアルゴリズムは以下のとおりとなる．*ssthresh* は TCP Reno がスロースタートフェイズから輻輳回避フェイズに移行するときのしきい値である．パケットロスを検出すると輻輳ウィンドウサイズを 1 あるいは半分に下げる．

$$cwnd \leftarrow \begin{cases} cwnd + 1 & (cwnd < ssthresh) \\ cwnd + \frac{1}{cwnd} & (cwnd \geq ssthresh) \end{cases} \quad (1)$$

2.2 CUBIC TCP

CUBIC TCP は，BIC TCP のスケーラビリティを維持しながら，TCP Fairness と RTT Fairness，制御手法の複雑さを改善した高速 TCP である[3].

† 工学院大学大学工学研究科院電気・電子工学専攻
Electrical Engineering and Electronics, Kogakuin University Graduate School
‡ 工学院大学工学部情報通信工学科 Department of information and Communication Engineering, Kogakuin University

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次の式のような 3 次関数を用いた制御によって実現している．

$$cwnd = c(t - K)^3 + W_{max} \quad (2)$$

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

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．一般に *C* は 0.4，βには 0.2 が用いられる．*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．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，ウィンドウサイズの増加から RTT の影響を排することで RTT Fairness の向上を目指しており，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を解決している．

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，輻輳ウィンドウサイズの値が性能に影響を与え，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．

3. 提案手法

本章で RTT を考慮して *K*（輻輳ウィンドウの回復時間）を調整し，公平性改善手法を提案する．CUBIC TCP は 2 節で述べたとおりパケットロスから輻輳ウィンドウが回復するまでの式(3)の *K* に依存している．*t*=*K* となると輻輳ウィンドウが回復し，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．一方，TCP Reno では一次関数で輻輳ウィンドウサイズを回復させるため，回復すべき必要な輻輳ウィンドウサイズが大きいほど回復に長い時間を要してしまう．よって，CUBIC TCP と TCP Reno の間に性能の不公平が生じると予想される．

そこで本稿では *K* の決定式を式(4)の様に修正し，公平性の向上を目指す手法を提案する．*x(RTT)*は RTT によって変化する値であり，RTT が小さいほど大きな値となる．

$$K = \sqrt[3]{\frac{W_{max}\beta}{C}} \times \mathrm{x(RTT)} \quad (4)$$

4. 性能評価

4.1 測定環境

図 1 のネットワークを構築し，TCP Reno と CUBIC TCP で同時に通信を行い，TCP Reno と CUBIC TCP の性能の公平性の評価を行った．全ての PC にて Linux が動作しており，PC1 では TCP Reno が，PC2 では CUBIC TCP が選択されている．通信は netperf を用いて行われ，PC1 から PC3 への送信(TCP Reno)と，PC2 から PC3 への送信（CUBIC TCP）を同時に行った．Network Emulator は，人工的にネットワーク遅延を発生させる装置である．Network Emulator にて TCP Reno コネクションの RTT を 1ms, 4ms, 16ms，CUBIC TCP コネクションの RTT を 1ms, 4ms, 16ms, 64ms に設定して評価を行った．全ての実験で受信ウィンドウサイズは 10MB とし，提案手法における *x(RTT)*は図 2 の通りとした．

測定結果を図 3, 4, 5 に示す．図 3 は Reno 側の RTT を 1ms，図 4 は 4ms，図 5 は 16ms に設定している．図 3 においては，提案手法の適用により RTT 公平性が大きく改善できていることがわかる．図 4 においても 4ms 対 64ms の場合を除き RTT 公平性が改善していることを確認でき，提案手法の有効性を確認することができ，これらの状況において提案手法は有効であると言えるが，図 4 の 4ms 対 64ms や図 5 において公平性の悪化も見られ，全ての状況に対応するには更なる改善が必要であると言える．

5. おわりに

本研究では，CUBIC TCP の輻輳ウィンドウ回復時間の調整を行い，TCP Reno との RTT 公平性向上手法を提案し，性能測定を行った．TCP Reno の RTT が低い時は，公平性の向上が確認できたが，RTT が高いときは公平性が下がる結果となった．

今後は，関数(x)の決定式の改善を行い，CUBIC TCP の輻輳制御手法の改善を行なっていく予定である．

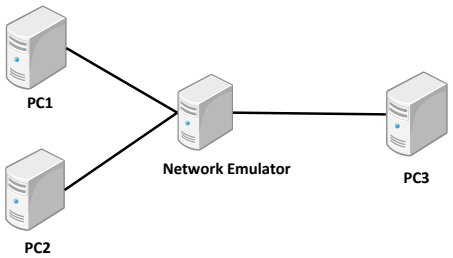

図 1　ネットワーク図

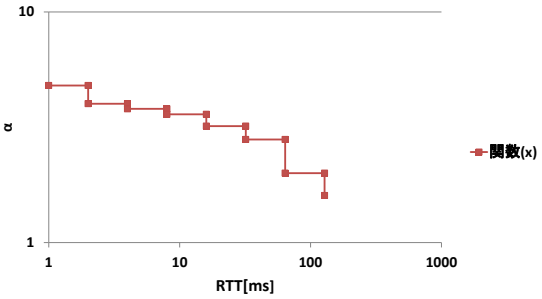

図 2　関数 *x(RTT)*

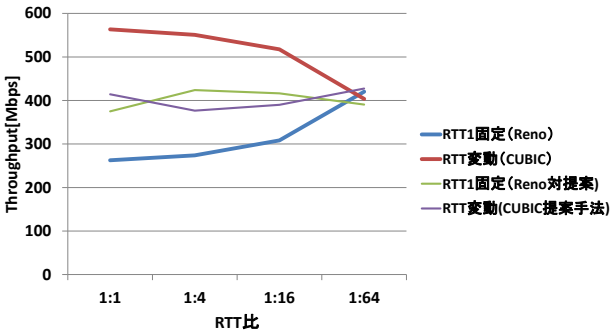

図 3 RTT と Throughput（RenoRTT1ms 固定）

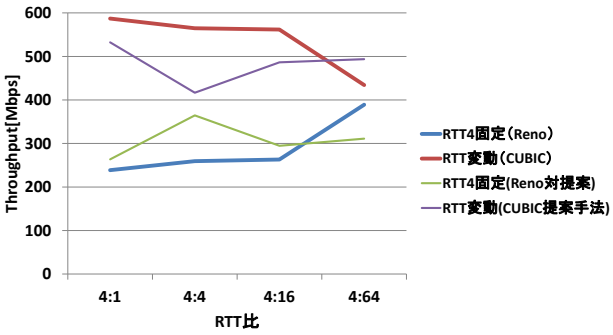

図 4 RTT と Throughput（Reno4ms 固定）

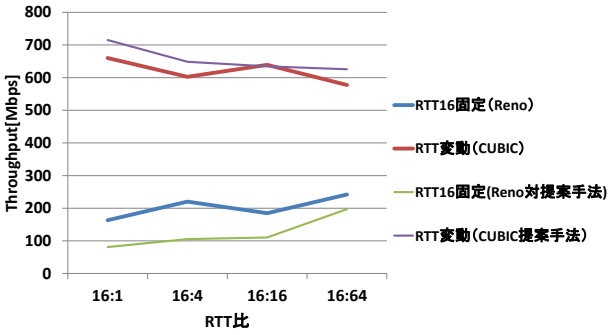

図 5 RTT と Throughput（RenoRTT16ms 固定）

謝辞

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

参考文献

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

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

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

[4] 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

[5] 長谷川剛，村田正幸，“TCP の輻輳制御機構に関する研究動向”. 電子情報通信学会論文誌　B Vol. J94-B No.5 pp.663-672