遅延時間制御の優先パケット破棄の実ネットワークへの適用

花井雅人[1]　山口 実靖[1]　小林 亜樹[1]

概要： この近年のネットワークの大容量化に伴い，太く長いネットワーク(ロングファットネットワーク)の帯域を古典的な TCP 輻輳制御アルゴリズムである TCP Reno では有効に活用することが困難となり，CUBIC TCP や Compound TCP などの複数の高速 TCP アルゴリズムが提案された．これら複数の TCP の提案により，TCP 間公平性の課題が生じた. 我々はこれらのアルゴリズム間での性能公平性(TCP 公平性)に着目し, 遅延時間制御を改変することにより TCP 公平性を向上させる手法を提案した．本手法は通信機器にて自身を通過するパケットを監視し，ネットワーク帯域の消費が大きい通信の推定を行う．そして，ネットワーク帯域の消費が大きい通信のパケットを優先的に破棄することにより TCP 公平性を実現する．また，実験室ネットワークを用いて上記手法の詳細な性能評価を行い，その有効性を示した．具体的には，各種ネットワーク遅延時間の環境におけるスループットと公平性，各種パケット破棄率におけるスループットと公平性，各種通信機器台数や各種コネクション数におけるスループットと公平性を示した．本稿では上記手法を実ネットワークや無線ネットワークに適用し，評価と考察を行う．具体的には，本手法を学内ネットワークに適用しその公平性を評価した結果を示し，本手法が実ネットワークにおいても効果的であったことを示す．また，本手法を IEE802.11ac 無線 LAN 環境に適用し公平性を評価した結果を示し，無線 LAN 環境においても有効であることを示す.

A Study on Applying Modified Controlled Delay for Practical Networks

Masato HANAI[1]　Saneyasu YAMAGUCHI[1]　Aki KOBAYASHI[1]

1. はじめに

TCP における古典的な輻輳制御アルゴリズムである TCP Reno は近年の高速なネットワークの帯域を使い切ることができないと言われ，BIC TCP[1]，CUBIC TCP[2], Compound TCP[3]などの次の世代の高速 TCP が複数提案されている．BIC TCP はバイナリーサーチにより最適な輻輳ウィンドウの大きさを検索するアルゴリズムであり，非常にアグレッシブに輻輳ウィンドウを増加させ高速な通信が可能である．ただし，輻輳ウィンドウの増加が積極的すぎるとも考えられている. CUBIC TCP は 3 次関数を用いて輻輳ウィンドウを増加させる手法であり，高速な通信が可能であると同時に BIC TCP よりも積極性を抑えてある．これは, Linux OS にて初期設定で選択されている TCP 輻輳ウィンドウ制御手法となっている．Compound TCP はハイブリッド型と言われており，TCP Vegas[4]の遅延ベースの考え方と TCP Reno などのロスベースの考え方の両方が組み合わせて使用されている．これは，Windows OS に搭載されている手法である．これらの新しい TCP アルゴリズムの提案により，通信性能の改善が達成されるとともに，TCP アルゴリズム間の性能公平性という問題が生じた．例えば，CUBIC TCP を使用している Linux OS と Compound TCP を使用している Windows OS がネットワークを共有している状況において同時に通信を行った場合，通信性能に著しく不公平となる問題があることがシミュレーションによる検証[5][6]や，実機を用いた検証[7][8]にて確認されている．

これらの不公平性の問題に対して, 我々は RED[9]を改良した Active Packet Dropping[10]という手法を提案した．この手法は，RED を適用している通信機器において，通信帯域の消費の大きなフローのパケットを優先的に破棄する手法であり, 性能評価において TCP 公平性の改善を実現できることが確認されている．しかし，RED はパラメータ設定が容易でなく普及も必ずしも進んでおらず，より普及が期待される手法に基づく方法について考察が重要となっていた.

より容易な設定で使用できる待ち行列管理手法として遅延時間制御[11]がある. CoDel は待ち行列における待ち時

1 工学院大学 大学院 工学研究科 電気・電子工学専攻
Electrical Engineering and Electronics, Kogakuin University Graduate School

間のみでパケット破棄の判断を行うより単純な手法であり，今後の普及が期待されている．我々は文献[12]にて，今後の普及が期待される CoDel に基づく TCP 公平性向上手法を提案した．本手法は，通信機器にてパケットを観察し，ネットワーク帯域の消費が大きいフロートとそれ以外のフローでパケット破棄の確率を変えるものであり，有線接続の実験用ネットワークにてその有効性が確認されている．ただし，本手法が実ネットワークにおいても有効であるかや，無線ネットワークにおいても有効であるかは検証がされていない．

本稿では，CoDel に基づく TCP 公平性改善手法[11]の実ネットワークにおける評価と，無線ネットワークにおける評価を行い，本手法がこれの環境においても有効であることを示す．具体的には，大学の学内ネットワークへ適用しての性能評価結果と，IEEE 802.11ac 無線 LAN 環境への適用結果を示し，TCP 公平性の改善が見られることを示す．

本稿の構成は以下の通りである．2 章にて CoDel に基づく TCP 公平性改善手法などの既存研究を紹介する．3 章にて，本手法の実ネットワークと無線 LAN への適用結果を示し，本手法がこれらの環境においても有効であることを示す．4 章にて，本稿をまとめる．

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 実装には輻輳制御機能が搭載されている．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 を用いる通信の性能はこのアルゴリズムに大きな影響を受ける．OS により様々な輻輳制御アルゴリズムが実装されおり，それぞれの OS で輻輳制御の仕方が異なる．

TCP 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の 3 つに分類できる．

ロスベース手法はパケットが破棄されたのを検出し，これに基づいて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，パケット破棄が検出された時には輻輳ウィンドウを減少させる．従来の TCP である TCP Tahoe，TCP Reno や，近年の Linux OS で使用されている BIC TCP[1]，CUBIC TCP[2]がロスベース手法である．

遅延ベース手法は，RTT の増減に基づいて輻輳ウィンドウを制御する手法である．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を期待することができる．欠点としてロスベース手法と同一ネットワークにおいて同時に通信を行っ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[4][5]. 代表的な遅延ベース手法に TCP Vegas[4]がある．

ハイブリッド型手法はロスベースと遅延ベースを組み合わせた手法である．代表的なハイブリッド型手法の TCP に Windows 系 OS に搭載されている Compound TCP[6]がある．本研究では，Linux OS の標準 TCP である CUBIC TCP と Windows 系 OS に搭載されている Compound TCP に焦点を当て考察を行う．

2.2. Bufferbloat

近年のルータなどの通信機器には大容量のバッファが搭載されている．現状のインターネットのルータのバッファは常にパケットで満たされており，各パケットは長い待ち行列を待つ必要があり，結果として現状のインターネット通信の通信遅延は常に大きい状態にあるとの指摘[13][14]があり，この問題は Bufferbloat と呼ばれている．Bufferbloat の解決策の一つとして，次節で述べる遅延時間制御手法(CoDel)[11]がある．

2.3. CoDel (Controlling queue Delay)

CoDel[11]は Queue のスケジューリングアルゴリズムである．CoDel は，あるパケットが Queue に入ってから Queue を出るまでの時間が *target* 時間以上になると，パケットを破棄する．*target* はチューニングパラメータであり，初期値は 5ms である．

RED と異なり CoDel は設定(チューニングパラメータ)が少なく，パケット破棄するか否かは待ち行列の滞在時間のみで決定される．RED の普及がかならずしも進まない原因の一つとしてパラメータの多さと，それらの設定の難しさがあげられ，CoDel ではより簡単な設定で高い性能を実現でき，今後は普及が進んでいくと期待することができる．

RED と同様に，CoDel にてルータ上のパケット待ち行列を制御することにより TCP 公平性の改善が実現できると考えられ，公平性の改善など関しては，今後は本手法を基にした考察を行うことが重要になっていくと考えられる．

2.4. 静的優先制御公平性改善手法

前節で述べたように，CoDel の適用により TCP 公平性の改善ができると期待できる．我々は文献[12]において CoDel 適用環境と非適用環境における TCP 公平性の評価を行い，一部の環境にて CoDel の適用により TCP 公平性の改善が可能であること，一部の環境では CoDel を適用しても公平性が低いことを示した．そしてネットワーク帯域を大きく消費する接続が通信機器にとって既知であることを前提とし，この前提の元で TCP アルゴリズム間性能公平性を向上する手法を提案し，性能評価により公平性の改善を示した．

しかし，一般にネットワーク帯域を大きく消費する接続は通信機器にとって既知ではなく，この前提なく適用可能な手法の考察が重要な課題となっている．

2.5. 動的優先制御公平性改善手法

動的優先制御改善手法[12][16]は，前節の静的優先制御公平性改善手法に基づいた手法であり，帯域消費の大きいフローのパケットを優先的に破棄する.ただし，ネットワーク帯域を大きく消費する接続が通信機器にとって既知であることを前提としていない．この手法は通信機器に最もパケットを送信した通信を最も帯域消費が大きいコネクションと定義する．通信機器は *rec_int* パケット受信するごとにコネクション情報を記録する．通信機器はパケット情報を *hist_len* 個保持する．そして，*stat_int* パケット受信する毎に履歴において最も高い頻度で登場するフローが調査される．こうして最も帯域を消費しているコネクションを推定する．本手法は有線接続の実験用ネットワークのみで評価されており，実ネットワークや無線ネットワークでの評価はされていない．

2.6. 動的優先制御公平性改善手法の性能

動的優先制御改善手法[12]では実験用ネットワーク上でTCP 公平性についての評価が行われている．本稿における評価結果と比較するために，本節にてその評価結果を紹介する．

図 1 のネットワーク[12]を構築し，CUBIC TCP コネクションと Compound TCP コネクションが混在する環境における通信速度を iperf [17]を用いて測定している．

測定結果[12]を図 2 に示す．横軸はネットワーク内における送信者-受信者間の往復遅延時間である．縦軸は各通信が得た通信速度で 10 コネクションの平均値であり.図より，TailDrop や CoDel 使用時は CUBIC TCP の通信速度と Compound TCP の通信速度には大きな差があり，TCP 公平性が非常に低いことがわかる．特に付加遅延時間=64ms において公平性が低くなっている．また，本手法を適用することにより付加遅延時間=64ms においても高い公平度が実現できていることがわかる．この結果より，本手法の有効性が有線接続の実験用ネットワークで確認されていることが分かる．

3. 実ネットワーク性能評価

CoDel に基づく動的優先制御改善手法[12][16]の実ネットワークや無線ネットワーク環境における有効性を検証するために，本手法を工学院大学学内 LAN および IEEE802.11ac 無線 LAN 環境に適用し，TCP 公平性の評価を行った．

3.1. 実ネットワークにおける評価

図 3 の測定環境にて Linux1-Linux2 間と，Windows-Linux2 間で iperf のコネクションを同時に確立し，通信速度を測定した．Linux1 は学内の別サブネットワークに設置してあるサーバであり，学内のネットワークを介して Linux2 とのコネクションを確立している．

Linux 1，Linux 2，Windows の 3 台の計算機の仕様は表 1

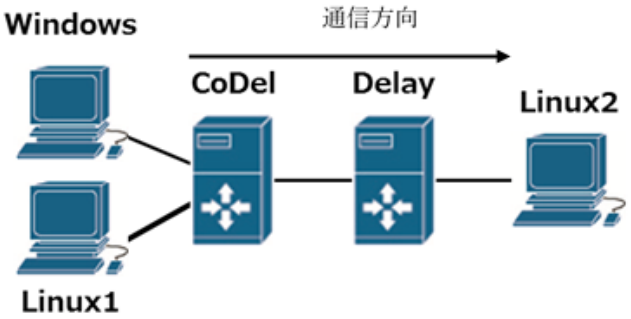

表 1　計算機仕様

CPU	AMD Turion, 2.20[GHz]
メモリ	4[GB]
OS	(Linux1) Linux 4.2.3 (Linux2) Linux 3.3.4 (Windows) Windows7 Enterprise

表 2　計算機仕様

CPU	Intel CelronG540, 2.4[GHz]
メモリ	2[GB]
OS	(Codel, Delay) Linux 3.17.4

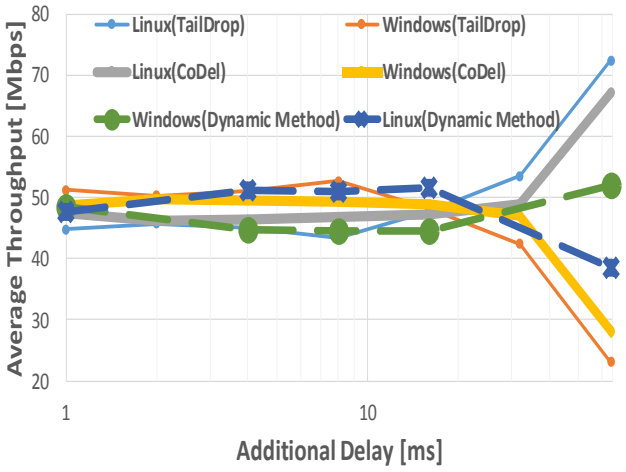

図 2　評価結果[12]

の通り，CoDel 及び本手法を搭載した CoDel 機およびネットワークの遅延をエミュレートする Delay 機の仕様は表 2 の通りである．Delay 機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，Linux Netem を用いて構築されている．ネットワーク機器は全て 1Gigabit Ethernet に対応している．

通信は，Linux1-Linux2 間と，Windows-Linux2 間で同時に行い，Linux1 と Windows を送信端末，Linux2 を受信端末としている．Linux1-Linux2 間の通信と，Windows-Linux2 間の通信は CoDel 機から Delay 機，Linux2 までのネットワークを共有している．Delay 機における人工的な遅延時間

は 1ms から 64ms 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 32MB としている．

評価結果を図 4，図 5 に示す．図 4 の横軸は Delay 機において人工的に付加した遅延時間，縦軸は 10 コネクションの平均スループットである．図 4 より，実ネットワークにおいても本手法を用いることで TCP 公平性を改善できることが確認できる．特に，TailDrop や CoDel にて公平性が低くなった遅延時間が 64ms のときに大きく公平性を改善できていることがわかる．

図 5 は Fairness Index を示している．Fairness Index は公平度を表す指標であり，0 から 1 の値をとり，1 に近いほど高い公平度を表している．図より，本手法を用いることによりいずれの遅延時間においても高い公平度を実現することができ，特に遅延時間が 64ms のときに既存の手法(TailDrop, CoDel)よりも高い公平度を実現できていることが分かる．また，図 2 と比較しても実験用ネットワークと近い公平性が実現されていることが分かる。これらより，有線接続の実ネットワークにおいても本手法が有効性であることが確認できる．

3.2. IEEE802.11ac 無線 LAN における評価

本節にて，無線 LAN 環境における本手法の評価を行う．図 6 の実験環境にて Android-Linux2 間と，Windows-Linux2 間で iperf のコネクションを同時に確立し，通信速度を測定した．図内の CoDel 機と Delay 機の間は 100Mbps Ethernet で接続されており，Android 機と Wireless router 間は IEEE802.11ac 2x2 にて接続されている．それ以外の接続は 1Gigabit Ethernet で接続されている．

本評価では，前節の Linux1 の代わりに Android を搭載したスマートフォン端末を用いており，無線 LAN アクセスポイントを経由して Android-Linux2 間で iperf のコネクションを確立している．Android 端末には Galaxy s7 edge を使用した．表 3 に Android 端末の仕様を示す．Android OS は Linux カーネルを搭載しており，TCP アルゴリズムとしては CUBIC TCP を使用している．これら以外は，前節の測定と同じ測定環境となっている．

測定結果を図 7, 8 に示す．横軸の値は Delay 機により付加したネットワーク内における送信者-受信者間の往復遅延時間である．図 7 の縦軸は各通信が得た通信速度である．図 8 の縦軸は Fairness Index である．

図 7 より，TailDrop を用いた場合は Android 搭載のスマートフォンと，Windows 機における性能公平性が極めて低く，Linux カーネルを搭載しており CUBIC TCP が動作する Android スマートフォンの性能が大幅に高くなることが確認できる．また，CoDel を用いることにより公平性の大幅な改善が達成されるが，遅延時間が大きな環境においては公平性が低いままであることが分かる．これに対して本手法を用いた場合はすべての遅延時間において高い公平度を実現できていることが分かる．特に，遅延時間が大きな状

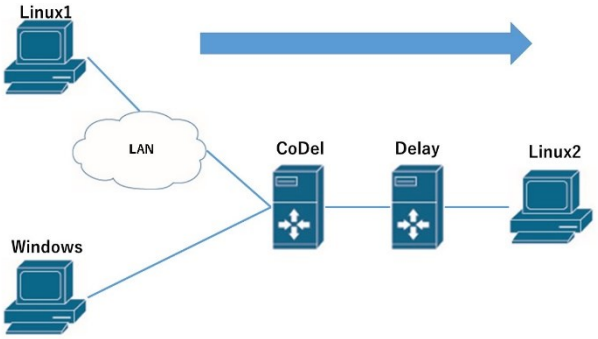

図 3　測定環境

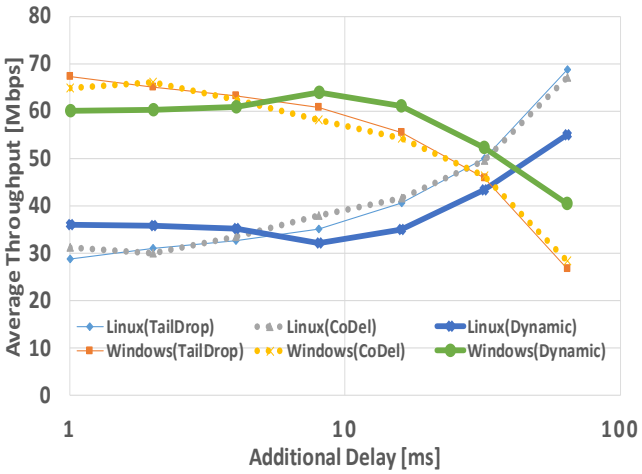

図 4　評価結果

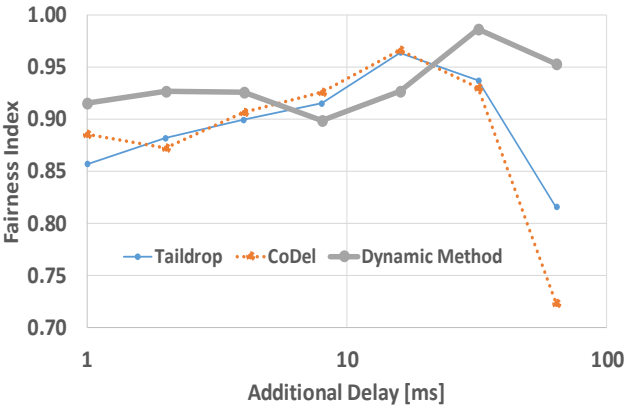

図 5　Fairness Index

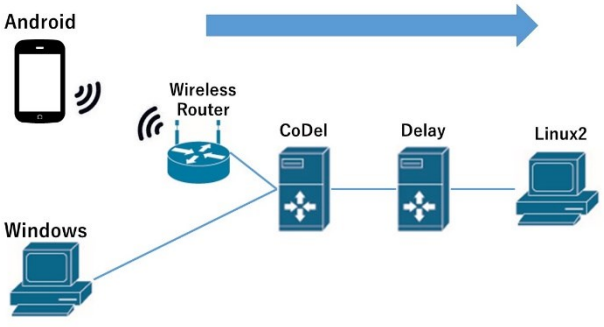

図 6　測定環境

況において CoDel よりも高い公平性を実現していることが分かる．また，図 2 と同程度の公平性が達成されていることが分かり，無線 LAN 環境においても優先ネットワーク環境と同程度の公平性が実現できることが分かる．これらのことより，提案手法が無線 LAN 環境においても有効であることが確認されたと言える．ただし，遅延時間が少ない状況において本手法により得られた公平度が CoDel により得られる公平度よりも低いことがあることが確認され，本手法のさらなる改善と原因の考察が必要であることも確認された．

4. おわりに

本稿では，CoDel を改変し TCP 公平性を改善する手法を紹介し，同手法が有線接続の実験用ネットワークにて高い公平性を実現できることが確認されていることを述べた．そして，同手法の有線接続の実ネットワーク環境における評価結果と，IEEE 802.11ac 無線 LAN 環境における評価結果を示し，同手法がこれらの環境においても高い公平性を実現できることを示した．また，一部の環境においては本手法により得られる公平度が CoDel により得られる公平度よりも低くなることがあることも示した．

今後はチューニングパラメータの決定に関する考察，セルラー回線や長距離公衆実ネットワークでの評価などをしていく予定である．

謝辞

本研究は，JST，CREST JPMJCR1503 の支援を受けたものである．

本研究は JSPS 科研費 25280022，26730040，15H02696 の助成を受けたものである．

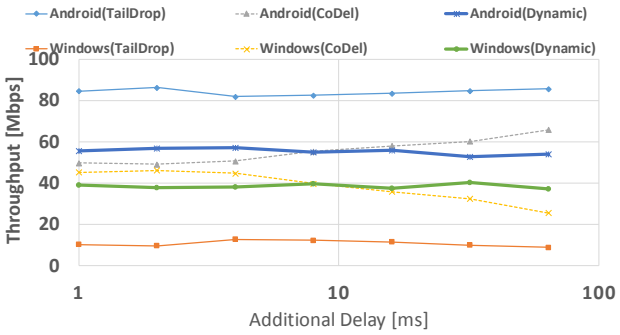

図 7　測定結果

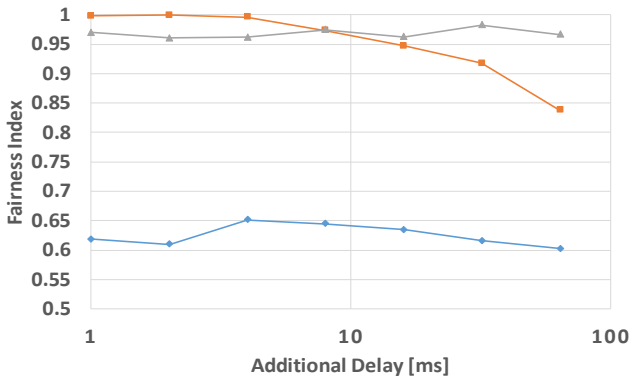

図 8　測定結果

表 3 計算機仕様

CPU	Snapdragon 820 2.15GHz
メモリ	4GB
OS	Android 6.0.1 (Linux 3.18.10)

参考文献

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

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

[3] 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.

[4] 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.

[5] 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

[6] 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

[7] Ryo Oura, Saneyasu Yamaguchi, “Fairness Analysis among Modern TCP Congestion Avoidance Algorithms Using Actual TCP Implementation and Actual Network Equipments,” ICNC 2011: 297-299

[8] Ryo Oura, Saneyasu Yamaguchi, “Fairness Comparisons among Modern TCP Implementations,” AINA Workshops 2012: 909-914

[9] S. Floyd and V. Jacobson, “Random early detection gateways for congestion avoidance,” IEEE/ACM Transactions on Networking, vol. 1, pp. 397.413, Aug. 1993.

[10] Akiyama, Y.; Kozu, T.; Yamaguchi, S., “Active packet dropping for improving performance fairness among modern TCPs,” in Network Operations and Management Symposium (APNOMS), 2014 16th Asia-Pacific , vol., no., pp.1-4, 17-19 Sept. 2014

[11] Kathleen Nichols and Van Jacobson. 2012. Controlling queue delay.Commun. ACM 55, 7 (July 2012), 42-50.

[12] 花井雅人，山口実靖，小林 亜樹 "遅延時間制御手法の動的調整による TCP 公平性の向上"，マルチメディア，分散，協調とモバイル(DICOMO 2016),,June. 2016

[13] 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

[14] Jim Gettys and Kathleen Nichols. 2011. Bufferbloat ACM (November 2011),

[15] 花井雅人，山口実靖，"実機実装を用いた遅延時間

制御手法の改良による TCP 公平性の向上", ネットワークシステム研究会(NS),(42)NS, March. 2016

[16] Masato Hanai, Saneyasu Yamaguchi, Aki Kobayashi, “Modified Controlling Queue Delay for TCP Fairness Improvement”, The 18th Asia-Pacific Network Operations and Management Symposium (APNOMS 2016), October 2016, Kanazawa, Japan.

[17] iperf homepage, https://iperf.fr/