無線 LAN 環境におけるハンドオーバを伴う移動端末のノード間競合に関する一検討

森内　彩加† 　安藤　玲未† 　村瀬　勉†† 　小口　正人†

† お茶の水女子大学　〒 112–8610 東京都文京区大塚 2–1–1
†† NEC　〒 211–8666 神奈川県川崎市中原区下沼部 1753
E-mail: †{ayaka-m,remi}@ogl.is.ocha.ac.jp, ††t-murase@ap.jp.nec.com, †††oguchi@computer.org

あらまし　近年，無線 LAN の普及，動画ストリームや音声などのマルチメディア通信の需要の増加によって，無線 LAN 環境における QoS(Quality of Service) 制御が大変重要となっている．QoS 制御として，各プロトコル層では様々な手法が既に提案されているが，実現が難しいものが多い．そこで，本研究では，QoS 制御の中でも，ネットワークの両端における制御が可能であることから，比較的実現が容易であると考えられるトランスポート層での制御を検討する．トランスポート層での制御の一つとして，通常使用する TCP をもとに，より積極的に帯域確保を目指すよう TCP の輻輳制御を変更した，QoS-TCP が提案されている．また，スマートフォンやタブレット端末などのモバイル端末の急速な普及により，モバイル端末を使用しながらアクセスポイント (AP) 間を自由に動き回る環境，つまり，ハンドオーバ (HO) を伴う移動の無線 LAN 環境においても，QoS 制御が重要である．

そこで，本研究では，HO を伴う移動通信環境において，QoS-TCP を利用することを検討する．既に，QoS-TCP を利用した評価として，ノート PC を利用した，HO を伴う移動通信環境での評価は行われているため，本稿では，Android 端末を利用した環境下での評価を行う．通常，移動端末は接続先において既に通信を行っている端末に比べて不利であるため，既存通信に割り込むことは難しい．しかし，ノート PC を利用した環境では，通常の TCP では既存通信に割り込むことができない場合でも QoS-TCP の場合は，帯域確保可能であった．一方，Android 端末を利用したモバイル環境においては，移動端末と AP 間の距離や AP のバッファサイズについて，ある条件を満たしたときのみ既存端末と同等に帯域確保できるということが明らかになった．

キーワード　TCP，無線 LAN，不公平，QoS，ハンドオーバ

A study on QoS characteristics for mobile station in wireless LANs

Ayaka MORIUCHI†, Remi ANDO†, Tutomu MURASE††, and Masato OGUCHI†

† Ochanomizu University　Otsuka 2–1–1,Bunkyo-Ku, Tokyo 112–8610 Japan
†† NEC Corporation
1753 Shimonumabe, Nakahara-ku, Kawasaki, Kanagawa, 211–8666, Japan
E-mail: †{ayaka-m,remi}@ogl.is.ocha.ac.jp, ††t-murase@ap.jp.nec.com, †††oguchi@computer.org

1. はじめに

近年，無線 LAN の普及，動画ストリームや音声などのマルチメディア通信の需要の増加により，無線 LAN 環境での QoS(Quality of Service) 制御が重要となっている．更に，スマートフォンやタブレット端末などのモバイル端末の急速な普及により，データの送受信をしながら動き回る機会が増加している．この時，ユーザは，モバイル端末を利用しながらアクセスポイント (AP) 間を自由に動き回っており，ハンドオーバ (HO) を繰り返している．そこで,「ハンドオーバを伴う移動通信」環境 (以下，移動環境) においても QoS 制御を検討する必要がある．本研究における QoS 制御とは，移動端末も既存通信と同様に帯域確保できることを目的とする．移動環境における QoS 制御については，各プロトコル層において様々な手法が検討されているが，我々はトランスポート層での制御である QoS-TCP を用いる手法を検討する．また，ライフログ動画を記録したり，動画を投稿して皆で共有したりする機会が増加することにより，uplink 通信 (端末から AP の方向) が増加する

ことが予想される．そこで本研究では，uplink のストリーミングをターゲットとする．

既存研究では，デスクトップ PC やノート PC を利用した移動環境 (Ubuntu10.04：Linux 2.6.32 使用． 以下，ノート PC 環境) において，QoS-TCP を利用した評価が行われてきた [1]. [1] では，移動端末が TCP である場合は，HO 直後，既に接続先で通信を行っている端末の中に割り込むことができないが，QoS-TCP を用いれば，既存通信に割り込むことができ，移動端末が TCP である場合よりも約 20 倍有利に帯域確保できることが既に明らかにされている．しかし，Android 端末を用いた環境では，HO 時の移動を伴う QoS-TCP の評価は行われていない．また，有線通信と AC 電源を前提としてデザインされた TCP を用いるデスクトップ PC やノート PC と，無線通信とバッテリのみが想定されている Android 携帯端末とでは，TCP の振舞などが異なり，既存研究と同様の環境で実験を行ったとしても，結果が異なることが予想される．

そこで本稿では，ノート PC 環境にて HO 後に帯域確保可能であった QoS-TCP を，Android 端末を用いた移動環境 (以下，Android 環境) に利用し，QoS-TCP の特性評価を行い，QoS-TCP が割り込める条件を調査する．以下，2 章でノート PC を利用した環境での HO を伴う移動端末についての既存研究について述べ，3 章では，Android 端末を使用した環境での不公平について説明する．次に，4 章では Android 環境下で HO 時に，TCP-AV が割り込める条件を示す．本稿における「割り込める」とは，移動端末も既存通信と同等以上のスループットが得られることを指す．更に，5 章では，ノート PC の TCP と Android 端末に搭載されている TCP-CUBIC の輻輳の違いについて述べ，最後にまとめを述べる．

2. 関 連 研 究

無線 LAN 環境における QoS 制御については，各プロトコル層において各種制御が提案されている．しかし実インターネット環境においては実現が難しいものが多い．例えば，VoIP などアプリケーションレベルに特化した QoS 制御では，多くの，あるいは新規のアプリケーションに対応できない．IP 層の制御では，ネットワーク全体に変更が必要になり，MAC 層の制御では，AP などの無線 LAN 機器自身の変更が必要になってしまうなどの課題がある．これに対し，トランスポート層での制御は，ネットワークの両端における制御が可能であるため，比較的実現が容易である．そこで，我々はトランスポート層における制御を検討する．また，UDP ではファイアウォールを通過できない可能性もあることから，信頼性の高い，TCP での制御を検討する．このトランスポート層における制御の一つに，通常使用する TCP に，QoS 制御を加えた QoS-TCP がある．

2.1 QoS-TCP

QoS-TCP について述べる．TCP の輻輳制御のみを利用して帯域保証を行う QoS-TCP として提案されている TCP-AV [2] について説明する．TCP-AV は，アプリケーションが要求する帯域の確保を目指す TCP である．TCP-AV は，TCP-Reno を拡張したものであり，最小 RTT を用いてスロースタート閾値を設定し，送信帯域が，アプリケーションから要求された帯域である目標帯域周辺で安定するように，輻輳ウィンドウを制御する．また TCP-AV は Linux をベースとしたプロキシ等の形として実装されているため，QoS を制御するその他のトランスポートプロトコルに比べて，送受信端末の変更をすることなく実現可能という特徴がある．また，パケットロス検出時にも，輻輳ウィンドウをできるだけ高く保つことで帯域確保を目指している．

ただし，競合する TCP の本数が多くなると fair-share(通信可能帯域を送信端末台数で割った値) も併せて低くなるため，帯域確保が困難になる．競合が増えると，次節で述べる不公平が起こりやすくなるが，そのような環境においても，TCP-AV は，fair-share 程度は確実に帯域を確保することが可能である．

2.2 無線 LAN 環境における不公平

QoS-TCP の帯域確保具合に大きな影響を与える不公平について述べる．不公平であるとは，同じ環境で通信しているにも関わらず，端末間でスループットが極端に異なる図 1 のような状態をさす．この不公平は，MAC 層の送信権制御，トランスポート層における輻輳ウィンドウ制御などが組み合わされ，AP における TCP-ACK あふれが原因で起こる [3]．そのため，不公平の問題は，通信する端末数が多くなると特に顕著に現れ，不公平となる端末数は，AP のバッファサイズに依存する．また，HO フローのような後発のフローほど，スループットを得られ難くなる．更に [4] では，一度不公平になってしまった端末は，その後スループットを上昇させられないということが示されている．

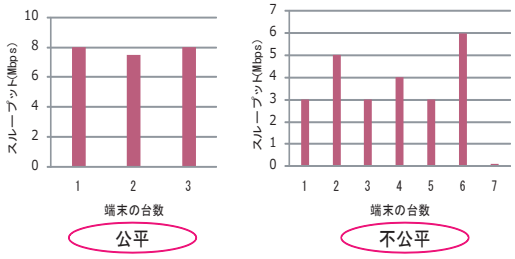

図 1 公平および不公平な状態

2.3 AP のバッファサイズ

不公平に関わる AP のバッファサイズは，ベンダによる実装次第であるため，一般には明らかになっていない．そのため我々は，既存研究，及び本稿で用いる AP のバッファサイズを独自に測定した．バッファサイズを直接計測することは不可能であるため，バッファが溢れるまで過大なトラヒックを与え，バッファが溢れたところがバッファサイズであると推定した．具体的な測定方法は，図 2 の実験環境のように 1 台の有線端末からシーケンス番号を付加した UDP パケット 1000 個を AP 経由で無線端末に送り出し，これを受信した無線端末で，受信回数と，送信端末で付加したシーケンス番号を比較する．

受信回数と送信端末で付加されたシーケンス番号が一致していれば，パケットがロスすることなく通信が行われているが，

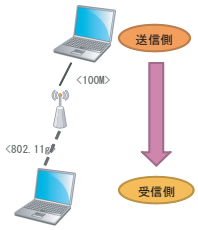

図 2　実験環境 1

番号が飛んでいる場合，その間のパケットはロスしているといえる．この原因が AP のバッファあふれであると考えられるため，番号の飛ぶ直前の数を AP のバッファサイズとする [4]．計測は 10 回行い，その平均値をとった．このような手法で，本稿で使用する AP のバッファサイズを測定した．結果は表 1 に示す通りである [5] [6] [7] [8].

表 1　AP のバッファサイズ測定結果

製品名	バッファサイズ (パケット)
BUFFALO WHR-HP-AMPG	135.1
Planex CQW-MR500	265.5
Planex GW-AP54SAG	284.3
Planex MZK-MF300N	566.9

2.4　既存研究

2.1 節で述べた TCP-AV を用いて, [1] では，図 3 のような実験環境において，検証を行っている．図 3 において，”送”は，各 AP にて通信を行っている送信端末 (背景端末) で，AP1 側，AP2 側に固定的に接続している送信端末数をそれぞれ N1(台)，N2(台) とする．”受”は受信端末，”移動”は移動端末を表わしている．移動端末は，全長 20m の所を，AP 間を結ぶ直線に対して平行に AP1 から AP2 に向けて移動し，中間地点で HO を行う．

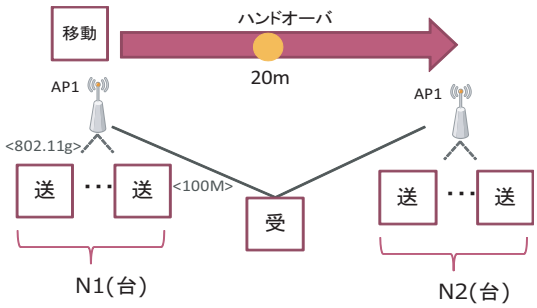

図 3　実験環境 2

使用したバッファサイズ 265.5 パケットの AP(Planex CQW-MR500) は，この実験環境においては，端末数 6 台までが公平，7 台以上が不公平であるため，N1=2，N2=6 とし，移動端末も AP1，AP2 と通信を行うことで，それぞれの AP において，3 台，7 台で通信を行うことになり，公平な環境から不公平な環境へ割り込めるか，また，このような不利な状況でも TCP-AV は帯域を確保することができるのか，検証を行っている．この時，各端末は，uplink 方向に TCP 通信を行っている．

ノート PC 環境において，移動端末のみのスループットの測定結果を図 4 に示す.

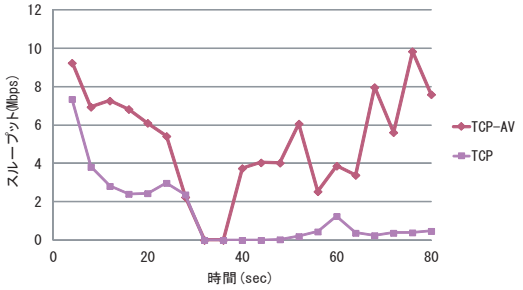

図 4　移動 TCP と移動 TCP-AV の比較

図 4 は，背景端末を全て TCP とし，移動端末を TCP としたときと TCP-AV としたときの結果である．移動端末が TCP であるときは，HO 後に帯域が取れずスループットも上がらないが，移動端末が TCP-AV であるときは，接続先が不公平となる端末数でもスループットをあげることができる．また，AP1 から AP2 へ TCP-AV が HO する前後では，電波が弱くなることによりスループットが減少するが，それ以外では，ある程度の帯域を確保できることが示されている．更に，TCP-AV は，移動端末が TCP である場合に比べて，平均約 20 倍有利に帯域を確保できることが明らかにされている．

この現象について，パケットの解析を行ったところ，背景の TCP 端末において，MAC フレームにおけるリトライアウトによる，TCP データ破棄が起こっていることにより，TCP-AV が割り込めていることが明らかになった．

3.　Android 端末の不公平

本章では，Android 環境における不公平について示す．実験方法としては，図 5 のように，1 台の AP と受信端末を用意し，”送”と表されている送信端末には 8 台の Android 端末を使用した．この実験環境において，Android 端末を同時に通信させて公平に帯域を分け合えるかを検証した．その結果，既存研究と同様の AP を用いて，ノート PC 環境では不公平が起こる 7 台以上である 8 台を同時に通信させても，不公平を確認することができなかった．これは，Android 端末に用いられている TCP が，複数台通信時に公平になるような特性を持っていることが原因の一つであると予想される．

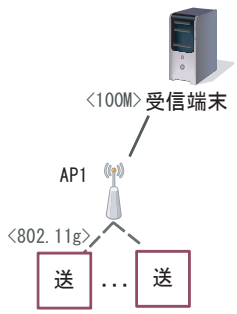

図 5　実験環境 3

4. TCP-AV が割り込める条件

本研究では，Android 環境での QoS-TCP の有効性を検証するため，図 3 の実験環境において，背景端末，及び移動端末に，Android 端末 (Nexus One と Galaxy S:Android2.2, Linux2.6.32.9 使用) を用いて実験を行う．また，AP は，既存研究と同様の AP(Planex CQW-MR500) を使用する．図 3 において TCP-AV が割り込める条件を示すため，N1 と N2 の台数を変化させ，評価には iperf [9] を用いて検証を行った．

4.1 検 討 課 題

TCP-AV が割り込める条件に関わる要素として，

（1） AP と移動端末との距離

（2） AP のバッファサイズ

の 2 点が挙げられる．(1) は，AP と移動端末との距離が遠いほど，ノイズによるビットエラーが発生しやすいため，どの程度の距離からビットエラーが減少し，背景端末に割り込めるのか，検証を行う．(2) は，AP のバッファサイズに応じて，HO 後の既存端末の送信データに対する TCP-ACK あふれが増加するため，パケットロスを引き起こすことで，TCP-AV が割り込める機会があると考えられることから，検証を行う．

4.2 距離による影響

本節では，検討課題の 1 つ目である，AP と移動端末との距離による TCP-AV の割り込み条件について検証する．既存研究と同じ AP(Planex CQW-MR500，265.5 パケット) を使用し，図 2 の実験環境において，N1/N2=2/7，3/6，4/5，5/4，6/3，7/2 と背景端末数を変化させ，それぞれ 8 回の評価を行った. 割り込みが行えた場合の背景端末数と AP までの距離の平均値を表 2 に示す.

表 2　TCP-AV 割り込み条件評価結果

背景端末数 (台)	AP までの距離 (m)
1〜3	2
4〜7	不可

これにより，接続先の背景端末数が 3 台以下であれば，AP から 2m 離れた時点まで，TCP-AV は，割り込み可能となるが，4 台以上だと，どんなに AP と近距離であっても，背景 TCP に勝てないことが示された．このように，ノート PC 環境では HO 後にも割り込めていた TCP-AV が，Android 環境では，背景端末数が少なく，AP に十分近接しないと割り込めないという結果になった．HO 後に割り込めなかった際の結果を図 6 に，割り込めたときの結果を図 7 に示す.

この実験結果により，ノート PC 環境に比べて Android 端末を用いた環境では，TCP-AV でも割り込むことが難しくなっていることが分かる.

また，背景端末数や，移動距離などの具体的な数値は，使用する機器 (例えば，バッファサイズの異なる AP や実験を行う環境) によって，変動があると考えられる.

4.3 AP のバッファサイズによる影響

本節では，検討課題の 2 つ目である AP のバッファサイズの

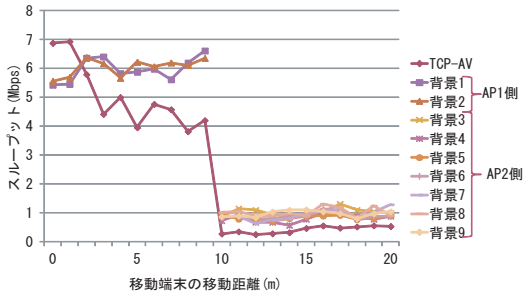

図 6　N1/N2=2/7 の結果

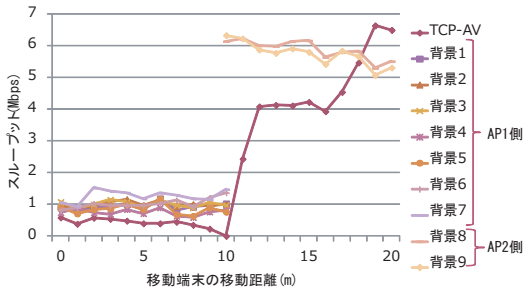

図 7　N1/N2=7/2 の結果

違いによる，TCP-AV の割り込み条件を明らかにする．使用した AP は，バッファサイズが約 135.1 パケットの BUFFALO WHR-HP-AMPG，AP のバッファサイズが約 265.5 パケットの Planex CQW-MR500，AP のバッファサイズが約 284.3 パケットの Planex GW-AP54SAG，AP のバッファサイズが約 566.9 パケットの Planex MZK-MF300N の 4 種類である.

その結果，AP のバッファサイズが大きいほど，移動端末である TCP-AV が既存通信に割り込みにくくなることが分かった．AP のバッファサイズによる TCP-AV の割り込み条件を，表 3 に示す．これは，AP のバッファサイズが小さいと，バッファあふれにより，TCP-ACK が失われやすくなることで背景端末のスループットが低下し，TCP-AV が背景端末の帯域を奪うことが出来て，TCP-AV の割り込みに有利な環境になるからであると考えられる.

表 3　AP のバッファサイズによる割り込み条件評価結果

AP 名	バッファサイズ (パケット)	背景端末数 (台)
BUFFALO WHR-HP-AMPG	135.1	7
Planex CQW-MR500	265.5	1〜3
Planex GW-AP54SAG	284.3	1〜3
Planex MZK-MF300N	566.9	不可

また，TCP-AV が移動距離に関係なく割り込める結果を図 8 に，移動距離に関係なく割り込めない結果を図 9 に示す.

このように，ノート PC 環境と Android 環境で結果に大きな違いが生じた理由については，ノート PC と Android 端末に用いられている TCP の差が原因の一つであると考えられる．2.2 節で述べたように，不公平は MAC 層の送信権制御，トランスポート層における輻輳ウィンドウ制御などが組み合わされ，AP バッファでの TCP-ACK あふれが原因で不公平が起こる.

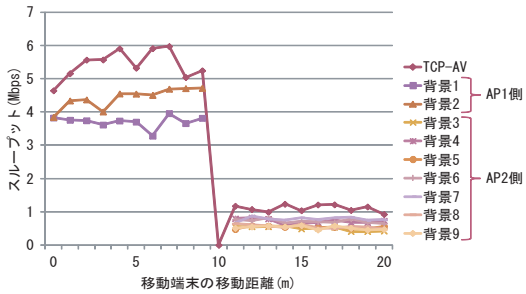

図 8　N1/N2=3/1 の結果

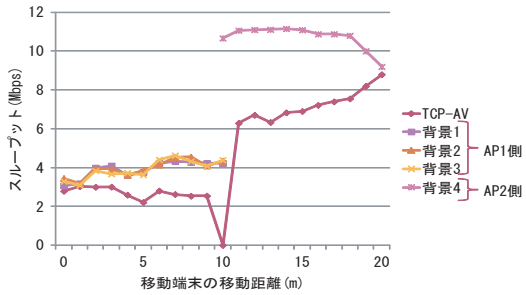

図 9　N1/N2=2/7 の結果

しかし，ノート PC で用いられている TCP(Linux 2.6.32.21-generic) よりも，Android 端末で用いられている TCP(Linux 2.6.32.9) の方が，複数台通信を行った際もパケットロスを発生しにくく，輻輳ウィンドウをあまり下げないような制御になっていることにより，不公平が起こりにくくなっている可能性があると考えられる．

5.　TCP の輻輳制御の違い

先に述べたように，ノート PC 環境において帯域確保可能であった TCP-AV を，Android 環境下に使用すると，不公平や HO 時に割り込める度合いなどの結果に違いがある．Android 環境で移動端末が割り込むすきがない，という現象が起こる理由は，ノート PC の TCP と Android 端末の TCP を比較した場合，Android 端末の TCP の方が複数台通信時に，多少輻輳が起こる環境でも，ウィンドウサイズを下げすぎず，ノート PC の輻輳制御と異なる振舞をすることが理由の一つであると予想される．

TCP の輻輳制御においては，ACK がロスして送信端末に返ってこない場合でも，その次の ACK を受信した送信端末は，輻輳ウィンドウを上げることができる．これにより，ACK が増加するので，ACK が多少ロスしたとしても次の ACK が返る可能性は高く，その後も送信端末が ACK を受信し続けられる．これにより，スループットも低下することなく保つことができる．

一方，ACK がロスして返ってこない場合に，タイムアウトを迎えてしまうと，輻輳ウィンドウは半分まで下がってしまう．このようになってしまった送信端末はパケットを送りだすことが出来ないので，その後も輻輳ウィンドウを上げることができない．

このように，ACK がロスしたときに次の ACK を受信できたかどうかで，輻輳ウィンドウを上げることのできる端末とできない端末が現れてしまうことにより，スループットに不公平が生じてしまう．ノート PC 環境ではこの現象が観測されている．

しかし，Android 環境では，ACK がロスしてタイムアウトを迎えても，輻輳ウィンドウの値を 1 以上の値に保ちやすくなっているため，無駄に輻輳ウィンドウを下げることなく，常に効率的にパケットを送信することが出来ていると考えられる．この仮説は，輻輳ウィンドウの比較により，示される．

そこで，図 5 のように AP，受信端末を 1 台ずつ用意し”送”と表されている送信端末を複数台同時に uplink 方向に通信させた際の輻輳ウィンドウの比較を行った．送信端末が，ノート PC の場合と Android 端末の場合の 2 通りで実験を行い，比較した．

実験結果を図 10 と図 11 に示す．本実験では，輻輳ウィンドウの解析に，カーネルモニタ [10] や Web100 [11] を利用した．これらは，通信時に，カーネル内部のパラメータの値がどのように変化したかを記録する事ができるツールである．Android 端末，ノート PC 共に，使用する端末以外，端末数や AP などの環境は，同じ実験環境で実験している．しかし，図 11 に示した Android 環境の輻輳ウィンドウをみると，ノート PC よりも輻輳ウィンドウを下げる機会が少ないことが分かる．一方，図 10 に示すノート PC の輻輳ウィンドウは，多少の輻輳にも過剰に反応し，すぐに輻輳回避のフェーズに入っていることが分かる．TCP-AV は，ノート PC 環境では，ノート PC が非効率的にパケットを送信している分の帯域を奪い，既存端末に割り込むことが出来るが，Android 端末は，全端末が効率的にパケットを送信しているため，なかなか既存端末に割り込むことが出来ないと言える．つまり，Android 端末の TCP の方が輻輳ウィンドウをあまり下げないように改善されていることから，ノート PC 環境のように，不公平が起こらず, 移動端末が帯域を奪うことができない場合があるといえる．Android 端末でこのような輻輳制御になっている理由としては，ノート PC の場合，通信は無線と有線の両方が想定され，有線環境であまりアグレッシブにパケット送出を行うと輻輳崩壊を引き起こす可能性があるのに対し，Android 端末の場合は，通信は無線環境のみが想定されていることから，無線通信のみに特化した制御を行うことが可能であるためと考えられる．

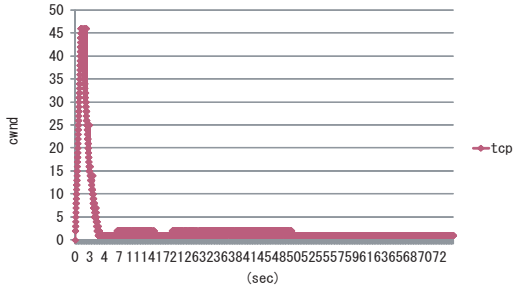

図 10　ノート PC8 台の通信時の輻輳ウィンドウ

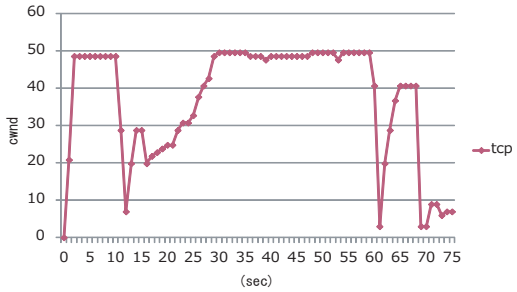

図 11　Android 端末 8 台の通信時の輻輳ウィンドウ

6. おわりに

デスクトップ PC やノート PC を用いた，HO を伴う移動通信環境で有効性が示されている QoS-TCP(本実験では TCP-AV) を，Android 端末を用いた環境に利用し，QoS-TCP が既存通信に割り込める度合いは，AP と移動端末間の距離や AP のバッファサイズに依存することを示した．具体的には，TCP-AV を用いても，接続先の端末数が少なく，かつ，AP との距離が十分に近接していないと，背景端末の通信に割り込むことができないことが分かった．また，AP のバッファサイズが小さいほど，移動端末が割り込める．これは，TCP-AV が割り込める際には，割り込めない時と比較し，多くの MAC フレームエラーが背景端末に発生しているためである．更に，TCP-AV が既存通信に割り込めない理由としては，ノート PC 環境の輻輳ウィンドウと Android 端末の TCP の輻輳ウィンドウの比較により，Android 端末の輻輳ウィンドウは, 輻輳時にもウィンドウサイズを下げすぎないようになっていることを明らかにした．

今後の課題としては，Android 環境で TCP-AV が TCP に割り込めない理由について，更に詳しく解析を行いたい．具体的には，Android 端末に使用されている TCP-CUBIC と TCP-Reno ベースで実装されている TCP-AV の TCP の振舞の違いを検証したい．これにより，ノート PC 環境で有効性が示されている QoS-TCP を，Android 環境で有効に利用するための改善案を提案したい．

文　献

[1] R.Ando, T.Murase, and M.Oguchi. ”Characteristics of QoS-Guaranteed TCP on Real Mobile Terminal in Wireless LAN,” In Proc. IEEE 2011 International Communications Quality and Reliability Workshop (CQR2011),May 2011.
[2] H. Shimonishi,et al.,”Congestion Control Enhancements for Streaming Media,”IEICE Transactionson Communications.,Vol.E89B,No.9, pp.2280-2291,Sep.2006.
[3] 新井絵美，平野由美，村瀬勉，小口正人：無線 LAN 環境における実機特有の帯域公平性についての検討と QoS 保証 TCP の性能評価，DBSJ Journal，Vol8,No.1, 2009 年 6 月.
[4] 安藤玲未，村瀬勉，小口正人：無線 LAN の様々な条件における帯域公平性の検証と QoS 保証 TCP の性能評価，マルチメディア，分散，協調とモバイル (DICOMO2010) シンポジウム，8E-3， pp.1898-1904, 2010 年 7 月.
[5] BUFFALO WHR-HP-AMPG: http://buffalo.jp/products/catalog/item/w/whr-hpampg/
[6] Planex CQW-MR500: http://www.planex.co.jp/product/router/cqw-mr500/
[7] Planex GW-AP54SAG: http://www.planex.co.jp/product/bwave/gwap54sag.shtml
[8] Planex MZK-MF300N: http://www.planex.co.jp/product/router/mzk-mf300n/
[9] Iperf:http://sourceforge.net/projects/iperf/files/iperf/2.0.4source/iperf-2.0.4.tar.gz/download
[10] 山口実靖，小口正人，喜連川優：iSCSI 解析システムの構築と高遅延環境におけるシーケンシャルアクセスの性能向上に関する考察，電子情報通信学会論文誌，Vol.J87-D-I，No.2，pp.216-231，2004 年 2 月.
[11] Web100：http://www.net100.org/