# アプリケーションに依存する Android通信制御パラメータの振舞の解析

熊谷菜津美† 平井 弘実† 三木香央理† 山口 実靖†† 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
†† 工学院大学 〒 163-8677 新宿区西新宿 1-24-2
E-mail: †{natsumi,hiromi,kaori}@ogl.is.ocha.ac.jp, ††sane@cc.kogakuin.ac.jp, †††oguchi@computer.org

**あらまし** 近年，スマートフォンユーザが急速に増加している．中でも Google 社開発の Android は，ソースコードがオープンであり自由に開発できるため，シェア率が高く注目を集めている．Android 端末上でユーザが利用するのはアプリケーションであるが，このアプリケーションを用いて様々な場面でネットワークに接続できるようになったことで，モバイル環境においても新しい情報がリアルタイムに得られるようになった．そこで本研究では，利用するアプリケーションによって通信の挙動にどのような特性が表れるかについて解析する．さらに，アプリケーションの通信特性に応じて制御を行い，通信性能の向上を目指す．

**キーワード** Android，アプリケーション，リアルタイム通信，ファイル転送，アップロード

# Analysis of Application-Dependent Behavior of Android Communication Control Parameter

Natsumi KUMATANI†, Hiromi HIRAI†, Kaori MIKI†, Saneyasu YAMAGUCHI††, and Masato OGUCHI†

† Ochanomizu University 2-1-1 Otsuka, Bunkyou-ku, Tokyo, 112-8610, JAPAN
†† Kogakuin University 1-24-2 Nishi-shinjuku, Shinjuku-ku, Tokyo,163-8677, Japan
E-mail: †{natsumi,hiromi,kaori}@ogl.is.ocha.ac.jp, ††sane@cc.kogakuin.ac.jp, †††oguchi@computer.org

## 1. はじめに

近年，従来の携帯電話からスマートフォンへのシフトが急速に進み，今ではなくてはならないモバイル端末になりつつある．各キャリアによる新製品の多くがスマートフォンになり，スマートフォンに対するユーザの関心も高まっている．従来の携帯電話で主に使われてきた音声通話やメール機能に加えて，スマートフォンではパソコンとしての機能を有しているため，時間や場所を問わずインターネット機能を利用できるようになった．また従来の携帯電話におけるネットワークアプリケーションは，通信回線に 3G などの広域無線アクセスのみを利用していたため，通信帯域が狭く，大きなデータをやり取りするアプリケーションは実行するのが難しかったが，スマートフォンでは一般に無線 LAN を使うことができるため，広帯域な通信を必要とするアプリケーションも実行可能となり，アプリケーションのバリエーションが大幅に広がった．

これにより，スマートフォンに搭載されているアプリケーションを利用することで，ニュースサイトやソーシャルネットワーキングサービスを通じて，手軽に日々の情報を知ることができ，反対にユーザ自身からリアルタイムに情報を発信できるようにもなった．またスマートフォンは初めから搭載されているアプリケーション以外に，好みのアプリケーションを後から追加できるためカスタマイズの自由度が高い．

現在，数多くのスマートフォン OS が登場しているが，その中でもトップシェアを占めているものが Android [1] である．Android は Google 社によって開発されたソフトウェアプラットフォームであり，主にスマートフォンやタブレット PC などのモバイル端末に搭載されている．Android が持つ特徴としては，ソースコードをオープンソースとして無償で提供している点が挙げられる．そのため一般ユーザでも自由に開発を進められ，開発コストの削減に繋がるという点から注目されている．そこで本研究ではこの Android を研究対象とした．特にその無線 LAN 通信性能に着目して研究を行い，データの通信特性が異なるアプリケーションを実行した際の通信の振舞を解析していく．

## 2. Android アーキテクチャ

図 1 に Android のアーキテクチャを示す．5 つの層に分かれており，下がハードウェアに近い層で上にいくほどユーザインターフェイスに近い層となっている．Android OS の中心部分は Linux であり，この部分も併せて Google から提供されている．

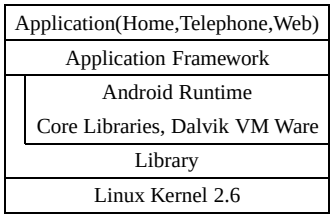


図 1　Android のアーキテクチャ

基礎部分には Linux2.6 カーネルを採用しており，この OS に各種コンポーネントを追加し Android というプラットフォームを構築している．Linux カーネルはハードウェア上でソフトウェアの最下層に位置し，ドライバによってデバイスの機能を Linux から利用できるようになる．Linux カーネルの上にはライブラリがあり，機能ごとにまとめられた汎用性の高いプログラム群をまとめている．このライブラリの機能は，アプリケーションフレームワークを経由して複数のアプリケーションから利用できるようになっている．Android ランタイムは Android 独自の仮想マシンである Dalvik と基本的な API を提供するコアライブラリで構成されている．Android のアプリケーションは全てこの Dalvik VM 上で動作している．その上にアプリケーションフレームワークがあり，アプリケーション動作に関係する，共通に利用される API を規定している．さらにこのアプリケーションフレームワーク上にアプリケーションがある．アプリケーションは Java で開発され，アプリケーションフレームワーク上で自由に構築することができる．

## 3. 研 究 目 的

これまで Android に関する研究において，複数台の Android 端末を同一アクセスポイントに接続し通信させた際に，安定した通信が行える場合と不安定な通信を行う場合があることが確認されている [2]．そこで，本来なら安定して十分な通信帯域を確保できる状況にもかかわらず，不安定な通信状況が確認された場合には，輻輳ウィンドウサイズを下げないように変更した独自の TCP に切り換えることにより，全ての端末上で安定した通信を行うための研究が行われている [3].

またこれを実現するために，同一アクセスポイント内の Android 端末間で互いの通信状況を通知し合い，他端末の通信状況を把握することで，その時の通信状況に合わせた通信制御を行うよう TCP を切り替えるためのミドルウェア開発が進められている [4].

Android 端末上でユーザが実際に利用する部分はアプリケーションであるため，各アプリケーションの通信特性にも応じて通信制御を行うことで，より良く通信帯域の利用が可能になり安定した通信ができると考えられる．そこで本研究では各アプリケーションの通信特性の違いに着目し，Android 端末上でアプリケーションを実行している際のカーネル内部の振舞を調べ，各アプリケーションの通信特性に応じた制御を行うことにより，より良い無線 LAN 通信環境の実現を目指す.

## 4. Android アプリケーション

### 4.1 アプリケーション通信特性

Android アプリケーションは Android Market に数多く登録されている．ユーザはこの Android Market から様々なアプリケーションをダウンロードし利用しているが，実行するアプリケーションによってデータの通信特性が異なっていると考えられる．データの通信方向により大きく分けると，主に情報のアップロードを行うアプリケーション，反対に求める情報をクラウドサーバからダウンロードすることを中心に行うアプリケーション，そしてその両方の機能を持つ双方向通信のアプリケーションの 3 つに分けられる．また扱うデータによっても通信方法は異なり，音声や映像などのデータを即時に送受信したい場合はリアルタイム通信を行い，単に写真や動画などのデータを送受信する場合はリアルタイム性の無いファイル転送を行っている．

### 4.2 実験対象アプリケーション

従来の携帯電話では，サイトからデータをダウンロードして楽しむ，という使い方が一般的だが，スマートフォンでは，端末およびネットワークアクセス性能の向上により，大きなデータのアップロードを必要とするアプリケーションも動作できるようになったことから，アプリケーションもアップロードを行うものが増えている．同一アクセスポイント内の複数台の Android 端末がアップロード通信を行う場合，端末同士で通信帯域を取り合うため，この通信帯域をうまく分け合うことで通信性能が向上すると考えられる．

そこで本研究で使用するアプリケーションとしては，主にデータのアップロードを行うものに焦点を当て，リアルタイム通信アプリケーションとファイル転送アプリケーションを選択した．一般に，リアルタイム通信には UDP 通信を行う場合が多いが，ファイアウォールを超えやすいなどの理由から TCP を使っているものもあり，本研究では TCP 通信を行うリアルタイム通信アプリケーションを実験に用いた．表 1 に実験対象アプリケーションを示す．Ustream では Android 端末側から動画のストリーム配信を行った．ES file explorer では自分で設定した接続先のローカルサーバへ約 67MB の動画ファイルを転送した．もう 1 つのファイル転送アプリケーションである Picasa Tool では，ウェブサーバへ 10 枚の計 12MB 程度の画像ファイルを転送した.

表 1　実験対象アプリケーション

| | |
|---|---|
| リアルタイム通信 | Ustream |
| ファイル転送 | ES file explorer |
| | Picasa Tool |

## 5. オリジナルシステムツール

本章では，我々が開発したオリジナルシステムツールについて説明する．

### 5.1 カーネルモニタ

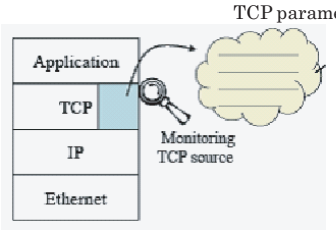


図 2　カーネルモニタ

本研究には，カーネルモニタというツールを用いている．図 2 に示すように，カーネル内部の TCP ソースにモニタ関数を挿入し再コンパイルすることで，通信時のカーネル内部のパラメータの値の変化の様子をログとして記録し出力できる．取得したログには，輻輳ウィンドウサイズ，スロースタート閾値，タイムスタンプ，ソケットバッファのキュー長などの TCP パラメータが含まれており，さらに，各種エラーイベント (Local device congestion，重複 ACK，SACK 受信，タイムアウト検出) がどのタイミングで発生したのかをモニタできるようになった．本研究では，このカーネルモニタを Android 環境に移植したツールを用いて，アプリケーション実行時の輻輳ウィンドウの遷移の様子を解析した．

### 5.2 通信性能の可視化ツール

輻輳ウィンドウの遷移を解析するために，独自に開発した通信性能の可視化ツールを利用した[5]．この可視化ツールは，カーネルモニタによって得られた輻輳ウィンドウサイズがどのように遷移するかを，Android 端末上で可視化し解析できるツールである．

## 6. 実 験 環 境

表 2 に本実験で使用した実験環境を示す．Android 端末は Linux2.6.35 をベースとした Android2.3 であるが，前述のようにカーネルモニタを組込んで再コンパイルしたカーネルを用いている．

表 2　Experimental Environment

| | | |
|---|---|---|
| Android | Model number | Nexus S |
| | Firmware version | 2.3.4 |
| | Baseband version | I9023XXKB1 |
| | Kernel version | 2.6.35.7-kaori1198-ge382d80-dirty |
| | Build number | GRJ22 |
| Server | OS | Fedora 15 |
| | CPU | Intel Celeron(R) 2.30GHz |
| | Main Memory | 2GB |

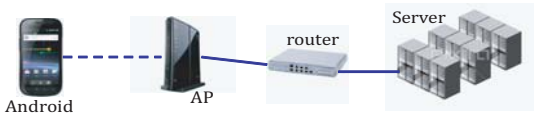


図 3　1 台の Android 端末通信時の実験環境

## 7. 1 台の Android 端末通信時の性能

図 3 は，Android 端末 1 台を IEEE802.11g 無線 LAN 機能を用いてアクセスポイントに接続し，サーバに対して通信を行った際の実験環境である．各アプリケーションを実行させた Android 端末 1 台のみを通信させた場合の輻輳ウィンドウを測定した．この測定結果を図 4〜6 に示す．縦軸は輻輳ウィンドウサイズ，横軸は時間を示している．図 4 は Ustream 実行時の測定結果であり，1 分間のストリーム配信を行っている．図 5 は ES file explorer 実行時の測定結果であり，67MB のファイルを転送している．図 6 は Picasa Tool 実行時の測定結果であり，10 枚の 12MB 程度の画像ファイルを転送している．Ustream と ES file explorer は輻輳ウィンドウサイズを安定して保つことができているが，Picasa Tool は細かく増減を繰り返していることがわかる．また，ローカルにサーバを置いた ES file explorer とウェブサーバへファイル転送する Picasa Tool を比較すると，Picasa Tool のほうは輻輳ウィンドウサイズがあまり上がらないことが確認できた．以降の実験においては，アプリケーションの特徴がよく現れている Ustream と ES file explorer を用いる．

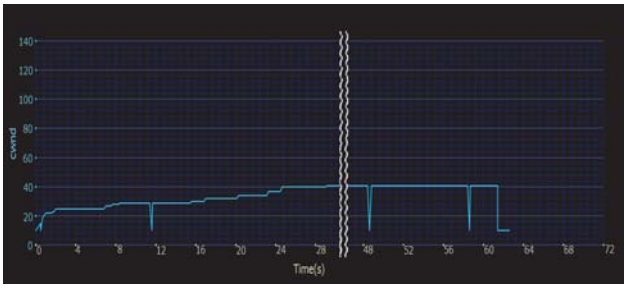


図 4　Ustream 実行時の輻輳ウィンドウサイズ

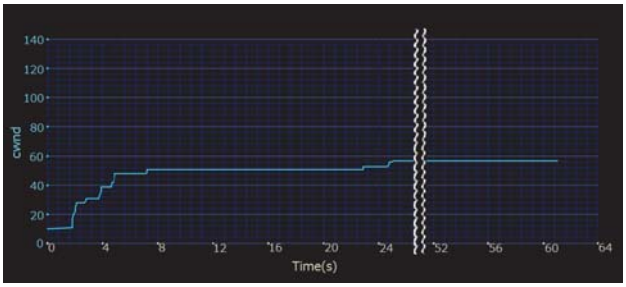


図 5　ES file explorer 実行時の輻輳ウィンドウサイズ

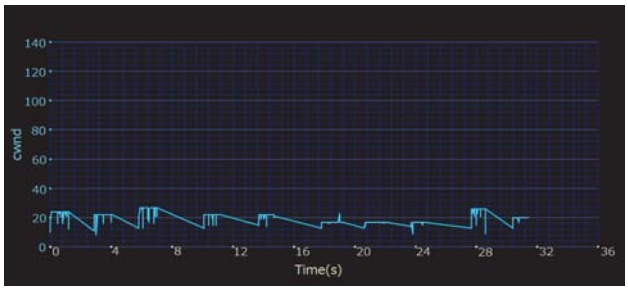


図 6　Picasa Tool 実行時の輻輳ウィンドウサイズ

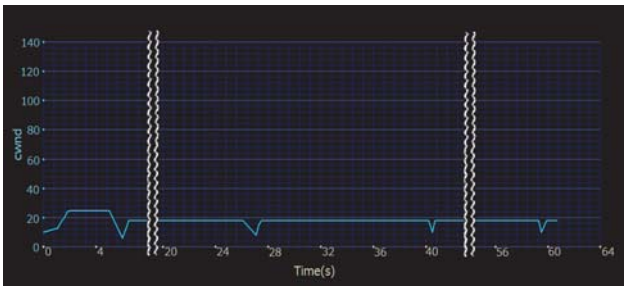


図 9　Android1 の輻輳ウィンドウサイズ

## 8.　複数台の Android 端末通信時の性能

### 8.1　Android 端末 4 台の同時通信

次に，図 7 に示すようにアプリケーションが混在した複数台の Android 端末を同一アクセスポイントに接続し通信させて実験を行った．Android 端末 2 台通信から始め，1 台ずつ増やしていき，4 台通信まで測定した．Android 端末上で実行するアプリケーションには，Android1 のみリアルタイム通信アプリケーションを用いて，残りの Android 端末ではファイル転送アプリケーションを実行した．ファイル転送アプリケーションには全て ES file explorer を使用した．

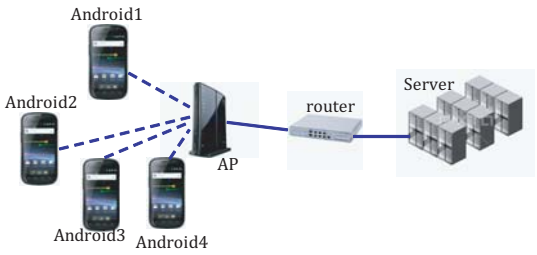


図 7　4 台の Android 端末通信時の実験環境

図 8，9 にリアルタイム通信を行う Android1 の輻輳ウィンドウの測定結果の例を示す．

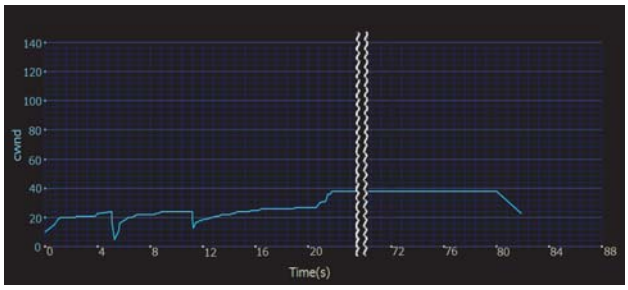


図 8　Android1 の輻輳ウィンドウサイズ

図 8，9 に示されるように，1 台の Android 端末で通信した時と同程度まで上昇し，高い値を保てる場合と，同一環境下にあるにもかかわらず，輻輳ウィンドウが上昇しきれない場合の 2 パターンに分かれることが確認された．また，ファイル転送を行う Android2～4 の輻輳ウィンドウは，1 台の Android 端末で通信した時と同程度まで上昇し，同一アクセスポイントに繋がって通信を行う端末数が 4 台まで増えても影響はほとんどないことが確認できた．

### 8.2　ファイル転送端末の輻輳ウィンドウの上限値抑制

リアルタイム通信端末のみ輻輳ウィンドウが上昇しきれない場合があることから，リアルタイム通信端末が優先的に通信を行うために，図 7 に示すものと同じ実験環境において，ファイル転送端末に Linux カーネルの輻輳制御部分の一部を修正した独自の TCP [2] を用いて，輻輳ウィンドウサイズの上限値を抑えて実験を行った．

図 10 に上限値を 30 まで抑えたときの，リアルタイム通信を行う Android1 の輻輳ウィンドウサイズの測定結果を示す．また図 11 に 4 台全ての端末に修正を行っていない通常の TCP の Android 端末を用いた場合とファイル転送端末の輻輳ウィンドウの上限値を 30 まで抑えた場合のファイル転送に掛かる時間の比較を示す．図 10 から，輻輳ウィンドウサイズが高い値を保てる場合と同程度まで上昇していることが確認され，ファイル転送端末の輻輳ウィンドウサイズの上限値を抑えることで，リアルタイム通信端末の輻輳ウィンドウを上げられることが確認できた．しかし図 11 から，ファイル転送端末の上限値を抑えたことで，ファイル転送に掛かる時間が長くなってしまうことが確認された．

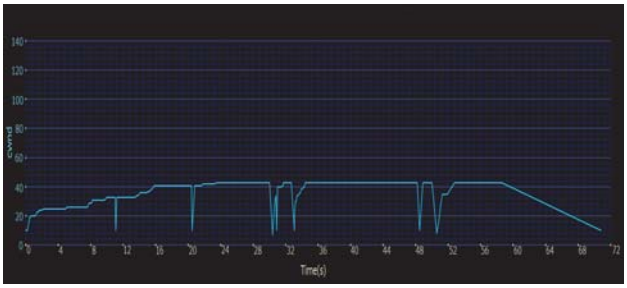


図 10　ファイル転送端末の上限値が 30 の時，Android1 の輻輳ウィンドウサイズ

### 8.3　リアルタイム通信端末の輻輳ウィンドウの上限値抑制

そこでファイル転送にかかる時間を短縮するために，図 7 の実験環境において，ファイル転送端末 (Android2～4) は通常の TCP のままで，リアルタイム通信端末 (Android1) に独自の TCP を用いて輻輳ウィンドウの上限値を抑えて実験を行った．4 台全て通常の TCP を用いた Android 端末用いた場合と Android1 の上限値を 20 まで抑えた場合の，ファイル転送にかかる時間の比較を図 12 に，Android1 で動画配信してから PC 上に流れ

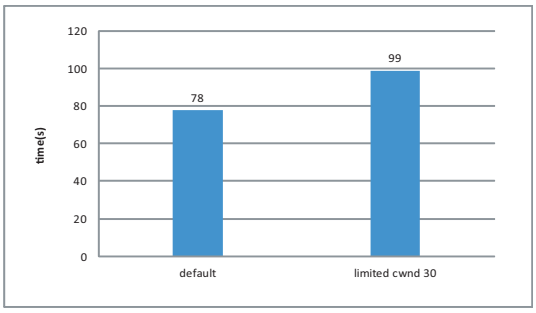


図 11　通常の TCP のみの場合とファイル転送端末に独自の TCP を用いた場合のファイル転送に掛かる時間比較

るまでにかかる時間の比較を図 13 に示す.

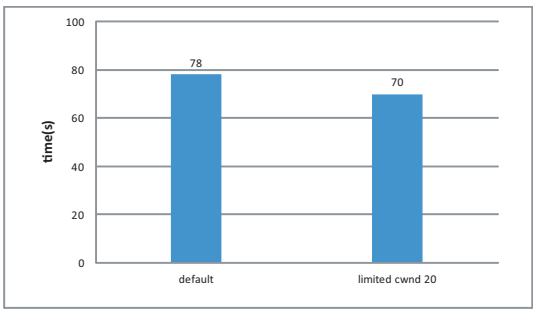


図 12　通常の TCP のみの場合とリアルタイム通信端末に独自の TCP を用いた場合のファイル転送に掛かる時間比較

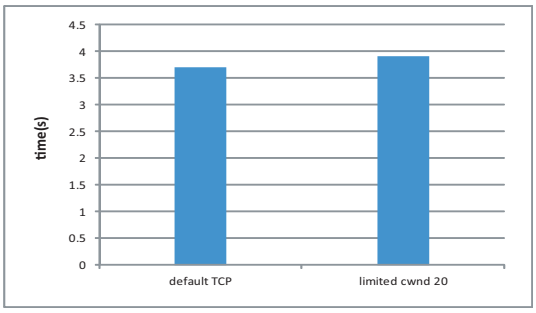


図 13　通常の TCP のみの場合とリアルタイム通信端末に独自の TCP を用いた場合の配信動画が PC 上に流れるまでの時間比較

図 11 から上限値を 20 まで抑えた場合，ファイル転送にかかる時間が短縮されることが確認された．また図 13 に示されるように，動画配信してから PC 上に映し出されるまでの時間はほとんど差がなく，配信動画の見え方にも違いが見られなかったことから，本研究の実験環境ではリアルタイム通信端末の輻輳ウィンドウの上限値を抑えても，配信動画の品質に影響なくファイル転送時間を短縮できることが確認された.

## 9.　通信状況の分からない他端末と Android 端末の協調した通信

前章までは同一アクセスポイントにつながる周囲の端末の通信状況が確認できる環境において通信性能の測定を行っていたが，この章では通信状況の分からない PC 端末と Android 端末が通信帯域を分け合う環境で実験を行った．具体的には 2 台の PC が通信している環境に Android 端末が入る状況を調べた．今回の環境においては，PC と比較して，Android 端末が通信帯域を取りすぎてしまうことから，Android 端末がのパケット送出量を抑えて同一アクセスポイント内の全端末で協調した通信を行った.

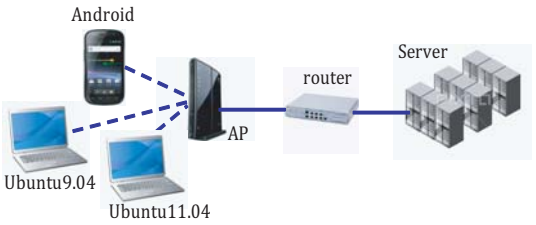


図 14　Android と PC を用いた実験環境

図 14 に実験環境を示す．1 台の Android 端末と 2 台の PC を同一アクセスポイントに接続してサーバに対して通信させた．このとき Android 端末では Ustream を実行させ，ubuntu9.04 と ubuntu11.04 が動作する 2 台の PC では iperf-2.0.4 [6] を用いてサーバへ通信を行っている.

### 9.1　測 定 環 境

4 種類の環境を用いて PC2 台の性能を測定した．比較した環境は，default TCP の Nexus S と PC2 台を用いた場合，Android と PC 間で TCP の公平性を保つために，Nexus S に比べてパケット送出の積極性の弱い HT-03A のカーネルを移植した Nexus S と PC2 台を用いた場合，輻輳ウィンドウの上限値を 20 まで抑えた場合にも Ustream の配信動画の品質に影響がなかったことから，HT-03A のカーネルを移植しさらに上限値を 20 まで下げてパケット送出量を抑えた Nexus S と PC2 台を用いた場合，Android 端末を含まず 2 台の PC のみ用いた場合の 4 パターンを比較した.

### 9.2　2 台の PC の性能比較

図 15 に 2 台の PC のスループットを示す.

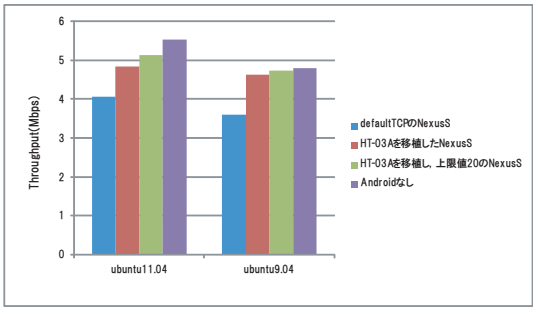


図 15　2 台の PC のスループット比較

どちらの PC も，Android 端末にパケット送出が積極的な default TCP の Nexus S を用いた場合が 1 番スループットが低いが，Android 端末を HT-03A のカーネルを移植した Nexus S に切り替えることでスループットが向上することが確認された．またパケット送出が 1 番消極的な，HT-03A のカーネルを移植

しさらに上限値を 20 まで下げた Nexus S を用いた場合には，Android 端末を含まず 2 台の PC のみ通信させた場合のスループットに近い値まで向上することが確認された．また Nexus S に 3 パターンのどのカーネルを用いた場合でも，配信動画の品質に問題なく Ustream を実行できていたことから，Android 端末のパケット送出を抑えることで，周囲の端末の通信状況を確認できない場合にも端末間で協調性を保った通信を行えることが確認できた．

## 10. まとめと今後の課題

本論文では様々な環境において，通信特性の異なる各アプリケーションを実行した Android 端末を通信させた際の輻輳ウィンドウを測定し解析を行った．その結果，各アプリケーションによって輻輳ウィンドウの遷移の様子が異なることを確認した．複数台の異なる通信特性のアプリケーションを同時通信させた場合，リアルタイム通信端末の輻輳ウィンドウの上限値を抑えることで，ファイル転送にかかる時間を縮めることができた．また，同一アクセスポイント内に通信状況を確認できない端末が存在した場合も，Android 端末のカーネルを切替えることで協調した通信を行えることが確認できた．

今回の実験の範囲では，ファイル転送による帯域の使用が Ustream による動画配信の品質に影響を及ぼさない範囲であったと考えられる．そこで今後は，ファイル転送に負荷をより高くし，Ustream の配信動画の品質に影響を及ぼす環境を調べて，その際にリアルタイム通信を行う Android 端末の通信性能の向上を目指していく．

## 文　　献


[1] Android:http://www.google.co.jp/mobile/android

[2] 三木香央理，山口実靖，小口正人: カーネルモニタを用いた Android 端末の無線 LAN 通信性能の解析と性能向上のための一検討，DICOMO2011，7H-2，2011 年 7 月．

[3] 三木香央理，山口実靖，小口正人: 無線 LAN 通信環境におけるカーネルモニタを用いた TCP 解析による Android 端末の性能向上手法，DEIM2012，C6-5，2012 年 3 月．

[4] 平井弘実，三木香央理，山口実靖，小口正人: Android 端末を用いた無線通信時の制御ミドルウェアに関する考察，電子情報通信学会 NS 研究会，NS2011-105，2011 年 11 月

[5] 平井弘実，三木香央理，山口実靖，小口正人: Android 端末における通信性能の可視化ツール，情報処理学会第 73 回全国大会，5V-9，2011 年 3 月．

[6] Iperf:http://downloads.sourceforge.net/project/iperf/iperf/2.0.4