

# Android 端末におけるアプリケーションに依存する通信特性に関する一検討

熊谷 菜津美<sup>†</sup> 平井 弘実<sup>†</sup> 三木 香央理<sup>†</sup> 山口 実靖<sup>‡</sup> 小口 正人<sup>†</sup>  
<sup>†</sup>お茶の水女子大学 <sup>‡</sup>工学院大学

## 1. はじめに

近年、スマートフォンが急速に普及している。従来の携帯電話の機能に加えて、スマートフォンはコンピュータとしての機能を併せ持つため時間や場所を問わず手軽にインターネット機能を利用できるようになった。本研究では、スマートフォン OS 中でもトップシェアを占めている Android を研究対象とし、特にその無線 LAN 通信性能に着目して研究を行った。

## 2. Android

Android とは Google 社開発のソフトウェアプラットフォームである。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 のアーキテクチャにはその基礎部分に Linux2.6 カーネルを採用している。ソースコードがオープンソースとして無償で提供されているため、ユーザが自由に開発を進められ、開発コストを削減できる点から注目を集めている。

## 3. 研究目的

現在、Android に関する研究で、同一アクセスポイント (AP) に繋がる他端末の通信状況に応じて通信制御を行うことで通信帯域をより良く利用するための研究が行われている。また、Android 端末間で互いの通信状況を通知し合い他端末の状況を把握することで、環境に合った通信制御を行うためのミドルウェア開発が進められている。

Android 端末上でユーザが直接利用している部分はアプリケーションであることから、各アプリケーションの特性にも応じて通信制御を行うことで、より良い帯域の利用が可能になると考えられる。本研究ではアプリケーションの通信状況の違いに着目し、Android のカーネル内の振舞いを調べてアプリケーションの特性に応じた通信制御を行うことで、より良い無線 LAN 通信環境の実現を目指す。

## 4. オリジナルシステムツール

本実験で利用したオリジナルシステムツールを示す。

### 4.1 カーネルモニタ

カーネルモニタは、カーネル内部の情報を記録し出力できるオリジナルシステムツールである。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、輻輳ウ

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 パラメータがモニタ可能になる。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いる。

### 4.2 通信性能の可視化ツール

また、カーネルモニタを用いて取得した TCP パラメータを Android 端末上で可視化できるツールを利用した [1]。本研究では、このツールを用いて輻輳ウィンドウサイズの遷移を可視化した。

## 5. 実験環境と実験対象アプリケーション

本研究の実験環境を表 1 に、実験対象としたアプリケーションを表 2 に示す。本研究ではデータ通信においてアップロードが主となるアプリケーションを選択し、その中でもリアルタイム通信とファイル転送を行うアプリケーションを実験対象とした。Ustream では動画配信の機能を利用した。ES file explorer では sftp 転送機能を利用し、自分で設定した接続先のローカルサーバへファイル転送を行った。もう 1 つのファイル転送アプリケーションとしては Picasa Tool を用いて、Picasa ウェブサーバへ画像ファイルのアップロードを行った。

表 1: 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                               |

表 2: 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      |

## 6. 1 台の Android 端末通信時の輻輳ウィンドウ測定

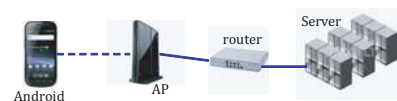


図 1: 1 台の Android 端末通信時の実験環境

まず、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の振舞いを測定した。図 1 に示すように、Android 端末 1 台を IEEE802.11g

A Study of Communication Characteristic of Android Terminals Depending on Applications

<sup>†</sup>Natsumi Kumatani, <sup>†</sup>Hiroshi Hirai, <sup>†</sup>Kaori Miki, <sup>‡</sup>Saneyasu Yamaguchi, and <sup>†</sup>Masato Oguchi  
 Ochanomizu University (<sup>†</sup>)  
 Kogakuin University (<sup>‡</sup>)

無線 LAN 機能を用いて AP に接続し、その先に有線接続されたサーバに対して通信を行った。Ustream では1分間ストリーム配信させてリアルタイム通信を行ったところ、輻輳ウィンドウは約 40 まで上昇することがわかった。ES file explorer では 67MB のファイル転送を行ったところ、輻輳ウィンドウは約 60 まで上昇することがわかった。Picasa Tool では 10 枚の 12MB 程度の画像ファイルを転送させて ES file explorer と同様にファイル転送を行ったが、輻輳ウィンドウは約 25 までしか上がらないことがわかった。

## 7. 複数台の Android 端末通信時の輻輳ウィンドウ測定

### 7.1 Android 端末 4 台での通信

次にアプリケーションが混在した複数台の Android 端末を同一 AP に接続し通信させた場合の輻輳ウィンドウを測定した。Android 端末 2 台の通信から始め、4 台通信まで測定を行った。図 2 に 4 台の Android 端末による通信時の実験環境を示す。Android1 のみリアルタイム通信アプリケーションを実行し、残りの Android2, 3, 4 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 ES file explorer を利用した。

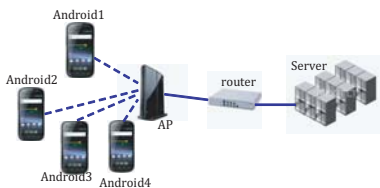


図 2: 4 台の Android 端末通信時の実験環境



図 3: Android1 の輻輳ウィンドウ (デフォルト値)

図 3 は 4 台通信での Android1 の測定結果である。輻輳ウィンドウは 20 前後までしか上がらず、1 台通信の場合より低下している。また、配信動画の画質も低下したことが感じられた。残りの Android2, 3, 4 は全て 1 台通信の場合と同様に約 60 まで上昇した。この結果から、同一 AP に接続して通信を行う端末数が増えてもファイル転送端末の輻輳ウィンドウには影響はないが、リアルタイム通信端末は輻輳ウィンドウが下がってしまうことがわかった。

### 7.2 輻輳ウィンドウサイズ上限値を抑えた Android 端末を含む通信

前節と同じ図 2 に示す実験環境において、Android2, 3, 4 の輻輳ウィンドウが 30 までしか上昇しないように設定し実験を行った。この時の Android1 の輻輳ウィン

ドウを図 4 に示す。輻輳ウィンドウは約 40 まで上昇し 1 台通信の場合と変わらない結果が得られた。一般にファイル転送には多少時間がかかっても、より早く良い画質でストリーム配信できることが望まれると考えられるが、この結果から、ファイル転送実行端末の上限値を抑えることで、リアルタイム通信実行端末の輻輳ウィンドウを上げることができたといえる。



図 4: Android1 の輻輳ウィンドウ (輻輳制御変更時)

## 8. Android 端末と wireless PC による通信時の輻輳ウィンドウ測定

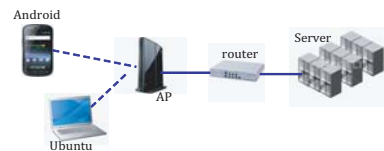


図 5: Android 端末と wireless PC による実験環境

次に図 5 に示すように Android 端末と wireless PC を 1 台ずつ AP に接続して実験を行った。Android 端末上では各アプリケーションを実行させて、ubuntu9.04 が動作する PC 上では iperf-2.0.4[2] を用いてサーバへ通信を行っている。Ustream を実行した場合、輻輳ウィンドウは約 30 まで低下することがわかった。現在、他のファイル転送アプリケーションを実行させて測定を行っている。

## 9. まとめと今後の課題

データのアップロードを行う各アプリケーションを実行した Android 端末の輻輳ウィンドウを測定した。実行する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がわかった。また複数台の異なるアプリケーションが動作する Android 端末が通信している環境では、リアルタイム通信実行端末のみ輻輳ウィンドウが低下するが、ファイル転送実行端末の輻輳ウィンドウサイズの上限値を 30 まで抑えることで低下した輻輳ウィンドウを上げられることがわかった。

今回の実験には全て同じ Android 端末を使用しているが、今後は異なる機種 of Android 端末も使用した場合の輻輳ウィンドウを測定していきたい。

### 参考文献

- [1] 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.
- [2] Iperf: <http://downloads.sourceforge.net/project/iperf/iperf/2.0.4>