Android 端末におけるアプリケーションに依存するカーネル内の通信特性

熊谷 菜津美　(指導教員：小口 正人)

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んでいる．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，いつどこでも手軽にインターネット接続を利用できるようになった．現在，スマートフォン OS の中でトップシェアを占めているものが Google 社開発の Android[1] である．Android はソースコードが無償で提供されているため，誰でも自由に開発を進められる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，その無線 LAN 通信性能に着目して研究を行い，アプリケーション実行時の通信の振舞を解析する．

2　Android アーキテクチャ

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．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 カーネルの上には Android ランタイムと各種ライブラリが存在する．Android ランタイムは Android アプリケーションの実行環境であり，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．またライブラリには機能別に汎用性の高いプログラム群がまとめられており，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される．ライブラリの上にはアプリケーションフレームワークがあり，その上にアプリケーションが乗っている．

3　研究目的

これまでの Android に関する研究で，複数台の Android 端末を同一アクセスポイント (AP) に接続し通信させた際に，本来なら安定して通信帯域を確保できる状況にもかかわらず，不安定な通信を行う場合があることが確認されている．このような場合に，輻輳ウィンドウサイズの上限値を切替えられる独自の TCP を用いることで全端末が安定した通信を行えるための方式が研究されている．

ユーザは Android 端末上のアプリケーションを利用するため，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，より良く通信帯域を利用して安定した通信を行えると考えられる．そこで本研究では各アプリケーションの通信傾向の違いに着目し，Android 端末上でアプリケーションを実行した際のカーネル内部の振舞を調べて，通信特性に応じた制御を行うことで，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

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，輻輳ウィンドウサイズ，スロースタート閾値などの 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た．

4.2　通信性能の可視化ツール

また独自に開発した通信性能の可視化ツールを利用し，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した．

5　実験環境と実験対象アプリケーション

表 1 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．また本研究で使用するアプリケーションとしては，通信傾向の違いに着目し，リアルタイム通信アプリケーションには Ustream を用い，ファイル転送アプリケーションには ES file explorer と Picasa Tool を用いた．本研究ではカーネル内のトランスポート層の TCP 部分を対象としており，輻輳ウィンドウを測定している．輻輳ウィンドウは送信側が通信するデータ量を制限するパラメータであることから，使用するアプリケーションにはデータのアップロード機能を利用して実験を行った．

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

6　実験概要

6.1　Android 端末 1 台の通信

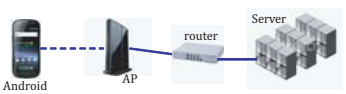

図 1: 1 台の Android 端末通信時の実験環境

図 1 に，各アプリケーション実行時の Android 端末 1 台のみを通信させた場合の実験環境を示す．Ustream を用いて 1 分間のストリーム配信を行ったところ，輻輳ウィンドウは 40 程度まで上昇した．ES file explorer を用いて自分で設定した接続先のローカルサーバへ 67MB のファイル転送を行ったところ，輻輳ウィンドウは 60 程度まで上昇した．Picasa Tool を用いてウェブサーバへ 10 枚の 12MB 程度の画像ファイルを転送したところ，輻輳ウィンドウは 25 程度まで上昇した．

6.2 Android 端末 4 台の通信

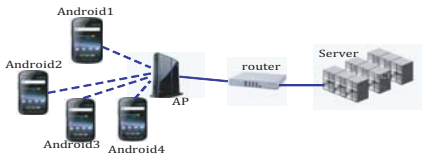

図 2: 4 台の Android 端末通信時の実験環境

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 に接続し通信させて実験を行った．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を行い，Android2, 3, 4 では ES file explorer を用いてファイル転送を行った．この結果ファイル転送を行う全端末の輻輳ウィンドウサイズは常に 60 程度まで上昇し安定した通信を行えるが，リアルタイム通信端末では，輻輳ウィンドウサイズが 1 台通信の時と同じ約 40 まで上昇できる場合と，その半分の約 20 までしか上昇できない場合に分かれることがわかった．

6.3 ファイル転送端末に独自の TCP を適用した場合

輻輳ウィンドウが上がりきらない状況が確認されたことから，リアルタイム通信端末が優先的に通信を行うために，図 2 の実験環境において，ファイル転送端末の輻輳ウィンドウの上限値を抑えて実験を行った．

図 3 に Android2, 3, 4 の上限値を 30 に設定した時の Android1 の測定結果を示す．安定して輻輳ウィンドウサイズを保てる時と同程度まで上昇していることが確認され，リアルタイム通信端末の輻輳ウィンドウサイズが低下する際には，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抑えることは有用であると確認できた．しかし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たことで，4 台全ての Android 端末が通常の TCP のときよりも 20(s) 程度ファイル転送に時間が掛かることが確認された．

図 3: Android1 の輻輳ウィンドウサイズ

6.4 リアルタイム通信端末に独自の TCP を適用した場合

そこでファイル転送時間を短縮するために，図 2 の実験環境において，ファイル転送端末は通常の TCP のままで，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た．Android1 の上限値を 20 にした場合，ファイル転送にかかる時間は，4 台全ての Android 端末が通常の TCP で輻輳ウィンドウが 40 まで上昇した場合よりも 8(s) 程度速くなることが確認された．また動画配信してから PC 上に流れるまでの時間や配信動画の見え方に違いがないことが確認されたことから，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，動画の品質に問題なくファイル転送時間を短縮できるといえる．

6.5 通信状況のわからない他端末との協調した通信

図 4: Android 端末と Wireless PC による実験環境

次に通信状況の分からない端末と Android を同時通信させて実験を行った．実験環境を図 4 に示す．既存研究より Nexus S の TCP より HT-03A の TCP のほうがパケット送出の積極性が弱い TCP であることが確認されていることから，通常の TCP の Nexus S より HT-03A のカーネルを移植した Nexus S を同時通信させた場合のほうが，2 台の PC と協調して通信を行えることが考えられる．そこで Android 端末に通常の TCP の Nexus S，HT-03A のカーネルを移植した Nexus S，HT-03A のカーネルを移植し輻輳ウィンドウの上限値を 20 に抑えた Nexus S を用いて，それぞれ，PC2 台と通信させた場合と PC2 台のみで通信を行う場合の性能を比較した．このとき Android 端末では Ustream を実行し，スループットの測定には iperf-2.0.4 を使用した．図 5 に測定結果を示す．どちらの PC も HT-03A のカーネルを移植した Nexus S を用いることでスループットが向上し，さらに上限値を抑えることで PC2 台通信の場合と同程度までスループットが向上することが確認された．

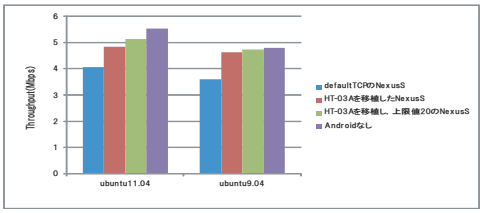

図 5: スループット比較

7 まとめと今後の課題

アプリケーションを実行する Android 端末を通信させた際の輻輳ウィンドウの測定を行った．異なる通信傾向のアプリケーションを同時通信させた場合に，リアルタイム通信端末に独自の TCP を用いることでファイル転送にかかる時間を縮められることが確認できた．また PC のような通信状況を確認できない端末とも協調して通信を行えることが確認できた．

しかし，一般にファイル転送とリアルタイム通信を比較すると，ファイル転送に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のほうが望まれると考えられる．そこで今後は Ustream の配信動画の品質に影響を及ぼす環境を調べ，通信性能を向上させていきたい．

参考文献

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

[2] 熊谷菜津美，平井弘実，三木香央理，山口実靖，小口正人: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，DEIM2012，2012 年 3 月発表予定