

アプリケーションデータを利用した Android スマートフォン向け アクティブ通信品質計測システム

立花 篤男^{†1} Anup Kumar Paul^{†1} 長谷川 輝之^{†1}

概要: モバイルキャリアが快適なモバイル通信サービスを提供するためには、通信ネットワークの状態/品質を詳細に把握することが重要である。これまで、通信ネットワークの状態/品質を把握する手法として、通信パスの両端のホスト間で試験パケットを送受信し、そのパケットの挙動を解析するアクティブ計測技術が研究開発されているが、アクティブ計測技術の多くは、ユーザが利用するアプリケーションのデータトラヒックとは別に試験パケットを送受信するため、頻繁にアクティブ計測を繰り返すと、ネットワークに流入するトラヒック負荷が増大してしまう課題がある。これに対し、筆者らは、Android スマートフォン向けに、アプリケーションのデータトラヒックを試験パケットに利用することにより、トラヒック量の増加を抑制するシステムを提案する。本稿では、提案システムの概要について述べるとともに、試験実装を用いた評価結果について述べる。

A Proxy-based Active Measurement Scheme for Android Smartphones

ATSUO TACHIBANA^{†1} ANUP KUMAR PAUL^{†1} TERUYUKI HASEGAWA^{†1}

Abstract- To widely deploy the performance measurement scheme on wireless access networks, in this paper, implementation design of an active measurement scheme with off-the-shelf Android smartphones and the data transformation servers on the network paths is presented. The presented scheme efficiently estimates the available bandwidth with a significantly fewer overheads of native active measurement based on a proxy technique that conveys application data by piggybacking inside the probing packets. At the same time, since the scheme is accomplished without relying on super-user (root) privileges on off-the-shelf Android smartphones, without any changes to the existing client applications and legacy servers, it has considerable potential for general acceptance. Through the preliminary experiments with the prototype implementation on the Internet, we confirmed the feasibility and the effectiveness of the proposed scheme.

1. はじめに

近年、スマートフォンによるモバイル通信はビジネスシーンや日常生活に深く浸透し、現在の社会基盤を支える重要な役割を担うとともに、ユーザの通信品質に対する要求水準も高くなっている。モバイルキャリアが高品質な通信サービスを安定してユーザに提供するためには、通信ネットワークの状態/品質を詳細に把握し、基地局の増設やトラヒック制御などの適切な運用対処を実施することが重要である。特に、通信帯域に関する品質指標は、エンドツーエンドのアプリケーション品質に直接影響を与えるため重要である。

エンドツーエンドの通信帯域特性を把握する手法に関し、これまで多数の研究が行われている。これらは、ネットワーク上を流れるユーザ実トラヒックを直接計測するパッシブ計測と、ネットワークパスの両端の端末間で試験パケットを送受信し、そのパケットの挙動から対象ネットワークパスの特性を計測するアクティブ計測の2つに大別される。パッシブ計測手法では、試験パケットをネットワークに挿入する必要がないため、ネットワークに与えるトラヒック

負荷が小さい特長がある一方、計測対象とする通信フローのサイズが不十分な場合や、計測対象の通信フローがレート制御などのトラヒック制御の影響を受けている場合は、通信帯域を正確に計測できない課題がある。

一方、通信帯域をアクティブ計測する手法も多数研究開発されている。最も初歩的な通信帯域のアクティブ計測手法としては、iperf[5]や計測専用ソフト(e.g., [4])を利用して、TCP スループットを直接計測する手法がある。しかし、この手法は TCP の輻輳ウィンドウを可用帯域付近まで増加させる必要があるため、大量の試験データと長い計測時間が必要になる課題がある。一方、ネットワークパスの両端の端末間で比較的少量の試験パケットを送受信し、遅延の増大などのパケットの挙動を解析することにより、可用帯域を間接的に推定する手法・ツールも研究されている(e.g., [6-11])。ここで、可用帯域とは、ネットワークパスが提供可能な最大通信帯域のことである。

近年、アクティブ計測技術の進展とともに、大規模な計測プラットフォーム/システムに関する研究開発も多数報告されている。例えば、PlanetLab[12]や M-LAB[13]は研究

^{†1} (株)KDDI 研究所
KDDI R&D Laboratories Inc.

者向けに構築されている代表的な計測プラットフォームであり、これらを利用して、様々な計測システムに関する研究が行われている。特に、近年では、ユーザ端末を直接利用し、ユーザ視点に立ったネットワーク品質/特性の計測を実現するアプローチが研究開発されている。例えば、Fantom[14]は、Firefox ブラウザの拡張として、ユーザ端末上にプログラマブルな計測プラットフォームを展開する。また、多くのユーザはスマートフォン用の専用アプリケーションを用いて、自らのコネクションの通信速度を計測する Speedtest.net[4]もユーザ視点に立ったアクティブ計測システムの一例と考えられる。

しかしながら、一方で、このようなアクティブ計測手法大規模なモバイル通信の監視に適用するためには、計測トラヒックによるネットワーク負荷の増大や、計測サンプル数の不足などの課題がある。特に、ユーザがアクティブ計測に参加する計測システムでは、試験パケットの送受信によって参加ユーザの通信料金が増加してしまい、ユーザがアクティブ計測に頻繁に参加できなくなる。この課題に対し、本稿では、Android スマートフォンユーザを対象として、ユーザ実トラヒックを試験パケットに埋め込んで利用することにより、アクティブ計測による通信トラヒック量の増大を抑制する手法を提案する。

2. 関連研究

1 章で述べた通り、通信帯域の計測手法に関し、これまで多数の研究が報告されている。パッシブ計測手法では、ネットワーク内部の装置でキャプチャしたユーザ実トラヒックを解析し、ボトルネックリンクや当該リンクにおける通信帯域を特定する([1-2])。パッシブ計測手法は試験パケットをネットワークに挿入する必要がないため、ネットワークに与える負荷が低い特長がある一方、計測対象とする通信フローのデータ転送量が不十分な場合は、TCP 輻輳ウィンドウサイズが利用可能な上限まで増加せず、正確な計測が困難となる課題がある。多数の TCP フローの中から通信帯域の把握に適した TCP フローを抽出する手法も提案されているが[19]、今日のモバイル通信における大半の TCP フローはデータ転送量が回線速度に比べて少量なため[20]、解析対象外と分類されてしまう。さらに、データ転送レートがアプリケーションプログラムにより制御されている通信フローに対しても、正確な計測が困難となる課題がある。例えば、人気アプリケーションである Youtube では、サイズの大きい動画データの転送を行うが、アプリケーション/アプリケーションサーバによって転送レートが制御されているため、パッシブ計測をそのまま適用することは困難である。これらの課題に対し、近年、キャプチャしたユーザ実トラヒックの解析において、近接する TCP パケットを疑似的なパケットペア(連続するパケットの組)とみなし、これらのパケットの挙動を解析することにより、可用

帯域を推定する手法が研究されているが[18]、モバイル環境における有効性については十分に検証されていない。

一方、エンドツーエンドの可用帯域をアクティブ計測する手法もこれまで多数研究されている。少量の試験パケットの送受により可用帯域を推定する代表的なツールとして、pathChirp[6]、Assolo[7]、WBest[8]、BART[9]、PTR[10]などがある。例えば、pathChirpは、パスの両端の端末間で、連続した複数のUDP試験パケットで構成されるパケットトレインをネットワーク内に送受信し、送受信ノード間の可用帯域幅を推定する。このとき、パケットトレイン中の試験パケットの転送間隔を指数関数的に減少させ、送信レートを徐々に上げることで、故意に輻輳を発生させた際に各試験パケットが経験するパケット遅延の増加傾向から可用帯域幅を推定する。この手法は、Self-Induced Congestion と呼ばれる。また、筆者らは、過去に、pathChirpと同様に、Self-Induced Congestionの推定原理に基づく可用帯域推定ツールとしてNEXTを提案した[21]。NEXTでは、パケットトレイン中の試験パケットの間隔を複数の指数関数に基づき調節することでpathChirpよりも効率的な試験パケットの配置を実現することで高精度を実現し、さらに、可用帯域の推定範囲を高速に調節することで、短い推定時間を実現する。他のツール例として、特定のTCPパケットをサーバに対して送信した際に返送されるICMPメッセージ応答を利用し、可用帯域を推定するAbget[22]やAbode[23]がある。これらのツールはクライアント側端末のみに計測プログラムをインストールすれば良く、高い実用性を持つが、ICMPが正常に通過するネットワークに適用範囲が限定される課題や、推定に必要な試験パケットの量がNEXTよりも多い課題がある。

アクティブ計測技術の研究が進展するとともに、様々な計測プラットフォームの研究も行われてきた(PlanetLab[12]、M-LAB[13]、Periscope[24]、pktd[25]、Scriptroute[26]、MAD[27]、precision probing[27])。特に、近年では、ユーザ端末などを直接的に利用する様々な計測システムも提案されている。例えば、1章で述べたように、Fathom[14]は、ユーザ端末を利用し、Firefoxブラウザの拡張として、プログラマブルな計測プラットフォームを実現する。また、文献[15]は、スループット計測アプリケーションであるSpeedtest.net[4]が収集した大量のスループットデータを解析している。これらのアプローチは有望であるものの、アクティブ計測によるトラヒック量の増加の抑制は行っておらず、大規模なモバイル通信の監視への適用は困難である。Measurement Manager Protocol (MGRP) [17] は、ネットワークパスの両端の端末上でカーネルレベルのサービスとして動作し、ユーザ実トラヒックを試験パケットに埋め込んで利用することにより、アクティブ計測によるトラヒック量の増加を抑制する。MGRPは提案手法と最も関連深い手法であるが、カーネルレベルの実装は移植性が低い

ため、既存のモバイル通信システムへの導入コストが高い課題がある。既存の計測システムと提案システムとの大きな差異は、提案システムはAndroidスマートフォン向けに、計測負荷を抑制しつつ可用帯域推定を実現する機能を持つ点である。

3. 提案システム

3.1 提案システムの概要

NEXT を含む多くの通信帯域推定ツールは、計測用プログラムをネットワークパスの両端の端末にインストールする必要がある。しかしながら、モバイルキャリアがサードパーティの管理する既存のアプリケーションサーバに計測プログラムをインストールすることは導入コストの観点から非現実的である。そこで、提案システムでは、Androidスマートフォン向けに計測プログラムを配布するとともに、通信経路上にデータ中継サーバを導入する。データ中継サーバは、既存アプリケーションサーバとスマートフォン間のデータ中継を実現するとともに、プロキシサーバとAndroidスマートフォンとの間で、ユーザ実データを利用した可用帯域のアクティブ計測を実行する。ここで、提案システムでは、既存のアプリケーションやアプリケーションサーバに対して、計測プログラムやデータ中継サーバの存在を隠蔽するため、既存のアプリケーションやサーバに対して一切の変更を加える必要はない。

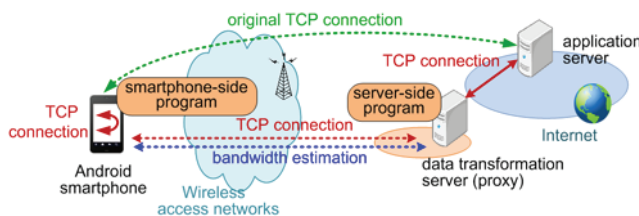


図1 提案システムの構成

3.2 NEXTの概要

NEXTはpathChirpと同様に、パケットトレインを用いて故意に輻輳を発生させるself-induced congestionに基づいて設計されている。送信端末は、連続した複数のUDP試験パケット(固定サイズ)で構成されるパケットトレインを受信端末に対して送信し、受信端末は、各試験パケットの経験する片道遅延変動の増加傾向を解析する。具体的には、片道遅延変動の増加が開始する地点を検出し、当該地点に相当する試験パケットペアの転送間隔、及び試験パケットサイズから試験パケット転送レートを算出し、可用帯域として推定する。NEXTでは、2つの異なるスプレッドファクター(γ_1 , γ_2)を用いて、パケットトレイン中の試験パケットの間隔を効率的に配置することで、高精度な推定を実現する。なお、NEXTでは、pathChirpと同様、一時的なクラスタヒックの変動等によって、片道遅延変動の増加開始

点の検出を誤る可能性があるため、複数のパケットトレインを注入するとともに、推定対象とする可用帯域の上限/下限値の調節を行う。ただし、NEXTは、1パケットトレインごとの推定結果を即座に調節に反映させることで、pathChirpよりも高速な推定を実現している。そこで、本稿では、可用帯域推定手法として、NEXTを採用する。ただし、提案システムはNEXT以外のアクティブ計測ツール(e.g., [6-10])の適用も可能である。

3.3 処理フロー

本稿では、提案システムの適用シナリオとして、今日のモバイル通信で主要なアプリケーションであるWebブラウザへの適用を想定する。本節では、ユーザがAndroidスマートフォンを用いてWebサイトを閲覧する状況を想定した際の、処理フローについて説明する。本シナリオでは、新規のTCPコネクションはスマートフォン側から発生し、HTTPレスポンス等のアプリケーションデータは既存アプリケーションサーバからスマートフォンの方向に転送される。すなわち、ダウンリンク方向の可用帯域が計測される。

(1) TCPフローのリダイレクト/フォワーディング

図2に提案システムの構成を示す。まず、TCPプロキシプログラム(以下、単に“プロキシ”と呼ぶ)は、Androidスマートフォンにおいて、既存アプリケーションのTCPフローを受信し、データ中継サーバに対して、TCPフローをフォワーディングする。また、データ中継サーバからアプリケーションデータを受信した場合は、アプリケーションプログラムに対してデータ転送を行う。スマートフォン上のTCPフローのリダイレクトは、Android(4.0以降)においてVPNサービスを提供するVPN Service Class[30]を利用することにより、root権限の取得を行わずに実現できる。

VpnService Classが有効化され、Androidスマートフォン上で、新たなTCPパケットが送信されるとプロキシは送信元のアプリケーション、および、データ中継サーバ側のプロキシ(サーバ側プロキシと呼ぶ)とTCPコネクションを確立する。サーバ側プロキシは、最初のTCPパケットを受信すると、既存アプリケーションサーバと新規のTCPコネクションを確立する。新たなTCPコネクションは5タプル(送信元IPアドレス/ポート番号、宛先IPアドレス/ポート番号、プロトコル)により識別され、Androidスマートフォン上のプロキシ(スマートフォン側プロキシと呼ぶ)はユニークなフローID(FID)を割り当て、TCPコネクションとFIDとの対応情報を当該TCPフローがアクティブな期間、マッピングテーブルとして保存しておく。スマートフォン側のTCPコネクションが確立された後、スマートフォン側プロキシはFID対応情報をサーバ側プロキシに対して、最初に送信するTCPパケットに埋め込んで通知する。サーバ側プロキシは受信したTCPパケットを評価し、FID対応情報を読み出し、含まれる接続先の既存アプリケーションサーバに対して、新規のTCPコネクションを確立する。これらの

3つのTCPコネクションが確立された後、データ転送が可能になる。

他のTCPコネクションが追加で確立される場合は、スマートフォン側プロキシは、アプリケーションとの間でTCPコネクションを確立し、サーバ側プロキシも同様に、既存サーバとの間でTCPコネクションを追加で確立する。一方、スマートフォン側プロキシとサーバ側プロキシの間には、新規にTCPコネクションは確立せず、既に確立されているTCPコネクションを介してデータ転送を行う。すなわち、スマートフォン側プロキシとサーバ側プロキシとの間のTCPデータ転送は、それぞれ異なるFIDに対応づけられる複数のTCPコネクションが1本のTCPコネクションに集約される形で実現される。

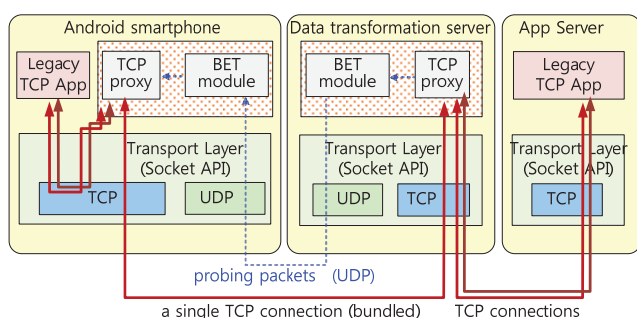


図2 実装構成

(2) アプリケーションデータの試験パケットへの埋め込み

図3に、提案スキームの構成要素を示し、さらに、Androidスマートフォン上のアプリケーションからインターネット上のサーバ方向に転送されるデータのフローを示す。ここで、Androidアプリケーション(ブラウザ)は3本のTCPコネクションを介してアプリケーションデータ(HTMLコンテンツ)をダウンロードすると想定する。

まず、データ中継サーバはアプリケーションデータを既存アプリケーションサーバから受信し、サーバ側プロキシは受信パケットからペイロード部を抽出するとともに、マッピングテーブルを参照し、データを転送すべきスマートフォン(のIPアドレス)を特定する。サーバ側プロキシは宛先IPアドレスごとに管理される内部の出力キューに挿入する。ここで、各データペイロードには、FIDを含む専用ヘッダが付与される。(図3において、2つのキューが存在するのは、ことなる時間スロットにおける異なる2状態を表現している。)

最初のペイロードがキューに挿入されると、該当バッファに関するバッファタイマーが起動する。以下に示す条件の一方が満足された場合、サーバ側プロキシはペイロードをキューから抽出する。

(Cond.1) バッファタイマーが閾値 T_{max} を超過する。

(Cond.2) バッファされたペイロードの合計サイズが閾値

B_{th} を超過した場合。ここで、 B_{th} は同一パケットトレインに属する試験パケットに埋め込める最大データ量に設定される。ここで、アプリケーションデータがバースト的にデータ中継サーバに到着する場合など、バッファされるデータ量は B_{th} を超過する場合もある。

もし、Cond.1 が満足された場合、サーバ側プロキシはキューから抽出したデータをBET(Bandwidth Estimation Tool)に送信する。BETは受信したデータを試験パケットのペイロード部に埋め込むとともに、送信タイムスタンプやパケットトレイン内の試験パケットのシーケンス番号など、可用帯域推定に必要な情報を含む専用ヘッダを付与する。ここで、試験パケットを埋めるためのデータが不足する場合は、BETモジュールは空データで試験パケットを埋める(図3におけるケース1)。

一方、もしCond.2が満足されると、サーバ側プロキシはキューからバッファされているうちの B_{th} バイト分のデータを取り出し、BETモジュールに転送し、試験パケットに利用する。一方、余りのデータは通常のTCPソケットを介して、スマートフォンとの間に確立されているTCPコネクションを介してデータ転送する(図3中のケース2)。これは、NEXTが少量の試験パケットの送受信により可用帯域を推定するため、通常のTCPデータ通信よりも低速なデータ転送速度となり、大量のデータがデータ中継サーバのバッファに蓄積され、アプリケーションの品質を提言させてしまう可能性があるためである。

(3) ロスパケットの復元

NEXTは試験パケットとしてUDPパケットを利用するため、アプリケーションデータを高信頼に転送するためには、パケットロスに対する対策が必要となる。また、故意に輻輳を発生させて可用帯域を見積もるSelf-Induced Congestionをベースに設計されているため、他の可用帯域推定技術と同様に、パケットロスが発生しやすくなる可能性もある。そこで、提案システムでは、以下に示す2つの機能を実装した。

- FEC(Forward Erasure Correction)を用いて、転送されるデータを冗長化する。OpenFEC[32]を用いて事前評価を行った結果、Reed Solomon Stable コーデック(2⁸)を用いて4パケットを付与することにより、20パケットで構成されるパケットトレインにおいて4個のパケットロスまでデータを復元できることを確認した。また、エンコードとデコードに要した時間は、それぞれ0.8s, 0.01s以下であった。このため、提案システムではこのアルゴリズムと設定を採用した。
- 受信端末で復元不可能なアプリケーションデータを検知した場合、受信端末は送信端末に対して、ロスデータの再送を依頼する(図3中のケース3)。データの再送処理が可用帯域の推定精度に影響することを回

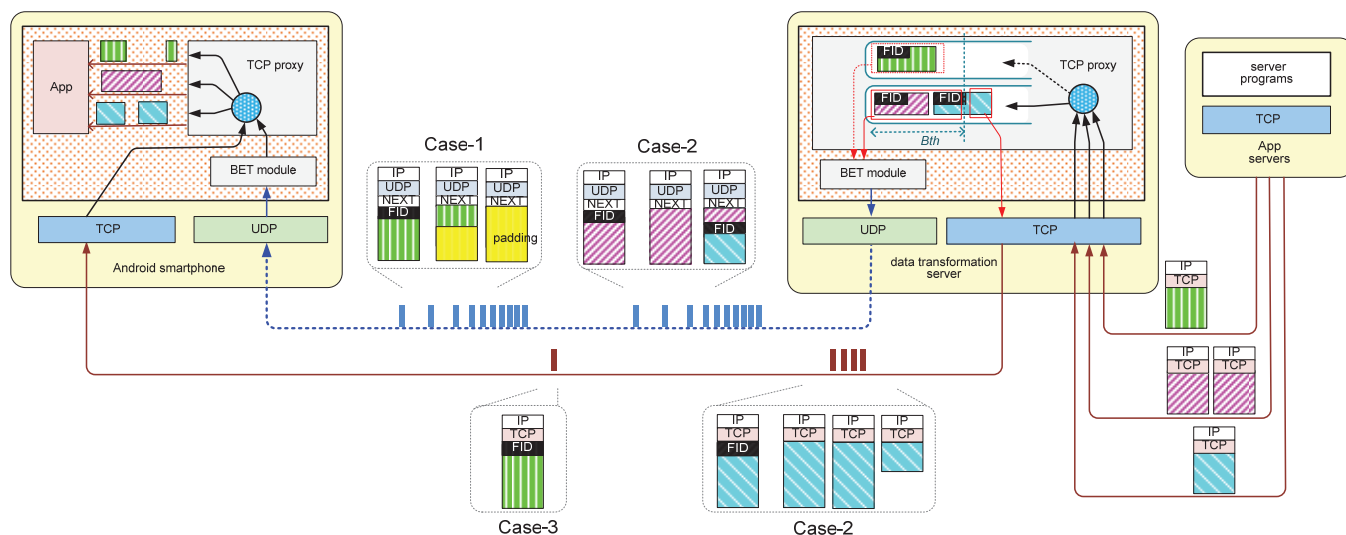


図3 データ処理フロー

避するため、データ再送は試験パケットの送受信と排他的に処理する。

(4) TCP コネクションの切断

TCP コネクションは以下の手順に従って切断される。

- 外部アプリケーションサーバとデータ中継サーバとの間の TCP コネクションが切断された場合、サーバ側プロキシはスマートフォン側プロキシに切断を通知するための専用メッセージを送信する。TCP コネクション切断前のデータ転送を確実に処理するため、当該メッセージは、データ中継サーバの出力キューにおいてアプリケーションデータの後の最後尾に追加される。言い換えると、この接続メッセージ自体も、試験パケットに埋め込まれて転送される。スマートフォン側プロキシが切断通知メッセージを受信すると、Android アプリケーションとの間に確立している TCP コネクションの中から該当する TCP コネクションを検索し、切断する。
- スマートフォン側プロキシと Android アプリケーションとの間の TCP コネクションが切断された場合、スマートフォン側プロキシは切断通知メッセージをデータ中継サーバに対して送信する。サーバ側プロキシは外部サーバとの間に確立している該当 TCP コネクションを切断する。このとき、該当 TCP コネクションに関するアプリケーションデータがデータ中継サーバのバッファに格納されている場合、それらのデータは即座に破棄し、不要なデータ転送を回避する。
- 上記の両方の場合において、切断された TCP コネクションに紐づけられた FID は解放され、新たに別の TCP コネクションが確立された際に再利用される。

4. 評価

4.1 実験構成

提案システムの有効性を検証するため、プロトタイプ実装による評価実験を実施した。図4に実験ネットワークの構成を示す。データ中継サーバは、光ファイバ回線(200Mbps)を介して1つのISPネットワークに接続されている。データ中継サーバのスペックは以下の通りである。CPU: Intel Xeon E3-1270@3.50GHz, RAM: 16GB, OS: Ubuntu 14.04.1(Linux Kernel 3.13)。また、スマートフォンには、SONY Xperia SOL22(CPU:Quad Core APQ8064@1.5GHz, RAM: 2GB, OS: Android 4.2.2)を使用し、LTEを介してインターネットに接続した。また、本実験環境において、スマートフォンとデータ中継サーバ間の往復遅延時間(RTT)は約120ms、データ中継サーバとアプリケーションサーバ(Webサーバ)間の往復遅延時間は約15-20msであった。

表1は、NEXTの設定パラメータを示す。初期の可用帯域幅の下限/上限として、それぞれ、1Mbpsと10Mbpsを設定した。また、パケットサイズは1200バイト(固定)に設定した。可用帯域のスプレッドファクターとして、 γ_1 と γ_2 は、表1に示す4通りの値を用いた(Case 1~4)。ここで、 γ_1 と γ_2 の値が小さくなるほど、可用帯域の推定粒度は細くなるが、パケットトレインを構成する試験パケット数が増大するため、計測負荷が増大することになる。表1に、1つのパケットトレインで送信可能な最大データサイズ(=B_{th})も併せて示す。

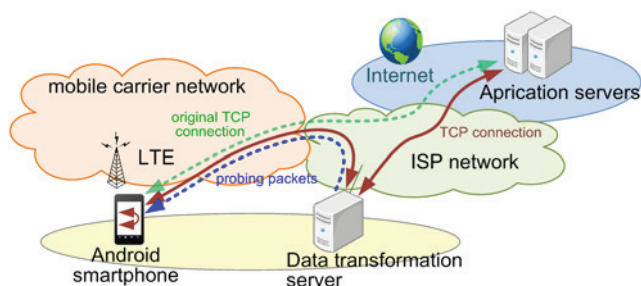


図 4 実験ネットワーク

表 1 NEXT のパラメータ

	Case 1	Case 2	Case 3	Case 4
spread factor: γ_1	1.2	1.15	1.1	1.06
spread factor: γ_2	1.1	1.05	1.03	1.02
# of probes/train	15	21	31	50
approx. total bytes/train (=Bth)	18K	25K	38K	61K

4.2 Web サイト閲覧試験

(1) 評価メトリック

- 提案システムのアプリケーション品質への影響を評価するため、Chrome ブラウザにより Web サイトを閲覧する場合のコンテンツダウンロード時間を評価した。
- アプリケーションデータを試験パケットとして利用するための効率性について評価するため、パケットトレインの送受信回数を集計した。
- 試験パケットへのアプリケーションデータの埋め込み処理が計測精度に与える影響を評価するため、提案システムと元の NEXT との計測結果を比較した。また、推定結果の妥当性を、異なる手法で計測した通信帯域と比較評価した。商用 LTE ネットワークにおける正確な可用帯域を取得することは困難なため、FTP を用いたスループットを代替で利用した。
- アクティブ計測によるオーバーヘッドを評価するため、Web サイトの閲覧時に送受信するデータ量を集計した。

(2) 実験手順

以下の手順に従って実験データを取得した。

(Step 1) Chrome ブラウザのキャッシュをクリアする。

(Step 2) Chrome ブラウザと Chrome Developer tools [33]を用いて、日本の有名サイトを閲覧しながら、Web コンテンツのダウンロード時間を計測する。また、tcpdump[34]を用いて、スマートフォンが受信するデータ量を計測する。

(Step 3) Step 2 に続いて、別の 9 個の Web サイトを閲覧する。ここで、各 Web ページにおけるコンテンツデータ量は 200~1.6MB であり、1つの Web ページあたり 50~150 の HTTP リクエストが送信される。

(Step 4) 大雑把な通信帯域を計測するため、データ中継サーバから 10MB のファイルをダウンロードし、平均スル

ープット(=ファイルサイズ/ダウンロード時間)を計算する。

(Step 5) Step 1~4 の手順を 10 回繰り返し、Web ページの平均ダウンロード時間を計算した。

(Step 6) Step 1~5 の手順を表 1 に示すパラメータを用いて繰り返す。

(Step 7) 提案手法の効果を評価するため、提案システムを用いずに Step1~5 の手順を実行した。すなわち、ブラウザは既存アプリケーションサーバから直接 TCP コネクションを介してアプリケーションデータを受信する。

(Step 8) さらに、試験パケットの送受信を無効にした状態で Step 1~5 の手順を繰り返した。この場合、アプリケーションデータはデータ中継サーバを経由し、TCP コネクションを介して転送される。

(3) 結果

- 図 5 に Web ページの平均ダウンロード時間を示す。

Case 1~4 において、Bth の値を変化させた場合であっても、平均ダウンロード時間に大きな差異はなかった。この結果より、ダウンロード時間は NEXT のパラメータに大きく依存しないことがわかる。Case5 と Case6 との比較において、平均ダウンロード時間がそれぞれ 6 秒と 8.5 秒となっており、大きく異なっている。これは、データ転送サーバを経由するために増加したパケット遅延が大きく影響したと考えられる。データ転送サーバの最適配置問題は今後の課題であるが、一般的に、データ転送サーバをモバイルキャリアのバックボーンに配置することにより、この影響は軽減できると考えられる。Case6 と Case1~4 を比較すると、ダウンロード時間の差異は 1.7 秒以下であった。このため、ユーザ実トラフィックデータの試験パケットへの埋め込みによるアプリケーション品質への影響は大きくないと考えられる。

- 図 6 に 10 個の Web ページを閲覧した場合に送受信された平均パケットトレイン数を示す。図 6 において、1つのパケットトレインに含まれる試験パケット数が小さくなるほど、パケットトレインの数は増加している。また、バッファタイムアウト値が大きくなるにつれ、パケットトレイン数が増加している。これは、より大きなバッファタイムアウト値の設定により、データ中継サーバにおいてバッファされるアプリケーションデータが Bth を超過する機会が増えたためと考えられる。

- 図 7 に、提案システムとオリジナルの NEXT により、Case1 のパラメータを用いて推定された可用帯域、および、FTP ファイル転送により計測された平均スループットの分布を示す。ここでバッファタイムアウトは 50ms を採用した。図 7 に示される通り、提案システムとオリジナルの NEXT との間に大きな差異はなく、ユーザ実データの埋め込み処理が推定精度に影響し

ていないことが確認できた。提案システムと NEXT の推定結果を FTP スループットと比較すると、推定結果のばらつきが大きい、より多数の推定結果を統計的に処理するなどの方法により、改善できる可能性がある。

- 表 2 に、Case1 と Case6 の場合に、スマートフォンが、10 個の Web ページを閲覧する際に受信したデータ量を示す。また、試験パケットに埋め込まれたアプリケーションデータのサイズも示す。表 2 より、アプリケーションデータを計測パケットに利用することにより、アクティブ計測の負荷の増加は 4.5%に削減された(0.32Mbyte/7.15Mbyte)。

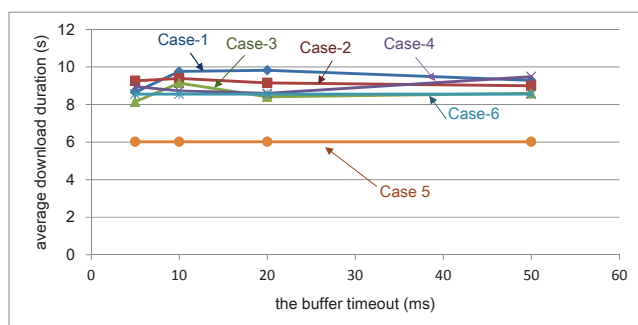


図 5 平均ダウンロード時間

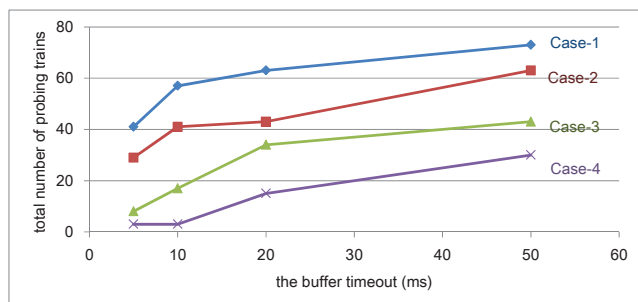


図 6 パケットトレイン数

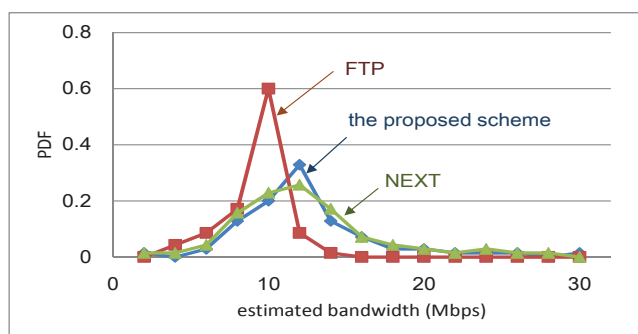


図 7 可用帯域の分布

表 2 データ受信量

	Case 1	Case 6
received data (Mbyte)	7.47 Mbyte	7.15 Mbyte
piggybacked data inside probing packets (Mbyte)	1.02 Mbyte	-

5. おわりに

本稿では、Android スマートフォン向けに、アプリケーションデータを試験パケットに利用することにより、アクティブ計測により発生するトラフィックの増加を抑制するシステムを提案した。プロトタイプ実装を用いて評価を行い、有用性を確認した。今後、より大規模なインターネット実験を実施する予定である。

参考文献

- 1) S. Katti, D. Katabi, C. Blake, E. Kohler, and J. Strauss, "MultiQ: automated detection of multiple bottleneck capacities along a path," Proc. of ACM IMC, 2004, pp. 245–250, New York, NY, USA, 2004.
- 2) F. Ricciato, F. Vacirca, and M. Karner, "Bottleneck detection in UMTS via TCP passive monitoring: a real case," Proc. of ACM CoNEXT '05, pp. 211–219, New York, NY, USA, 2005.
- 3) D. Bonfiglio, M. Mellia, M. Meo, N. Ritacca, D. Rossi, "Tracking Down Skype Traffic," Proc. of IEEE INFOCOM, 2008.
- 4) Speedtest.net. <http://www.speedtest.net>, 2011.
- 5) A. Tirumala, F. Qin, J. Dugan, J. Ferguson and K. Gibbs, "Iperf. Testing the Limits of Your Network," <http://dast.nlanr.net/Projects/Iperf>.
- 6) V.J. Ribeiro, R.H. Riedi, R.G. Baraniuk, J. Navratil, and L. Cottrell, "pathChirp: Efficient Available Bandwidth Estimation for Network," Proc. Passive and Active Measurement Workshop, 2003.
- 7) E. Goldoni, G. Rossi, and A. Torelli, "Assolo, A New Method for Available Bandwidth Estimation," Proc. of the 2009 Fourth International Conference on Internet Monitoring and Protection. Washington, DC, USA: IEEE Computer Society, 2009, pp. 130–136.
- 8) M. Li, M. Claypool, and R. Kinicki, "WBest: A Bandwidth Estimation Tool for IEEE 802.11 Wireless Networks," in Local Computer Networks, 2008. LCN 2008. 33rd IEEE Conference on, Oct. 2008, pp. 374–381.
- 9) S. Ekelin et al., "Real-time measurement of end-to-end available bandwidth using kalman filtering,," Proc. of IEEE NOMS 2006, pp. 73–84.
- 10) N. Hu and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques,," IEEE Journal on Selected Areas in Communications, vol. 21, no. 6, pp. 879–894.
- 11) A. Shriram and J. Kaur, "Empirical evaluation of techniques for measuring available bandwidth," Proc. of INFOCOM 2007, pp. 2162–2170.
- 12) B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "PlanetLab: An Overlay Testbed for Broad-coverage Services," ACM SIGCOMM Computer Communication Review, 33(3), July 2003.
- 13) Measurement Lab. <http://www.measurementlab.net/>.
- 14) P. Romi M. Dhawan, J. Samuel, R. Teixeira, C. Kreibich, M. Allman, N. Weaver, V. Paxson, "Fathom: a browser-based network measurement platform," Proc. of ACM Internet Measurement Conference, November 14–16, 2012.
- 15) J. Sommers and P. Barford, "Cell vs. WiFi: On the Performance of Metro Area Mobile Connections," Proc. of ACM Internet Measurement Conference, November, 2012.
- 16) P. Papageorgiou and M. Hicks. "Merging Network Measurement with Data Transport," Proc. of IEEE PAM 2005.
- 17) P. Papageorge, J. McCann, and M. Hicks, "Passive Aggressive Measurement with MGRP," Proc. of ACM SIGCOMM, 2009.
- 18) T. Pogel, J. Lubbe, L. Wolf, "Passive Client-based Bandwidth and Latency Measurements in Cellular Networks," Proc. IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2012.

- 19) A. Gerber, J. Pang, O. Spatscheck, S. Venkataraman, “Speed Testing without Speed Tests: Estimating Achievable Download Speed from Passive Measurements, Proc. of ACM IMC, pp. 424-430, 2010.
- 20) A.K. Paul, A. Tachibana, and T. Hasegawa, “Next: New enhanced available bandwidth measurement technique, algorithm and evaluation,” Proc of IEEE PIMRC, Washington DC, USA, 2014, pp. 443-447.
- 21) Y. Zhang, Å. Arvidsson, “Understanding the Characteristics of Cellular Data Traffic,” Proc. of CellNet’12 (SIGCOMM WKSHP), pp.13-18, 2012.
- 22) D. Antoniadis, M. Athanatos, A. Papadogiannakis, E. P. Markatos, and C. Dovrolis, “Available bandwidth measurement as simple as running wget,” Proc. of IEEE PAM, 2006.
- 23) X. Xing and S. Mishra, “Where is the tight link in a home wireless broadband environment?,” Proc. of IEEE/ACM MASCOTS, 2009.
- 24) K. Harfoush, A. Bestavros, and J. Byers, “PeriScope: An Active Probing API,” Proc. of IEEE PAM, 2002.
- 25) J. M. Gonzalez and V. Paxson, “pktd: A Packet Capture and Injection Daemon,” In IEEE PAM, 2003.
- 26) N. Spring, D. Wetherall, and T. Anderson, “Scriptroute: A facility for distributed Internet measurement,” In USITS, 2003.
- 27) J. Sommers and P. Barford, “An Active Measurement System for Shared Environments,” In ACM IMC, 2007.
- 28) A. Pásztor and D. Veitch, “A Precision Infrastructure for Active Probing,” Proc. of IEEE PAM, 2001.
- 29) X. Liu, K. Ravindran, and D. Loguinov, “Multi-hop Probing Asymptotics in Available Bandwidth Estimation: Stochastic Analysis,” Proc. of ACM IMC 2005.
- 30) VPNService Class;
<http://developer.android.com/reference/android/net/VpnService.html>
- 31) lwIP: <http://savannah.nongnu.org/projects/lwip/>
- 32) OpenFEC; <http://openfec.org/>
- 33) Chrome Developer Tools; <https://developer.chrome.com/devtools>
- 34) tcpdump; <http://www.kandroid.org/online-pdk/guide/tcpdump.html>