修士論文

スマートフォンが持つ複数の無線回線の帯域集約手法

Bandwidth aggregation with multiple wireless communication on smartphone

平成 25 年 2 月 6 日提出

指導教員 江崎 浩 教授

東京大学大学院 情報理工学系研究科
電子情報学専攻

学生証番号：48116423　氏名：小坂 良太

概要

スマートフォンなどのモバイル端末は複数の無線モジュールを有しているため複数の無線回線に接続が可能である．しかしながら，多くの端末において複数の無線回線を切り替えて接続しており同時に接続を行なってはいない．無線回線は電波環境によって帯域が小さくなる場合や帯域が変動する場合があるため，複数の無線回線に接続するだけではなく帯域を集約する手法が必要となる．複数の回線の帯域を集約するプロトコルは存在するが，それらを使用するには OS の改変が必要になるため既存のスマートフォンにて適用することは困難である．そこで本研究では OS に改変の必要のない手法を提案する．本手法ではトランスポート層に TCP を使用し，アプリケーション層にてマルチパス通信を実現するもので，TCP を使用しているモバイル端末であれば OS に改変をせずに使用が可能である．帯域を集約するために，転送するデータを各回線に割り当てなければならないが，帯域に応じて割り当てを行わなければ帯域を十分に活用することはできない．無線回線では帯域が変動するため事前に測定するだけでは不十分である．そこで，データを分割し，そのブロックごとに転送を行い，転送が完了した回線に次のブロックを割り当てるという手法を提案する．この手法では，事前に帯域を測定する必要がなく，また一方の回線の帯域が小さくブロックの転送に時間が掛かっている場合に他方の回線で容易に再送が行うことが可能である．そのため，通信品質の異なる回線を集約した場合であっても，集約を行なった回線のうち最も良い回線の通信品質が保証される．また，事前に帯域を測定する必要がないため，ファイルサイズが小さいなどの理由によりデータの転送時間が短い場合であってもマルチパス通信が可能である．

WEB アクセスを行うアプリケーションに対してプロトタイプを実装し評価した．その結果，再送処理が動作した場合に通信帯域が集約できることを示した．また，実際のスマートフォンにおいても動作の確認を行い，実際の無線回線を使用したマルチパス通信を行い，帯域の集約が実環境においても行えることを示した．

目次

図目次

表目次

第 1 章

序論

1.1 背景と目的

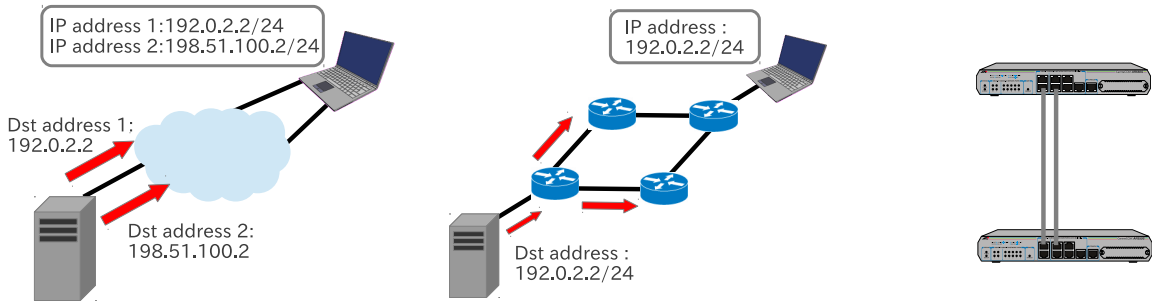
一部のフィーチャーフォンやスマートフォンは 3G 回線 (第 3 世代移動通信システムを用いた回線) や無線 LAN といった無線ネットワークに接続するための無線モジュールを複数有しており，いずれかの電波が届く場所であればインターネットに接続が可能である．また，モバイル端末自体に複数の無線モジュール端末を有していない場合であっても，モバイルルータやテザリングと呼ばれる機能を持つ携帯電話を介すことで異なる回線からインターネットに接続が可能である．このように様々なモバイル端末が様々な場所において様々な回線でインターネットに接続するようになってきた．

接続可能な無線網の種類が増えているが，モバイル端末の多くはそれら無線網のうち一つを選択し通信を行なっている．例えば，3G 回線と Wi-Fi(Wi-Fi に対応している無線ルータが提供するネットワークを本論文では Wi-Fi と表記する) に接続が可能なスマートフォンは Wi-Fi モジュールの電源が入っていて且つ接続可能な無線アクセスポイントが存在する場合に 3G 回線によるデータ通信を行わないという制御がなされている場合がある．この条件下でユーザが 3G 回線を用いてデータ通信を行いたい場合は Wi-Fi モジュールの電源を切らなければならない．この問題に対して，モバイル端末向け通信事業者 (本論文ではキャリアと呼ぶ) は Wi-Fi の強度が弱い場合や通信品質が悪い場合に自動で 3G 回線を用いて通信する仕組みを導入している [38]．しかしながら，3G 回線も Wi-Fi も電波が届きにくい環境ではどちらに繋いでも通信帯域が満足に得られないという問題が残っている．そこで本研究では単に接続を切り替えるのではなく，複数の無線網を用いて同時に通信する手法を提案する．この手法によりユーザの得られる通信品質の改善を目指す．

複数の通信網を用いて同時に通信するマルチパス通信は大きく分けると下記の 3 種類ある (図 1.1).

1. いずれかのノードが複数の IP アドレスを持ち，どちらの IP アドレスに対してパケットを送信するか決めるもの (図 1.1(a)).
2. クライアントとサーバ間の経路が複数ある場合に，送られてきたパケットをどちらの経路に流すか決めるもの (図 1.1(b)).
3. LAN などにおいて複数の物理回線を束ねて仮想的に一つのリンクとして扱い通信をするもの (図 1.1(c)).

本論文で述べるマルチパス通信は 1. のことを指す.



(a) 送信者 (受信者) が複数の IP アドレスを用いて通信を行うもの

(b) 経路上でパケットの割り振りを行うもの

(c) 物理回線を仮想的に一つのリンクとして扱い通信を行うもの

図 1.1　マルチパス通信の分類

通信品質を表す指標として通信帯域，通信遅延，通信遅延ゆらぎ，パケットロス率などが挙げられる．通信遅延揺らぎとは，各パケットの間隔がどれくらい異なるかを表したものであり，Voice over Internet Protocol(VoIP) といったアプリケーションの場合に問題となる [23]．モバイル端末上では様々なアプリケーションが稼働しておりアプリケーションごとに通信品質の必要要件が異なるが [34] [12]，本論文では WEB アプリケーションを想定することにする．WEB アプリケーションでは HTTP と呼ばれるプロトコルを主に使用しており，HTTP は WEB アクセス以外にビデオストリーミングにも利用され，今後の新しいサービスやアプリケーションにも利用されると言われている [28]．そのため，WEB アプリケーションを対象とすることでその他のアプリケーションにも適用が見込めると考えている．

1.2　本論文の構成

本論文の構成は以下のようになっている．第 2 章では，日本における無線環境の実態とマルチパスの重要性を述べ，第 3 章では，マルチパス通信を行う関連研究 · 技術について紹介する．第 4 章では，アプリケーション層においてマルチパス通信を行う手法を提案し，第 5 章では，第 4 章で述べた提案手法の実装に関して述べる．第 6 章では提案手法と既存研究との比較を行い，第 7 章で考察を行う．第 8 章で結論を述べる．

第 2 章

マルチパス通信の重要性

2.1 日本における無線通信

ノートパソコンや一部のフィーチャーフォン，スマートフォンといったモバイル端末は複数の無線モジュールを持つようになった．モバイル端末が接続可能な無線ネットワークとして，3G·WiMAX·LTE(Long Term Evolution)·Wi-Fi などが挙げられる．Wi-Fi を除く無線回線は通常キャリアが提供しており，キャリアと契約を結ぶことで通信を行うことが可能となる．Wi-Fi はキャリアが提供する場合もあるが，ユーザが有線回線の末端に Wi-Fi ルータを各自設置する場合もある．キャリアが提供する無線インフラは会社ごとに無線のカバー範囲や通信品質が異なり，料金体系も多数存在する．課金の種類は大きく四つに分類される．従量制 · 定額制 · キャップ制 · 定額従量制である．キャップ制とは従量制に上限金額が設定されており，その上限金額を支払った場合はいくら利用しても金額が変わらない料金体系である．定額従量制とは基本料金に利用料金が含まれており，利用時間や通信料の超過した部分に対して従量制で課金する料金体系である．

定額制やキャップ制の場合は同じ上限金額を支払っているユーザにおいて通信量が大きく異なることがあるため，キャリアによっては通信量に応じて帯域制限を施すことがある．帯域制限を行うキャリアの通信速度を測定した所，キャリアの公開する実施概要に記されている通り実際に午前 2 時までの帯域が制限されていた (図 2.1)． この事例から，モバイル端末とその通信の増加によって，モバイルネットワークの混雑が避けられない状況になっていることが分かる．トラヒックの抑制の実施として通信量が多いユーザに対する帯域制限以外に，帯域制限が初めから行われている契約形態を設けている場合もある．通信可能なエリアは 3G 回線などと同一であり，通信速度の上限を 3G 回線よりも低くしたもので，回線契約の費用を低く定めることが多い．

キャリアが提供する無線回線の通信品質は通常ベストエフォート型であり，キャリアが公表する通信帯域と実際の通信帯域は異なる．通信品質の低下を引き起こすボトルネックはキャリア内外の両方に存在し，どちらで発生しているのか測定を行う手法は存在するが [3, 4, 9, 19]，本論文ではキャリア内におけるボトルネックに着目する．3G ネットワークを支える技術として UMTS(Universal Mobile Telecommunications System) があり，キャリア内で発生するボトルネックとは図 2.2 における無線ネットワークサブシステム (Radio Network Subsystem:RNS) およびコアネットワーク上で発生するものを指す [20].

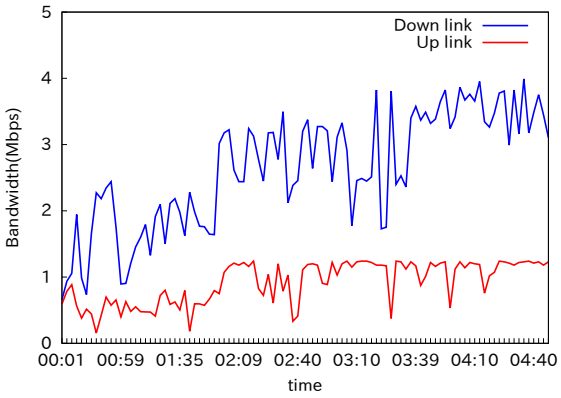


図 2.1　実際に行われた帯域制限の様子

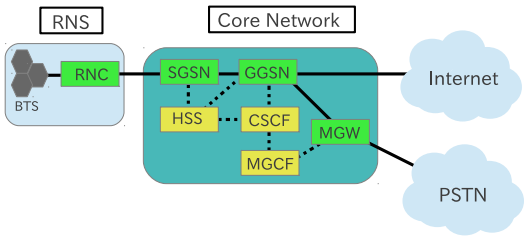


図 2.2　UMTS の構成図

2.2　通信障害のモデル化

何らかの原因により通信品質が低下することをここでは通信障害と表現する．ユーザが複数回線を使用している場合に遭遇する通信障害のモデル化を行う．

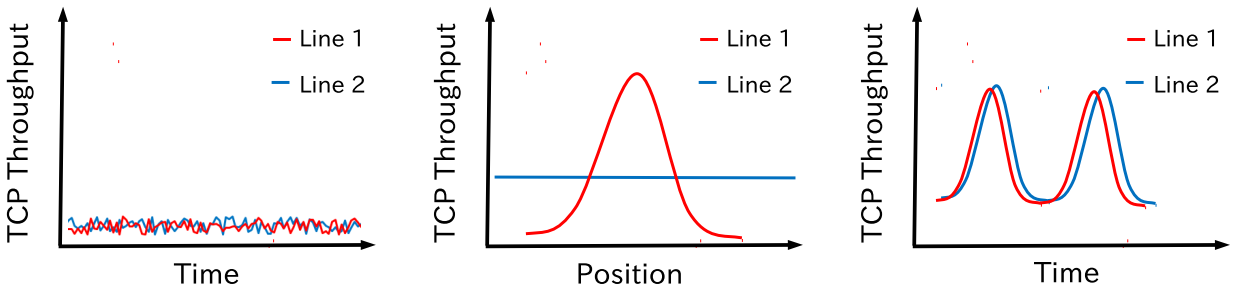
まず，いずれの回線も通信品質が低いという通信障害がある (図 2.3(a))．これは，ユーザのいる地点に多数の他のユーザが存在しアクセスポイントに高負荷がかかっている場合や，ユーザが地下といった電波の届きにくい場所にいる場合に発生する．この通信障害の特徴は，ユーザが複数の無線通信網への接続を契約しており通信網をユーザが選択できる状態であっても，それぞれ通信環境が悪く通信ができないという点である．

次に，位置により帯域が大きく変動する回線において，その帯域が低い所にユーザがいるという通信障害がある (図 2.3(b))．接続可能な回線が複数ある場合にモバイル端末は静的に接続先を設定するものが多く，優先度の高い回線において位置によって帯域が小さくなるとユーザは十分に通信が行えなくなる．この通信障害の特徴は，モバイル端末が静的に接続先を設定するのではなく，実際の回線の品質に応じて回線を切り替えることによって回避が可能というものである．

ユーザが電車や車などに乗って移動する場合にも通信障害が発生する (図 2.3(c))．この通信障害の特徴は，ユーザの乗車している移動体が移動している場合はどの回線も通信障害が発生し，停車している場合はどの回線も帯域が元に戻るという点である．

2.3　マルチパス通信の利点

マルチパス通信の利点は主に通信品質の向上である．複数の回線を集約することで通信帯域を向上することが可能であり，アプリケーションに応じた回線の使い分けをすることも可能である [14]．また，冗長化としても活用可能であり，いずれかの回線が通信障害を起こした時に他方の回線で通信品質の確保も可能となる．マルチパス通信の他の利点としては，トラヒックオフロードが挙げられる．キャリアはモバイル端末の増え続けるトラヒックに備え，ユーザに無線ルータを条件付きで配布 (またはレンタル) を実施



(a) いずれの回線も帯域が小さい場合　(b) 位置により帯域が変動する回線がある場合　(c) ユーザの移動によって帯域が変動する場合

図 2.3　通信障害のモデル化

し，公衆無線 LAN のサービスに取り組むことで 3G 回線のトラヒックオフロードを行っている．ユーザも使用可能なデータ量に制限がある契約をしている場合はその他の回線を使用することで制限に掛からないようにすることが可能となる．マルチパス通信を行うことで，キャリアやユーザはより柔軟なトラヒックオフロードが可能となる．

第 3 章

関連研究

3.1 はじめに

複数の無線通信網を扱う環境が整ってきたため，既にマルチパス通信を行う商品もいくつか開発されている．例えば動画ストリーミング向けのアップロード専用端末が挙げられる．これはカメラで撮影した動画データをあるサーバにアップロードする際に使用するもので，その端末は複数の無線網を集約しつつサーバにアップロードを行うというものである．その他には，ラップトップ向けのものとして Dispatch [31] と呼ばれるものがある．これは USB ハブの形式を取り，Dispatch に接続された複数の Wi-Fi モジュールやデータ通信用端末を集約して通信を行うものである．しかしスマートフォンでは無線モジュールが中に組み込まれているため，このデバイスを現状扱うことはできない．

本研究ではマルチパス通信のためのハードウェアが実装されていないスマートフォンにおいても適用可能なマルチパス通信の実現を検討している．そこで，ソフトウェアにおける既存研究について以下に詳しく述べる．

3.2 ネットワーク層におけるアプローチ

3.2.1 mobile IP を拡張したマルチパス通信

インターネットで広く用いられている TCP/IP で使用される IP アドレスはノードの識別子 (ID) として扱われているが，IP アドレスはノードが所属するネットワークから与えられたものであり，接続するネットワークが切り替わった場合などにおいて IP アドレスが変更されると，そのノードが行なっていた通信が途切れるという問題があった．mobile IP [26] と呼ばれるプロトコルではノード (mobile node:MN) に対して接続するネットワークに依存しない一意の識別子 (home address:HoA) を割り当て，通信相手 (CN) にはその識別子を用いて通信をさせることで，MN が実際に通信に使うアドレス (care-of address:CoA) が変化した場合でも通信の継続を実現している．MN に HoA を割り当てるのは自宅代理人 (home agent:HA) と呼ばれるもので，HoA 宛に届いたパケットを MN が持つ CoA 宛に転送を行う．

mobile IP は無線通信時に発生するハンドオーバーに対して用いられるが，伊東らが提案した mobile IP SHAKE [35] では，一つの HoA に対し複数の CoA を登録することで HA と MN 間のマルチパス通

信を実現した．登録する CoA は必ずしも同一の MN が持つ必要はなく，複数のモバイル端末でクラスタを生成し，そのクラスタ内のノードが持つアドレスを CoA として登録することを伊東らは想定している．図 3.1 に mobileIP SHAKE の概要を示す．MN1 と MN2 は bluetooth や Wi-Fi といった近距離無線通

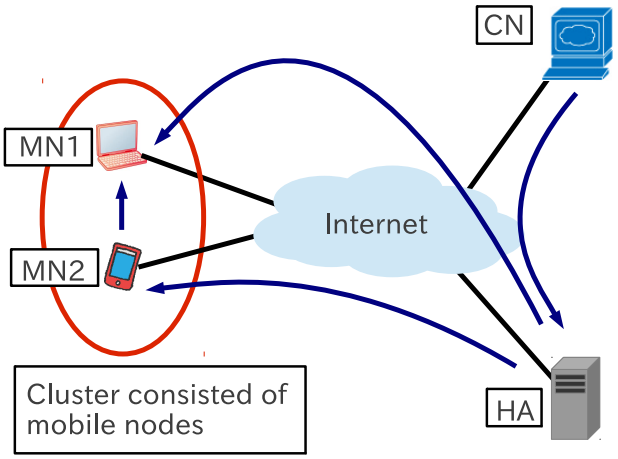


図 3.1　Mobile IP SHAKE の概要

信によってクラスタを構築する認証を行い，正しく認証が行われた場合に MN1 は自身の HA に MN2 の CoA を登録する．CN から受け取った MN1 宛のパケットを HA は MN1 もしくは MN2 にパケットをカプセル化して転送する．MN2 が MN1 宛のパケットを受信した場合は近距離無線通信によって MN1 に再転送を行う．HA から各 MN へのパケットの割り振りに関して三つの方式が提案，実装されている．

1. HN と MN 間の帯域に応じて分配する方式
2. HN と MN 間の遅延に応じて分配する方式
3. 仮想的な送信キューの待ち時間と遅延から送信パケットの到着時刻を予想する方式

遅延と帯域を考慮した方式 3 を用いることで通信品質の異なる回線による影響の削減が行われた．また，TCP を用いて通信を行なった場合に遅延差の影響によりパケットの順序が入れ替わり不要な ACK が発生するが，各経路ごとにシーケンス番号を管理するという輻輳制御も研究されている [36].

その他に，mobile IP を用いた研究として，階層化モバイル IPv6(HMIPv6 [32]) において MN の持つ複数の無線インターフェースを活用するトラヒック経路集約の手法が提案された [2] [21]．HMIPv6 とは IPv6 におけるモバイル IP(MIPv6 [27]) を拡張したプロトコルで，Mobility Anchor Point(MAP) と呼ばれる HA と同様の機能を持つノードを用いる．MAP は MN が異なるネットワークに移動したことを HA に隠蔽する．つまり，MN の CoA が変更された場合に MN は HA ではなく MAP に CoA を登録する．これによりハンドオーバー時の瞬断を削減することが可能となる．提案方式 (図 3.2) では MN が複数のインターフェースを持ち，同時に複数のアクセスポイントに接続できる場合に HA と MAP にてマルチパス通信が行えるようプロトコルを拡張した [37]．HA は MN への経路としてどの MAP にパケットを送るのかという情報を保持し，MAP は MN への経路としてどのルータにパケットを送るのかという

情報を保持している．CN はパケットを HA に送り，HA と MAP は保持する経路表と経路の通信品質によってトラヒックの分配を行う．経路の通信品質は HA-MAP 間と MAP-MN 間でプローブパケットを流すことで測定することを想定している．しかしながら，実運用の面でいくつか問題がある．まず，日本国内のキャリアは IPv6 に対応していないため端末は IPv4 を用いており，HMIPv6 を導入することができない．また，トラヒックの分配を行うために MAP-MN 間と MAP-HA 間の通信品質を HA が把握する必要があるため，キャリアの協力が必要になる．キャリアは自ネットワークの負荷を減らすため通信品質を下げて HA に伝える可能性があり，実際の通信品質に応じてトラヒックの分配を行うことが困難である．

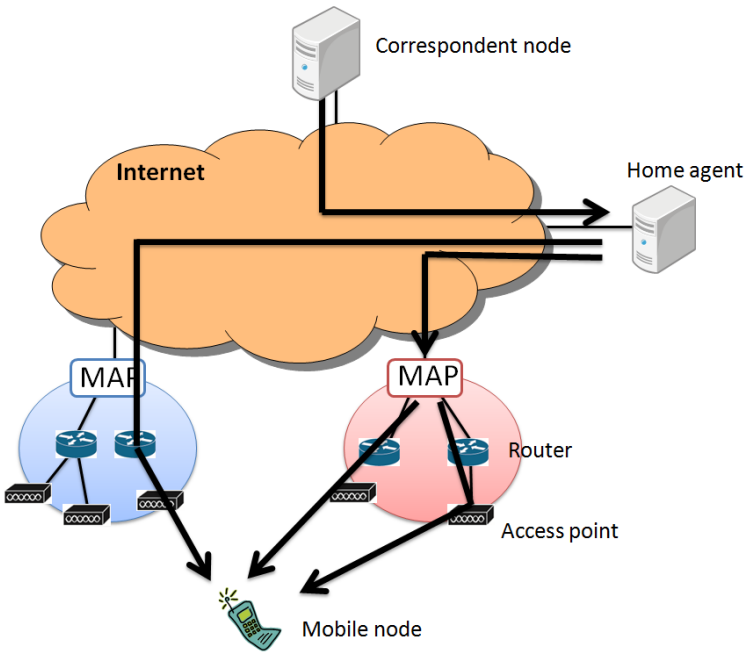


図 3.2　HA 上と MAP 上におけるトラヒックオフロード

mobile IP を用いたマルチパス通信は以上のように HA や MAP 上でパケットの分配を行うことで実現されている．しかしながら，mobile IP は様々な実装がなされているものの標準でサポートしている OS が少なく，mobileIP SHAKE や MIPv6 における経路集約は普及に時間がかかると考えている．

3.2.2　ネットワークプロキシを活用した手法

mobile IP SHAKE と同様にサーバとクライアントの間にパケットの割り振りを決定するプロキシサーバを置き，プロキシサーバとクライアント間でマルチパス通信を行う手法が研究されている [11] [6] [10] [7]．[6] はその上の層で TCP が使われることを想定しており，[11] [10] [7] では UDP を想定している．

TCP を想定した手法では，パケットの順序入れ替えと再送処理が発生しにくい PET(Packet-Pair based Earliest-Delivery-Path-First algorithm for TCP applications) と呼ばれるパケットの配分アルゴリズムと，パケットの順序入れ替えが発生した場合に正しい順番に戻して TCP レイヤーにパケットを渡

すバッファ管理 (buffer management policy:BMP) を用いてマルチパス通信を実現した．マルチパス通信を行うにあたり既存のトランスポート層やアプリケーション層の変更が少ないというメリットがあるものの，TCP において問題となるパケットの順序入れ替えによってスループットが低下するため，PET と BMP を用いた場合でもアプリケーションレイヤーで TCP コネクションを複数用いる手法と比較すると性能が出ないことが実験で示されている．

UDP を想定した手法では，プロキシサーバ上でプロキシサーバとクライアント間の遅延と帯域を監視し，帯域に応じてパケットの分配を行ない遅延に応じて送信時に遅延を挟むことで，より多くのデータを順序入れ替えが発生しにくい形で送信を行なっている．仮に到着パケットの順序が入れ替わった場合はパケットロスとして扱い再送などは行わない．プロトタイプ実装では，遅延の監視は定期的にクライアントに TCP パケットを送信し ACK の到着時間により測定を行い，複数ある経路のうち遅延が最大の経路に合わせて他の経路にも遅延を挿入している．また，帯域の監視はクライアントから届いたデータパケットを元に行い，帯域の割合が $x : y$ の場合は分配するパケットも $x : y$ にしている．パケットの分配はアルゴリズム 1 に従って行う．パケットの分配にあたり送信ベクトル V を定義する．$V = \{0, 1, 2, ..., m\}$ の

アルゴリズム 1 createSendVector(n, w_0, w_1)

Input: Vector length $n \in \mathbb{N}_{>0}$ and weights $w_0, w_1 \in \mathbb{R}_{\geq 0}$

Output: Send vector V of length n

```
V = zeros(n); { initiatize V with n zeros}
r = w1/(w0 + w1); { calculate weight ratio}
r = round(r * n)/n; { adjust r such that r * n is an integet}
for i = 1 to r * n do
  V(⌈i/r⌉) = 1;
end for
```

時，クライアントが持つネットワークインターフェース (NIC)0〜m に対してラウンドロビン方式で送信し，$V = \{0, 1, 0, 1, 0...\}$ は NIC 0 と 1 に対して交互に送信することを表している．V のサイズ n はあらかじめ余分に用意し，その n と測定された帯域 w_0, w_1 を入力とし，V を更新する．クライアントが NIC を三つ以上持っている場合は $w_0 + w_1$ を一つの帯域と見なして，それと w_3 を新たに入力することで V を更新する．評価として，単純なラウンドロビンを用いた場合と比較すると，パケットロスが 25% から 2.2% に軽減され，帯域は 31% 向上した．

3.3　トランスポート層におけるアプローチ

3.3.1　TCP を拡張したマルチパス通信

トランスポート層のプロトコルとして使われる TCP は標準でマルチホーム機能がないため，TCP を拡張しマルチホーム機能を持たせるという研究がなされている．

pTCP

H.Hsieh らが提案する pTCP [16] ではトランスポート内にて TCP-v(TCP-virtual) と呼ばれるコネクションをネットワークインターフェースごとに作成しマルチパス通信を行う．pTCP は TCP-v のラッパーとなり，アプリケーション層には TCP と同等の API を提供する．各 TCP-v はそれぞれ仮想送信バッファと仮想受信バッファを持ち，輻輳制御やパケットロス発生時の処理などは通常の TCP として振る舞う．TCP-v はデータを送信する場合に pTCP に $send()$ という命令を発行し pTCP を通して IP 層にデータを渡す．IP 層からデータを受信した場合に pTCP は $recv()$ という命令を発行し適切な TCP-v にデータを渡す．また pTCP はインターフェースの接続 (切断) を感知し $open()(close())$ という命令で動的に TCP-v の追加 (削除) を行う．TCP-v において輻輳制御は ACK が重複した場合はウィンドウサイズを半分にし ACK のタイムアウトが発生した場合はウィンドウサイズを 1 に戻す．また輻輳の発生した際には送信バッファ上のデータを一度 pTCP に戻し，TCP-v は再度データが割り当てられるまで待つという仕組みを取っている．評価として，アプリケーション層でマルチパス通信を行うものを作成し，それと比較している．帯域が変動する場合，アプリケーション層で実装したものは集約する回線数の増加に伴いスループットは低下したが pTCP ではほぼ理論値の性能が出ている．通信可能なインターフェースが一時的になくなるというハードハンドオーバーに対しても接続の維持が行えている [15]．しかしながら，実機にて実装を行われておらず，3G 回線や Wi-Fi と組み合わせた実環境下でどのように振る舞うのか明らかにされていない [5]．

Multipath TCP(MPTCP)

MPTCP は IETF が標準化を行なっているプロトコルである [13]．MPTCP には以下の特徴により TCP との互換性が保たれている [29]．

1. NAT やファイヤーウォールといったミドルボックスを通しても機能しなければならない
2. 既存の TCP アプリケーションに改変なく使用できなければならない
3. 通常の TCP へのフォールバックを行えるようにする必要がある

MPTCP の接続には TCP と同様の手順が踏まれ，その際に MP_ CAPABLE という TCP オプションを使う．通信相手が MPTCP に対応していない場合は通常の TCP として通信を継続する．MPTCP のコネクションが確立された後にいずれかのノードが IP アドレスを複数持つ場合に MP_ JOIN という TCP オプションを使った SYN パケットを送信することで新たなコネクションを張る．MPTCP では再送処理のため 64 ビットの Data Sequence Number(DSN) と呼ばるシーケンス番号を用いる．同一の DSN を持つデータを他の通信 (サブフロー) を通して送信することで再送を実現している．また各サブフローは 32 ビットのシーケンス番号を用いている．

S.C.Nguyen らはヘテロジニアスな環境下で MPTCP の測定を行なった [25]．有線と有線，有線と Wi-Fi，Wi-Fi と 3G 回線の 3 通りについて測定した所，ホモジニアスな環境下では帯域が大きくなったが，ヘテロジニアスな環境では帯域が小さくなった．つまり，品質の良い回線にて TCP 通信を行なった方が性能が良いという結果になった．

MPTCP は Linux 用のソースコードが公開されており，またいくつかの Android 端末でも動作が確認されている．しかしながら，トランスポート層のプログラムはカーネルに実装されることが多いため，Android OS 上で用いる場合には端末に使われている Android OS のソースコードを入手しインストールするか，メーカー側がアップデートファイルを配布しなければならないという問題がある．

3.3.2　SCTP を拡張したマルチパス通信

SCTP [33] とは TCP と同様に到着順序を保証する信頼性のあるプロトコルであり，また，標準でマルチホーム機能をサポートしている．回線が混雑している場合に輻輳制御は TCP より性能が良い [24]．標準化されているプロトコルにおいてマルチホーム機能は冗長性のためのものであり冗長回線を用いて新しいデータを送信することは推奨されていない．そこで SCTP において冗長回線を用いた帯域集約を行うプロトコルである CMT(Concurrent Multipath Transfer) が提案された [17] [18]．CMT の実現にあたり送信者の輻輳ウィンドウ (cwnd) が予想された値より大きく下回るという問題があったが，以下に示す三つの欠点を解決することで改善された．

1. 送信者による不必要な再送
2. 送信者側における cwnd の更新が少ないことによって生じる cwnd 増加の抑制
3. ACK トラヒックの増加

1. という問題に対して，SACK に付随するパケットの損失情報 (gap report) とそれぞれのパケット (正確には，データがチャンクされた TSN) の宛先アドレスを元に，パケットロスなのかパケットの順序入れ替えなのかを判断することで不必要な再送を抑制している．2. という問題は，新しいデータに対する ACK のみに従って cwnd を更新していたことが原因であり，TCP や SCTP のようにひとつの宛先に対して新しいデータを送信するという方式では機能するものの CMT のような複数回線を用いる場合では機能しなかった．そこで，CMT ではデータをどの宛先に送信したのかをトラッキングすることで回線に応じた cwnd の増加を行うようにした．3. という問題は，SCTP や TCP においてパケットが順序通りに受信できた場合に ACK の送信を遅らせること (Delayed ack) で ACK トラヒックを削減してきたが，CMT では複数回線を集約しているためパケットが順序通りに届かないことが多く ACK パケットが増加するというものである．そこで，CMT では順序通りに届かない場合でも ACK の送信を遅らせることで ACK トラヒックの削減を行なった．

SCTP を拡張したマルチパス通信のメリットは SCTP 自体にマルチホーム機能をサポートしているということが挙げられるが，欠点として TCP や UDP と異なるプロトコルであり既存のアプリケーションの改変が必要となる [25]．また MPTCP と同様に，カーネルに CMT が実装された場合にスマートフォンへの対応において問題が発生する．

3.4　アプリケーション層におけるアプローチ

ネットワーク層やトランスポート層でマルチパス通信を実現した場合に，その実装はカーネルに行われることが多く，既存のモバイル端末への適用が困難であった．そこで既存のオペレーティングシステムや

プロトコルスタックへの改変が必要ないアプリケーション層におけるアプローチとして ARMS が提案された [39]．ARMS では SCTP のソケット API を活用してマルチパス通信用の $ARMS\_snd\_rcv()$ という関数を作成した．ソケット API である $sctp\_getpaddrs()$ を用いて通信相手の IP アドレスを取得し，$sctp\_sendmsg()$ において $SCTP\_ADDR\_OVER$ フラグを立てて送信することで冗長回線 (セカンダリパス) を用いてデータを送信する．複数経路のデータの割り当てに対して，cwnd を用いることを検討していたが，CMT の場合と同様にセカンダリパスの cwnd が正しく増減していないため性能が出なかった．そこで，データ転送に際にセカンダリパスはまずダミーデータを送信し帯域の測定を行い，プライマリパスとの帯域比に応じてデータの分配を行うという手法を用いることで改善を行なった．実験において，20MB と 200MB のダウンロード時間の計測を行なった．20MB の場合は帯域の測定に時間がかかるため元の帯域が大きい場合にダウンロード時間が長くなった．200MB の転送では 8Mbps と 2Mbps というヘテロジニアスな環境でもダウンロード時間は短くなった．ARMS を用いるメリットは SCTP のソケット API のフラグを用いることでマルチパス通信を行なっているため導入がしやすいという点である．ARMS にはいくつか課題が残っており，まず実装されたものは 50MB 転送ごとに帯域の再測定を行なっており，その際に 3 秒ほどシングルパス通信となる．また，データを割り振った後に片方の通信が完了するとデータ転送量が 50MB に到達するまでは片方の通信を待つことになるためシングルパス通信となる．以上をまとめると AMRS のデメリットは転送量が小さい場合に冗長回線が十分に使えないということと，今回の実装ではデータ転送の前にデータの分配を行うため，無線環境のような動的に帯域が変化する場合に未対応であるということである．また，Windows やスマートフォンで用いられる OS では SCTP は標準でサポートされていないため，ARMS を使用するにはまず SCTP に対応させなければならないという問題がある．

3.5　おわりに

本章では，マルチパス通信をソフトウェアで実現する既存研究を OSI 参照モデルの層ごとに分類し，その特徴を述べた．トランスポート層以下の層で実装がされた場合にカーネルに実装することが多く普及が難しいことを述べた．カーネルの変更が必要ない手法として ARMS を紹介したが，WEB アクセスのような小さなファイルをダウンロードする場合や無線環境のような帯域が変化する場合に性能が低下するという問題があった．スマートフォンの 3G 回線と Wi-Fi の帯域を集約するマルチパス通信の実現には OS のサポートを必要とせず，ヘテロジニアスな環境でも帯域を活用できる手法が必要となるが，既存研究においてこの条件を満たすものは存在しない．

第 4 章

提案手法

4.1 はじめに

第 3 章で述べたように，OS に手を加えずにマルチパス通信を行うにはアプリケーション層で実現することが望ましい．また，既存研究である ARMS において輻輳制御方法とデータの割り当て方法が問題となっていた．そこで筆者は，以下の手法を提案する．

1. 複数回線の輻輳制御を分けて行う
2. データを細かく分割して，そのブロックごとに転送を行う

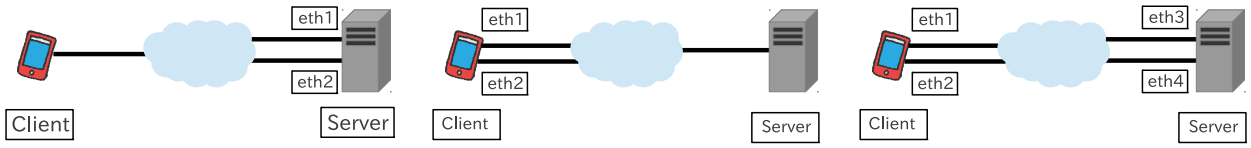
SCTP においてプライマリパスとセカンダリパスは一つのコネクションとして扱われていたため，改良前の CMT や ARMS において cwnd が正しく反映されていなかった．そこで，複数回線があった場合に，複数回線それぞれに対してコネクションを張り，各回線にて通常の通信を行わせる．これにより，一方の回線において輻輳が発生した場合でも他の回線に影響を与えないことが保証される．次に，それぞれの回線へのデータ割り当てを静的に行うと複数送信を完了した回線の帯域が活用できていないという問題に対して，動的に割り振りを変えることができるよう回線に割り当てるのは分割されたデータにし，送信が完了した回線に次のデータを再度割り当てを行うことでデータの転送が完了するまで複数回線の帯域を活用する．この手法は帯域の測定が必要ないため通信開始から複数回線にて通信が可能である，この手法を実現するにあたり考察すべき点を以下に詳しく述べる．

4.2 トランスポート層の選定

ARMS では SCTP を用いることでマルチホーム機能をトランスポート層に委託していた．しかしながら，SCTP はスマートフォンにおいて標準でサポートされておらず，OS の改変が必要となる．一方で TCP は標準でマルチホーム機能を有していないが，アプリケーション層で複数回線の同期を実装することでマルチホームとして動作が可能である．そこで，本研究では既存の OS において標準でサポートされている TCP を選択することにする．

4.3　TCP のマルチパス通信への対応

TCP においてノードとノードを複数の IP アドレスで一つのコネクションを張ることを想定していない．しかしノードが複数の IP アドレスを用いてそれぞれコネクションを張ることは可能である．ノードが複数の IP アドレスを持つパターンは図 4.1 の 3 通りである．サーバのみが複数の IP アドレスを持つ



(a) サーバのみが複数の IP アドレスを持つパターン　(b) クライアントのみが複数の IP アドレスを持つパターン　(c) サーバとクライアントが複数の IP アドレスを持つパターン

図 4.1　ノードが複数の IP アドレスを持つパターン

場合 (図 4.1(a)) は，クライアント側は設定する必要がなくサーバ側で戻り経路フィルタ (Reverse Path Filtering) を無効にすることで eth2 に来たパケットに対して eth1 を使用してパケットを送り返すことができる．eth2 に来たパケットに対して eth2 を用いてパケットを送り返すにはソースルーティングと呼ばれるポリシーベースルーティングを行う必要がある．クライアントのみが複数の IP アドレスを持つ場合 (図 4.1(b)) も同様である．サーバとクライアントがそれぞれ複数の IP アドレスを持つ場合 (4.1(c)) は，上記二つの手法に加えて，クライアント側では eth4 へのパケットを eth2 が所属するネットワークのゲートウェイに送信するよう設定を行い，サーバ側では eth2 へのパケットを eth4 が所属するネットワークのゲートウェイに送信するよう設定を行うことでそれぞれコネクションを張ることができる．これは通常の宛先ベースルーティングを使用した方法である．本研究の対象となるのはモバイル端末が持つ複数の回線の活用であるため図 4.1(b) と図 4.1(c) の構成となる．また，一方の回線が切断されても他方で通信を行うことを想定すると戻り経路フィルタを無効にするという非対称経路ルーティングは適切ではない．本提案手法でマルチパス通信を行う場合はクライアント側でソースルーティングを行うか，サーバ側にも複数の IP アドレスを持たせ宛先ベースルーティングを行う必要がある．つまり，ライブラリなどがソースルーティングに対応していない場合はサーバとクライアント両方が複数の IP アドレスを持つ必要がある．既に実装されている MPTCP ではソースルーティングを用いており，サーバ側が複数の IP アドレスを所有していなくてもクライアントはマルチパス通信を行うことが可能である．

4.4　イニシエーション

複数回線の輻輳制御を独立に行うため，本提案手法では複数の回線それぞれにコネクションを張る．そのため，別々に接続されたコネクションを同一のものと認識する仕組みが必要である．TCP を拡張した MPTCP では SYN パケットに MP_CAPABLE というオプションを用いており，トランスポート層にてイニシエーションを行なった．しかしながら，SYN パケットにオプションを追加する場合にインターネット上のパスにおいてそのオプションが取り除かれるという可能性がある [29]．アプリケーション層でイニシエーションを行う場合は TCP のペイロードに接続情報を持たせることができるため，本提案手法では，イニシエーションの情報は実データと同じように送受信を行う．詳細は第 5 章にて述べる．

4.5　分割転送

複数回線による帯域の集約においてデータの分配方法は重要な要素となる．既存研究において分配方法は以下の 3 通りであった．

- パケットごとに割り振る [35] [6] [13] [17]
- 静的にデータを割り振る [39]
- データを細かく分割して，そのブロックを割り振る [16]

1. の場合のメリットは再送時に重複するパケットが少ないということであり，デメリットは新たにシーケンス番号を付加する必要がありペイロード内にシーケンス番号を埋め込む場合は帯域の低下に繋がるということである [30]．2. の場合のメリットは実装が用意であるということであり，デメリットは帯域が変化する環境下では十分に帯域を活用できないということである． 3. の場合のメリットは輻輳制御が既存のトランスポート層のものを使えるということであり，デメリットは再送の単位がブロックのままだと重複するデータが多く帯域が低下する可能性があるということである．

集約する回線が無線であるため，1. または 3. が適している．また，パケットごとに割り振るというのはトランスポート層の動作であるため 3. が適切であると考えられる．集約する回線が 2 本でデータ分割が仮に 10 等分割であったとすると最悪の場合には 10% のデータが重複し，20 等分割の場合は 5% である．これを重複が発生しなかった場合と共に平均を取ると 3〜5% になる．帯域の集約においてデータの重複にようる影響は少ないのではないかと考えている．しかしながら，集約する回線数が 3 本以上の時は重複するデータは増加する．

次に分割したデータの割り当て手法に関して，アルゴリズム 2 を提案する．まず分割数を入力とする．分割数はファイルサイズによらず一定，もしくはサーバとクライアントは通信し合い事前に同期されているとする．$Data[i]$ は分割後のデータを表し，添字の i は i 番目のデータであることを表す．送信側では回線ごとにスレッドが立てられており，どのデータが送信されたのか $Data$ の状態 ($status$) によって共有される．一度目の for 文では未送信のデータを送信する．全てのデータが送信中または送信完了になった時点で二度目の for 文を実行する．二度目の for 文では他の回線が送信中のデータを送信する．これが再送として扱われる．つまり，未送信データがなく送信中のデータがある場合には空いている回線を用い

てそのデータの送信を行うというものである．そのため，元々送信を行なっていた回線も送信を続けている．いずれかの回線で分割化されたデータの送信が終わった場合に，残りの送信中の回線は送信の中止を行い，転送されたデータは破棄する．

アルゴリズム 2 の特徴は通信している回線のいずれかが通信障害を起こした場合でも他の回線に影響を及ぼさないことである．欠点は再送の単位が分割後のブロックごとであり，前述の通りデータがいくつか重複するということである．既存研究において通信品質が大きく異なる回線を集約した場合に帯域がシングルパス通信の場合に比べて小さくなることがあったため，本提案手法では通信品質の良い回線が空いている場合に積極的に再送処理を行うこととした．データの重複が発生するため帯域は複数回線の帯域の和とはならないが，複数回線のうち最も良い回線の帯域が保証されると推測している．

アルゴリズム 2 allocateData(n)

Input: Division number $n \in \mathbb{N}_{>0}$

```
for i = 1 to n do
  if Data[i] is unsent then
    Data[i]'s status ⇐ senging
    send Data[i]
    Data[i]'s status ⇐ sent
  end if
end for
for i = 1 to n do
  if Data[i] is sending then
    send Data[i]
    Data[i]'s status ⇐ sent
  end if
end for
```

4.6 おわりに

本章では，既存の OS において標準でサポートされている TCP を用いたマルチパス通信の手法について述べた．複数回線へのデータの割り振りに関して，帯域を測定する手法では通信帯域が動的に変化する環境に対応が困難であるため，動的にデータを割り振るアルゴリズムを提案した．その際に積極的な再送処理を行うことで，複数回線のうち最も良い回線の帯域が保証されることを述べた．

第 5 章

実装

5.1 はじめに

提案手法により帯域の集約が可能かどうかを示すため実際に動作するアプリケーションにて実装を行う．アプリケーションは図 5.1 のように，クライアントノードとプロキシサーバ上で動作するプロキシプログラムである．クライアントノードではバックグラウンドプロセスとして動作させる．クライアントノード上の任意のブラウザにおいて HTTP プロキシサーバとしてローカルホストを設定すると，クライアントノード上でバックグラウンド実行されているプロキシプログラムが要求メッセージ (リクエストと呼ぶことにする) を受け取り，プロキシサーバにリクエストを再転送する．プロキシサーバ上ではリクエストのあった WEB サーバにリクエストの再転送を行い，ファイルのダウンロードを行う．プロキシサーバからクライアントノードにてファイルの転送を行う際にファイルを分割しマルチパス通信を行い，帯域の集約を図る．本アプリケーションを選択した理由は，既存のブラウザソフトウェア (ブラウザ) と WEB サーバソフトウェアに影響を与えないため普及が容易であると考えたためである．

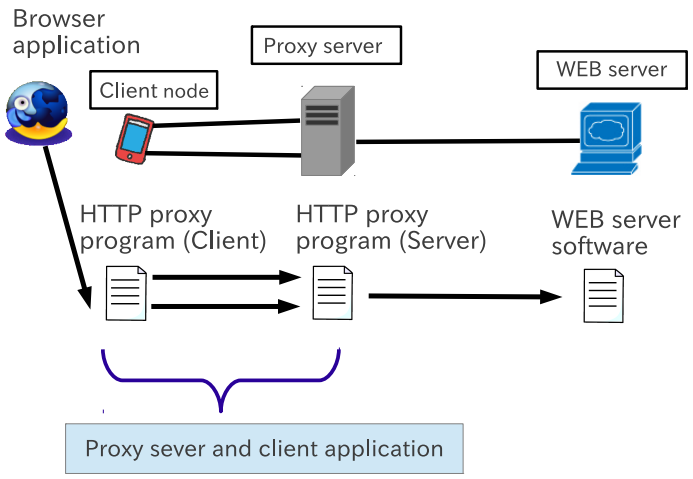


図 5.1　実装を行なったアプリケーションの概要

提案手法ではデータの分配方法とイニシエーションの手法のみ定義しており，実動作に関しては何も

定義していない．しかしながら，WEB アクセスでは比較的小さなファイルをダウンロードすることが多く，複数回線の同期やサーバ · クライアント間の同期にかかる時間が影響を及ぼし，実装方法によってはダウンロード時間が伸びてしまう．そこで，実装を行なったプロトタイプを基に，どのようなオーバーヘッドが存在するか述べ，次にオーバーヘッドを削減した場合の実装方法について述べる．

5.2　プロトタイプの実装

HTTP/1.1 では，ファイル (リソース) の一部を要求する範囲リクエスト (Range Request) が定められている．WEB サーバがこれを許可している場合，クライアントは GET メソッドを用いる際に Range リクエストヘッダを用いることでファイルの一部だけをダウンロードすることができる．しかしながら，これを用いてマルチパス通信を行う場合にいくつか問題点があった．まず，クライアントは取得したいファイルのサイズが未知のためファイルの範囲を指定することができない．HEAD メソッドと呼ばれるヘッダのみの情報を要求するメソッドを用いた場合でも WEB サーバがファイルサイズ (content-length) を返さない場合がある．また，ファイルサイズを取得できた場合でも分割数に応じてリクエストを複数発行するのは WEB サーバに負担がかかる．そこで，クライアントノードとプロキシサーバ上でのみ解釈可能な HTTP ヘッダを定義した (図 5.2)．実際に動作させた場合のヘッダは図 5.3 のようになっている．

```
X-multi-download = "X-multi-download" ":" file-number "-" fragment-number
example: "X-multi-download:5-1"
```

図 5.2　追加した HTTP ヘッダと用例

```
accept-encoding:identity
x-multi-download:1-0
host:192.0.2.2
accept:*/*
user-agent:Wget/1.13.4 (linux-gnu)
connection:close
proxy-connection:Keep-Alive
```

図 5.3　プロキシプログラムを動作させた時のヘッダ

”X-”から始まるヘッダ名は仕様書で定義されることは無いため，常にユーザ定義ヘッダとして使用することができる [8]．定義したヘッダはクライアントの GET リクエスト (リクエスト行が GET メソッドであるリクエストを GET リクエストと表記する) に用いられ，ファイル番号とフラグメント番号という二つの値で構成される．ファイル番号はファイルごとに一意に与えられる番号で，ファイル名が同じであっても異なるデータである場合はファイル番号も異なる．プロトタイプではファイルの分割数を 10 に固定しており，フラグメント番号は 0 から 10 の値を取る．図 5.4 にて動作の説明を行う．フラグメント番号 0 は特殊な値であり，プロキシサーバがそのヘッダを含む GET リクエストを受け取った場合に，HEAD リクエストと似た動作を行う．プロキシサーバは WEB サーバに対して GET リクエストを発行し，HTTP

ヘッダとデータのダウンロードを行うが，クライアントに対しては HTTP ヘッダのみを返す．プロキシサーバがフラグメント番号 1 から 10 のヘッダを含む GET リクエストを受け取った場合に先程ダウンロードを行なった該当ファイルを分割し，クライアントに送信する．この時，プロキシサーバは WEB サーバに GET リクエストを発行しない．クライアントは未受信のデータに対して X-multi-download ヘッダを用いてリクエストを送信し，全てのファイルがダウンロード完了時点でブラウザにデータの転送を行う．以上により，クライアントは事前にファイルサイズを取得する必要がなく，また WEB サーバに複数のリクエストが発行されるのを防いでいる．プロトタイプにおいてアルゴリズム 2 はクライアント上に実装を行なった．つまり，プロキシサーバはファイルの送信状態を保持していない．

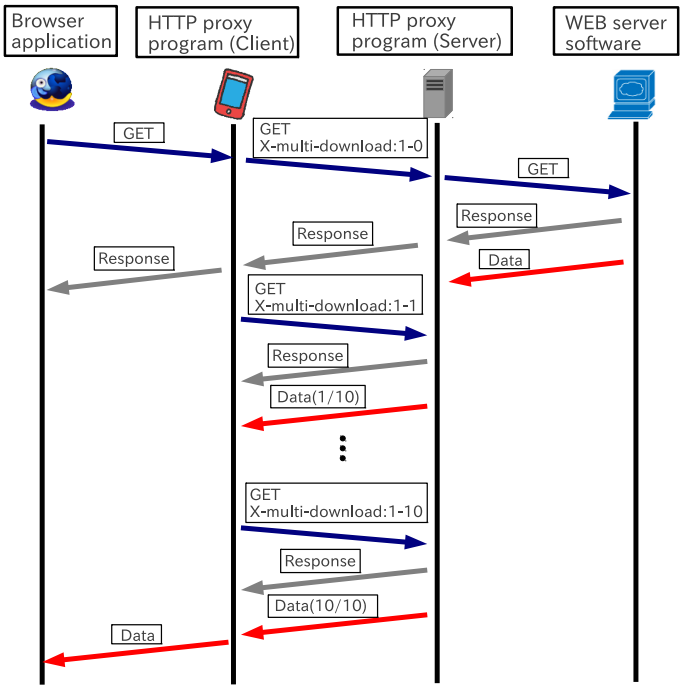


図 5.4　作成したプロキシプログラムの挙動

次に複数回線のイニシエーションについて述べる．プロトライプ実装では，X-multi-download ヘッダがイニシエーションを兼ねている．つまり，異なる回線から同じファイル番号に対するリクエストを受け取った場合に同じファイルを指しているものとして扱う．セキュリティの問題は存在するが，認証機能を付与することで解決できると考えている．ユーザを一人と仮定しているが，X-multi-download ヘッダにユーザ番号を付与することでこちらも拡張が可能であると考えている．

プロキシサーバはファイルの送信状態を保持していないと説明したが，ファイル番号の履歴は保持している．フラグメント番号 0 のリクエストが来た際に，そのファイル番号が初めてのリクエストであれば WEB サーバにリクエストを発行し，初めてではない場合に WEB サーバにリクエストを発行しない．これはクライアントが複数の回線を持つ場合に，回線の数だけ”X-multi-download:ファイル番号-0”というリクエストがプロキシサーバに届くためである．プロキシサーバは WEB サーバからの応答メッセージを保持し，同一ファイル番号でフラグメント番号 0 のリクエストが来た際にその応答メッセージを用いて応答を行う．そのため，クライアント側では複数の応答メッセージを受信することとなる．その際の処理

を図 5.5 に示す．$line1$ と $line2$ は回線を表すオブジェクトであり，その回線が応答メッセージを受信している場合に，$response()$ は応答メッセージに含まれているステータス番号 $(code)$ とヘッダ (hdr) を返し，未受信の場合には $False$ を返す．回線数が複数あった場合でも同様の形で処理が行われ，いずれかの回線から応答メッセージを受信した際にブラウザに応答メッセージを送信し，遅れて来た応答メッセージは破棄する．

```
while line1.response() == False and line2.response() == False:
    pass
code, hdr = line1.response() if line1.response() else line2.response()
```

図 5.5　複数の応答メッセージを受信時のクライアント側での動作

今回は回線数に応じてプログラムを作成した．つまり回線数が 2 本の時と回線数が 3 本の時では別のプログラムを実行しなければならない．回線数に応じた切り替え，および動的な IP アドレスの変化への対応は未対応である．

5.3　プロトタイプにおいて発生したオーバーヘッド

プロトタイプを用いて様々な環境下で測定を行なった所，環境によってダウンロード時間が削減された場合もあったが，帯域の集約が十分に行われていない場合が多く存在した．ここでは特に影響を及ぼしたオーバーヘッドに関して述べる．

プロトタイプでは一つのファイルをダウンロードするためにクライアント側では 11 回の新しい GET リクエストを発行していた（図 5.4）．そのため，TCP におけるスリーウェイハンドシェイクがその度に行われており，往復遅延時間の 10 倍の時間が余計にかかっていた．無線環境では遅延が大きいため，ダウンロード時間に影響があった．また，それぞれが新しいコネクションであるため，TCP の輻輳制御であるスロースタートが毎回発生していた．そのため，ダウンロードをしている時に帯域が十分に活用できていない時間が多く存在した．

プロトタイプで改善すべき点は 2 点ある．まず，一つのファイルをダウンロードするにあたり TCP コネクションを維持するという点である．次に，分割されたデータのどの部分を転送するのかクライアントが指示していたが，この方式ではやはり往復遅延がダウンロード時間に加算されるため，この手法以外の方法でどのデータを転送するのかプロキシサーバとクライアントノードで同期する必要があるという点である．

5.4　プロトタイプの改良

どのデータを送信するのか事前に通知する手法は往復遅延が発生するため，送信するデータにヘッダを追加するという手法に変更した．ヘッダは”フラグメント番号 - データサイズ”という形式にしている．こ

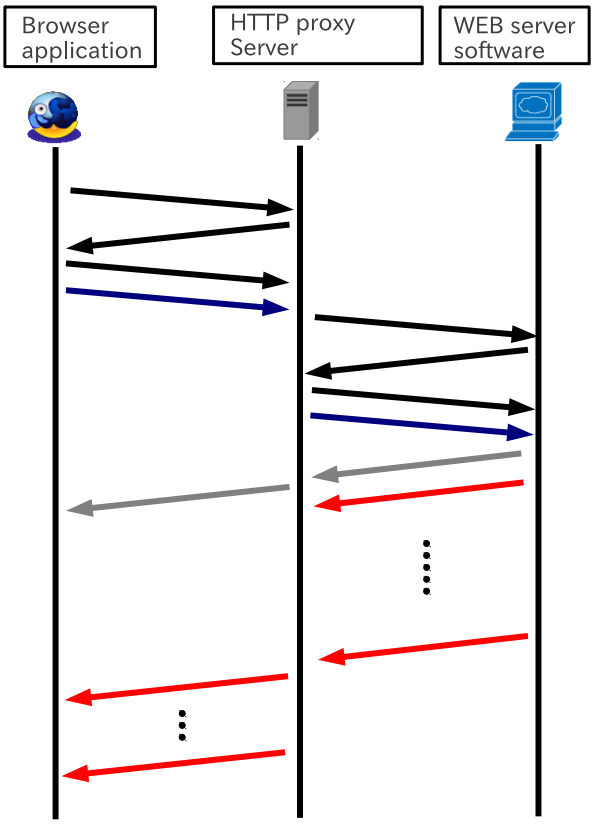
のヘッダは HTTP ヘッダとは無関係である．フラグメント番号とデータサイズは整数で，それを区切り文字 (-) で分けている．後述するが，TCP コネクションとファイル番号はあらかじめ紐付けされているためヘッダ内にファイル番号のようなものは記述しない．また，一つの TCP コネクションを用いるためクライアント側が分割化されたデータの終端を読み取る必要があり，ヘッダにデータサイズを記述することで受信したデータのデコードを行えるようにしている．使用していたライブラリが一つのリクエストに対して一度のデータ転送しか行えなかったため，データ転送用の TCP ソケットを用いて，そちらで通信するようにした．コネクションの確立後にクライアントから受信したいファイルのファイル番号を送信し，その後は受信待ち受け状態となり，分割化されたファイルが全てダウンロードできるまで受信を行う．

改良を行なったプログラムではスリーウェイハンドシェイクおよびスロースタートは通常の HTTP プロキシサーバを介したものと同程度であると予想する．動作を表したものが図 5.6 である．図ではブラウザとプロキシクライアント間の動作を省略している．これは同一ノード内での通信でありノード間の通信遅延に比べて小さいためである．また，複数回線の通信を示すためクライアントノードの左側に別の回線 (line2) の動作が追加されている．いずれの図においてもスリーウェイハンドシェイクの様子を明示し，それ以外の ACK パケットは省略している．実装を行なったプログラムでは，データ通信用のソケットの作成を行うが，コネクションの確立は WEB サーバからプロキシサーバにファイルの転送が終わるまでに完了していればオーバーヘッドがかからないため，図 5.6(a) と図 5.6(b) は同一のタイミングでプロキシサーバからデータが送信されている．そして，line1 では輻輳制御も同様に行われる．line2 という回線が line1 に比べて通信品質が低い場合でも line1 の通信に影響を及ぼしていない．図 5.6(b) では line2 が line1 より遅延が大きい様子を表している．プロキシサーバから同一のタイミングで応答メッセージが送信されるが line2 から来たものは破棄される．またデータに関しても line1 と line2 を用いて同一のタイミングで送信される．line1 と line2 は異なるフラグメント番号を持つデータがそれぞれ送信される．

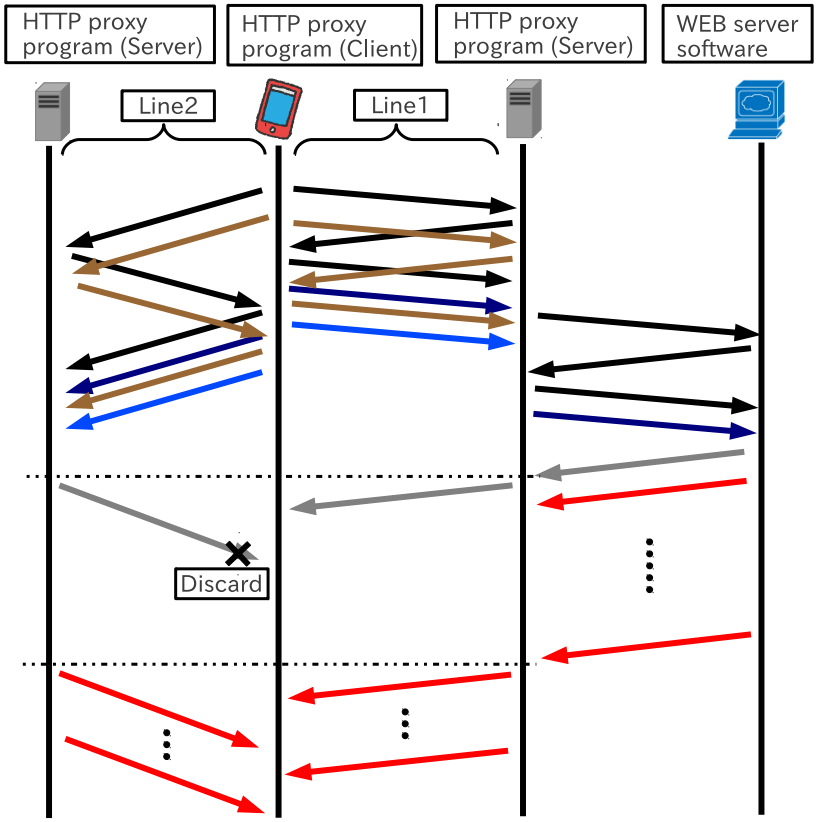
改良を行なったプログラムにおいて発生するオーバーヘッドは 3 点である．まず，新たに付加したヘッダによるデータの増加だが，パケットごとではなく分割されたデータごとに付与しているため，ヘッダのデータサイズは 1KB 程度であり，影響は少ないと考えられる．次に，プログラムの処理にかかる時間である．通常のプロキシとは異なり，データの分割や再構築などを行うため，この処理に時間がかかる．残るオーバーヘッドは，複数回線のソケットを閉じるのにかかる時間である．第 4 章で述べたように，分割されたデータが全て受信完了した際に，送信中 (または受信中) の通信は停止される．今回の実装ではソケットは 8KB ずつ転送を行なっているため，その 8KB が転送完了時点でソケットが閉じられる．そのソケットを閉じる際にかかる時間がどの程度影響を及ぼすか第 6 章で評価を行う．

5.5　HTTP プロキシによるオーバーヘッド

今回実装を行なったものは HTTP プロキシアプリケーションであり，図 5.6(a) においても WEB サーバからプロキシサーバへのダウンロード時間が加算されている．この時間を削減するため，プロキシサーバにおいて，送信予定のデータのダウンロードが完了した時点でクライアント側への送信を開始するという仕組みを追加した．これにより，分割数が 10 の時，HTTP プロキシによって発生するダウンロード時間は約 1/10 となる．この仕組みに関しても第 6 章にて評価を行う．



(a) 通常のプロキシサーバを介した際の挙動



(b) 実装を行なったプロキシプログラムを介した際の挙動

図 5.6　通常のプロキシサーバと実装を行なったプロキシサーバとの比較

5.6　スマートフォンにおける動作検証

本研究の対象はスマートフォンにおける複数の無線網の帯域集約であるため，実機にて正しく動作するのか検証を行なった．動作検証に用いた端末は，以下の 3 機種である (うち 1 機種は異なる二つの OS のバージョンを使用)．これら端末において，3G 回線と無線 LAN(Wi-Fi) の帯域集約を行う．

- L-07C(Android 2.3.3)
- SC-04D(Android 4.0.4)
- SC-05D(Android 2.3.6)
- SC-05D(Android 4.0.4)

まず，マルチパス通信を行うにあたり，複数の回線に同時に接続する必要がある．Android 端末上ではネットワークを制御するプロセスが動いており，使用していない回線は切断される．つまり，Wi-Fi が接続している時は 3G 回線が切断され，Wi-Fi が切断された時は自動で 3G 回線に再接続を行う．そこで，そのプロセスを通さずに Wi-Fi に接続し，自動で 3G 回線の切断を行わないよう操作を行なった．Android は標準で使用できる Linux コマンドが少ないため，操作には Busybox と呼ばれるソフトウェアを用いて Linux コマンドを使用する．Android の設定画面から 3G 回線 (データ通信) をオンにし，Wi-Fi をオフにする．これにより 3G 回線の APN(Access Point Name) が正しく設定されていれば 3G 回線での通信が行える状態になる．次に，insmod コマンドを用いて Wi-Fi のモジュールを読み込む．これは，Wi-Fi をオフにした際に自動でカーネルモジュールがアンインストールされてしまっているためである．モジュールを読み込むと Wi-Fi の NIC 名が ifconfig コマンドで見つかるため，その NIC を ifconfig コマンドで起動する．次に，wpa_supplicant コマンドを用いて Wi-Fi の AP(Access Point) に接続し，その無線 LAN が DHCP に対応している場合は dhcpcd コマンドにて IP アドレスの取得を行う．図 5.7 にその流れと実際に用いたコマンドを示す．

```
# buxybox insmod /etc/wifi/wireless.ko
# busybox ifconfig wlan0 up
# wpa_supplicant -Dwext -iwlan0 -c/mnt/sdcard/wpa_supplicant.conf -B
# dhcpcd wlan0
```

図 5.7　手動で無線 LAN に接続する場合の流れ

複数回線の使用が可能になった後，route コマンドを用いてルーティングの設定を行う．図 5.8 のように Android 上でも Linux と同様に設定が行える．最後にプログラム実行のための環境構築を行う．プログラムをバックグラウンドで実行するため Android のアプリである「SL4A」というものを使用した．

動作が確認できなかったのは SC-05D(Android 4.0.4) のみであった．これは OS のバージョンが変わった際に wpa_supplicant の設定項目が変更されたためである．OS のバージョンが同じである SC-04D

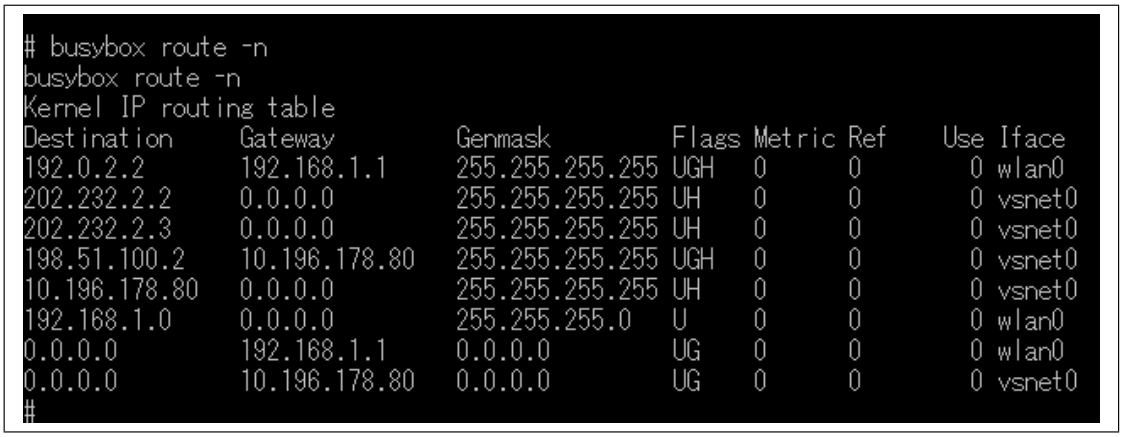


図 5.8　スマートフォンにてルーティングテーブルの設定を行なった時の画面

では動作したため，正しい設定を行うことで SC-05D(Android 4.0.4) でも動作が行えると考えている．

5.7　おわりに

本章では，具体的なアプリケーションに行なった実装について述べた．分割したデータの転送にあたり HTTP/1.1 で用いられる範囲リクエストを拡張した方式ではスリーウェイハンドシェイクおよびスロースタートによってダウンロード時間が増加したため，プロトタイプに改良を加えることでこれを削減した．また，スマートフォンにおける設定と動作検証の結果について述べた．

第 6 章

評価結果

6.1 はじめに

どのような回線を想定するのかを決定するために，まず無線回線における通信品質の測定を行う．測定を行なった無線網をモデル化し，その通信品質下で提案手法と既存研究の評価を行う．まず，同一のアプリケーションを用いて評価を行い，次に提案手法と既存研究それぞれに合わせたアプリケーションを用いて評価を行う．評価するのは以下の 2 点である．まず複数回線の帯域の集約が可能かどうかという点と，品質の異なる帯域を集約した場合に集約する前と比べて帯域がどの程度小さくなったかという点である．提案手法および既存研究では，ヘテロジニアスな環境において帯域が小さくなることがあるため，その評価を行うということである．そして，モデルに基づいて行なったシミュレーションとの比較を行うため，スマートフォンの実機においても同様の測定を行う．

6.2 無線環境の通信品質の測定

NS-2 や OPNET Modeler と呼ばれるネットワークシュミレータで無線の通信品質の測定を行なっている研究は存在するが [22] [1]，キャリアが提供する無線回線は理論値と実測値が大きく異なるため，日本のキャリアに対して実際に通信品質の測定を行なった．測定を行なったのは下りの帯域と遅延とパケットロス率である．

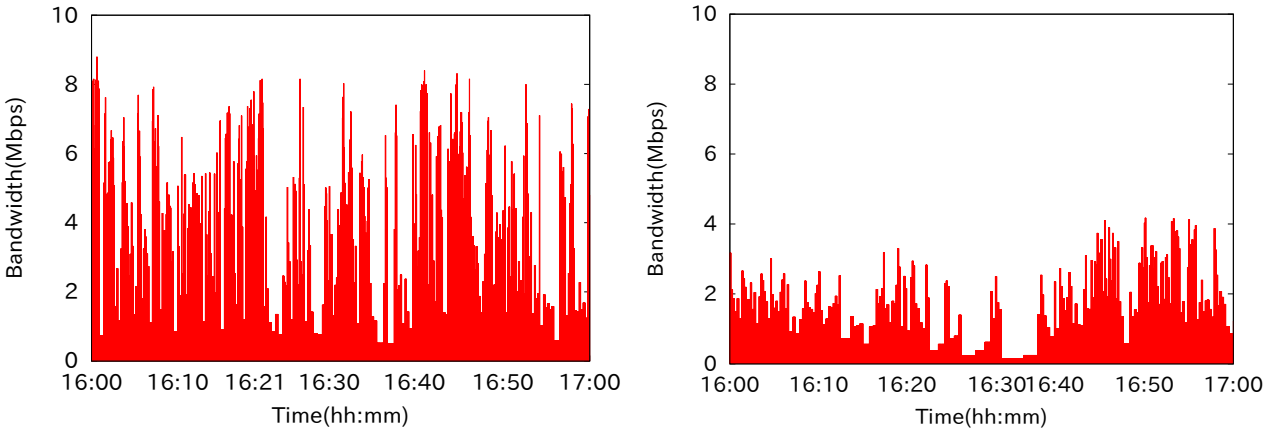
下り帯域の測定

下り帯域の測定は下記条件下で行なった．

- 測定場所
  - 電車内 (山手線)
- データ通信端末
  - 端末 1:MW-U2510(UDO1SS) IEEE802.16e Mobile WiMAX Wave2
  - 端末 2:D22HW HSPA/W-CDMA
- 測定に使用したパソコン

- OS:Windows7
- メモリ：2Gbytes(うち 300Mbytes を RAM ディスク化)
- RAM ディスク:300Mbytes

ほぼ同性能の CPU(Core2Duo) を使用しているラップトップパソコン 2 台に対してデータ通信端末を 1 台ずつ接続した．測定した結果を保存する際に頻繁にアクセスしハードディスクが故障するという事があったため，測定時にはメモリを一部 RAM ディスク化し，そこに保存した．測定には wget コマンドを用いて，帯域が十分であると考えられる回線に繋がっている WEB サーバから 3Mbytes のファイルをダウンロードする．これを山手線に乗車している間に繰り返す．帯域は 1 ファイルをダウンロードするのにかかった時間から計算される. 測定した帯域はその時間における平均帯域であるため，図 6.1 では時間軸方向に拡大している (また，端末 2 の測定中に時刻 16:35 にて RAM ディスクの書き込みが失敗し 5 分ほど測定が行われなかった)．端末 1 は最大 8.8Mbps，最小 0.5Mbps であり (図 6.1(a))，端末 2 は最大 4.1Mbps，最小 0.1Mbps となった (図 6.1(b))．図 6.1 を 0.5Mbps ごとに区切り，累積分布関数として表したものが図 6.2 である．電車に 1 時間乗車した際の帯域の平均値は端末 1(WiMAX) では 3Mbps，端末 2(HSPDA) では 1.5Mbps となった．



(a) WiMAX における帯域　　(b) HSPDA における帯域

図 6.1　電車内における無線網の帯域

遅延とパケットロス率の測定

往復遅延とパケットロスの測定は下記条件下で行なった.

- 測定場所
  - 電車内 (山手線)
  - 建物内
- 測定に用いた機材
  - L05A HSPA/W-CDMA/GSM

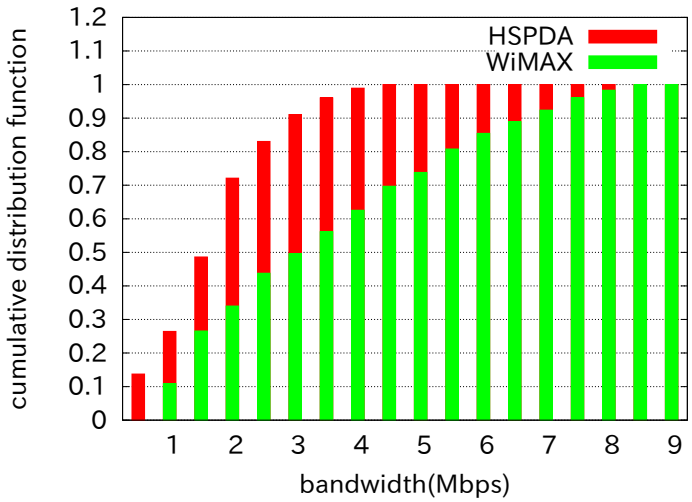


図 6.2　測定した帯域に対する累積密度関数

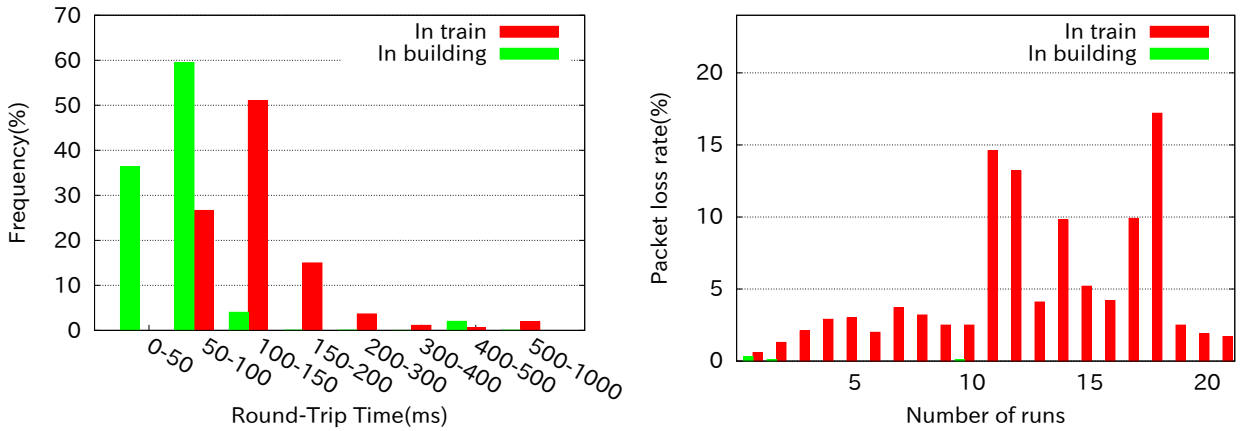
- 測定に使用したパソコン
  - OS:Windows7
  - メモリ：2Gbytes(うち 300Mbytes を RAM ディスク化)
  - RAM ディスク:300Mbytes

電車内と建物内で同一のラップトップパソコン 1 台を用いて測定を行なった．建物内での計測ではパソコンを移動していない．測定には fping というソフトウェアを用いた．fping は ping コマンドよりも ping パケットの送信間隔をより短くすることが可能なソフトウェアである．送信間隔は 1ms に設定しており，ping リプライが返信された時に即座に次の ping パケットを送信する．送信先は tracert により求めた最近隣ノードに対して行う．これは少なくともどの程度の遅延が発生するか測定するためである．事前検証により最近隣ノードは同一であったため，測定中も変化していないと想定している．ペイロードサイズはデフォルトの 32bytes に設定し，タイムアウトもデフォルトの 1 秒に設定した．往復遅延の値をヒストグラムで表したものが図 6.3(a) であり，ping パケットの送信回数を 1000 回に区切りパケットロスの割合を示したものが図 6.3(b) である．

図 6.3(a) を見ると，ヒストグラムのピークが屋内環境に比べて遅延が大きい方へ移動している．しかしながら，遅延が 200ms 以上のパケットは全体の 7% であった．タイムアウトの時間を 1 秒に設定したこともあり，10% 以上のパケットロスが発生する区間が存在した．しかしながらほとんどの区間ではパケットロス率は 5% 以下であった．

6.3　同一のアプリケーションにおける既存研究との比較

図 5.1 のアプリケーションを用いて評価を行う．評価は，帯域を集約しない手法 (シングルパス通信と表記する) と既存研究である MPTCP を用いて集約した手法 (MPTCP と省略する) と本論文で提案した手法の比較を行う．シングルパス通信ではクライアントとプロキシサーバ間において複数回線が存在する



(a) 往復遅延のヒストグラム　　(b) パケットロス率

図 6.3　遅延とパケットロス率

場合に帯域の広い回線のみを使用して通信するものとする．帯域が同じ回線が複数ある場合はそのうちの一つを使用するものとする．既存手法として MPTCP を選択した．なぜなら MPTCP は TCP を基にしているためアプリケーションの改変が必要なく，比較的新しい Linux カーネル (3.5.0) にて動作が確認されているためである．提案手法以外の測定ではプロキシプログラムは実装を行う前のプロキシとして動作するアプリケーションを使用する．提案手法において分割数は 20 で固定とする．分割数に関する考察は第 7 章にて行う．WEB サーバにはサイズの異なる三つのファイル (50KB,500KB,5MB) を用意し，これらのファイルをダウンロードする時間を測定する．ブラウザとして wget コマンドを使用した．wget コマンドを開始した時間と終了した時間を記録し，その差分をダウンロード時間として扱う．そのためダウンロード時間には HTTP のリクエストやレスポンスを送受信する時間も加算される．ダウンロードは一つのファイルに対して 40 回行う．

同一アプリケーションを用いた評価にあたり，各回線を下記のようにモデル化を行う．3G 回線は測定から得た値を基に設定した．また，品質の悪い 3G 回線は帯域を最大帯域が狭い契約を想定し，また往復遅延とパケットロス率に関しては 3G 回線の 2 倍質が悪いものとしている．

- 有線： 帯域 (50Mbps)，往復遅延 (10ms)，パケットロス率 (0%)
- 無線 LAN: 帯域 (20Mbps)，往復遅延 (20ms)，パケットロス率 (0%)
- 3G 回線: 帯域 (3Mbps)，往復遅延 (100ms)，パケットロス率 (2%)
- 品質の悪い 3G 回線: 帯域 (100Kbps)，往復遅延 (200ms)，パケットロス率 (4%)

評価は図 6.4 のネットワークを用いる．ルータを含む全てのノードは Ubuntu12.04 がインストールされており，そのうちの 1 台を sysctl コマンドにより PC ルータ (ルータと呼ぶ) として扱う．ノード間は 100BASE-TX 対応の USB-NIC を用いて接続し，ネットワークの制御を行う tc コマンドを用いて回線の品質を変化させる．tc コマンドは上り回線に対して制限をかけるソフトウェアであるので帯域は上下に対して同品質となるように両端の NIC に設定し，往復遅延とパケットロス率に関しては一方のノードで半分の制限がかかるよう設定を行う．つまり，往復遅延が 200ms の場合は両端の NIC にそれぞれ 100ms

ずつの遅延を挿入する．ルータとクライアント間は最大で三つの回線で接続され，ルータとプロキシサーバ間，ルータと WEB サーバ間はそれぞれ 1 回線で接続される．五つの回線は全て異なるセグメントとなっている．図 5.1 の構成ではなく図 6.4 のようにルータを使用した理由は，MPTCP ではノードが持つ IP アドレス全てに対してセッションを張るためクライアントとプロキシサーバがそれぞれ二つずつ IP アドレスを持つ場合に 2×2 の計四つのコネクションが張られてしまいコネクション数の増加による影響が出てしまうためである．シングルパス通信や提案手法でもコネクション数の増加によって帯域の向上が見込めるが，本章ではコネクション数ではなく同一環境下におけるプロトコルによる比較を行うためコネクション数は必要数だけ張るようなネットワーク構成を選択した．ルータとプロキシサーバ間，ルータと WEB サーバ間は有線を想定した帯域制限を行い，クライアントとルータ間の 3 回線は無線を想定した帯域制限を行う．ここではまず 2 回線の評価を行う．

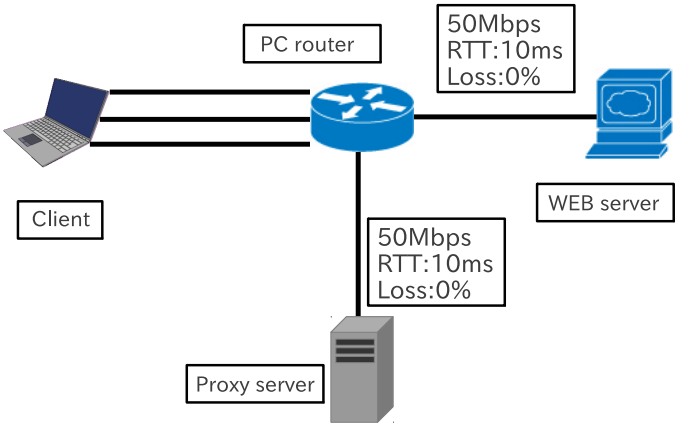


図 6.4　評価を行うネットワーク構成

無線 LAN と 3G 回線の 2 回線を集約することを (無線 LAN,3G 回線) と表記する．ホモジニアスな環境として (無線 LAN, 無線 LAN)，(3G 回線,3G 回線)，(品質の悪い 3G 回線, 品質の悪い 3G 回線) にて測定を行い，ヘテロジニアスな環境として (無線 LAN,3G 回線)，(無線 LAN, 品質の悪い 3G 回線)，(3G 回線, 品質の悪い 3G 回線) にて測定を行う．また (無線 LAN, 無線 LAN) の環境下で 5MB のファイルをダウンロードすることを (無線 LAN, 無線 LAN,5MB) と表記する．

ホモジニアスな環境における測定結果は図 6.5 である．図中に記述されている数字はダウンロードにかかった平均時間を表しており，上から順にシングルパス通信，MPTCP，提案手法となっている．date コマンドにてナノ秒まで取得しているが，表記する際に 3 桁となるように四捨五入を行なっている．ダウンロード時間が 10 秒や 100 秒を超す場合の 0.01 秒や 0.1 秒は QoE(Quality of experience) には影響しないと考えている．棒グラフはシングルパス通信を 100 とした時の値を示している．MPTCP において最もダウンロード時間の短縮が行えたのは (品質の悪い 3G 回線, 品質の悪い 3G 回線,5MB) でありダウンロード時間は 55% となった．最も効率が悪い場合は (無線 LAN, 無線 LAN,50KB) でありダウンロード時間は 84% となった．効率が悪い場合であってもシングルパス通信に比べるとダウンロード時間は短縮

されている．同様に提案手法では (品質の悪い 3G 回線, 品質の悪い 3G 回線,5MB) の場合に 51%，(無線 LAN, 無線 LAN,50KB) の場合に 95% であった．

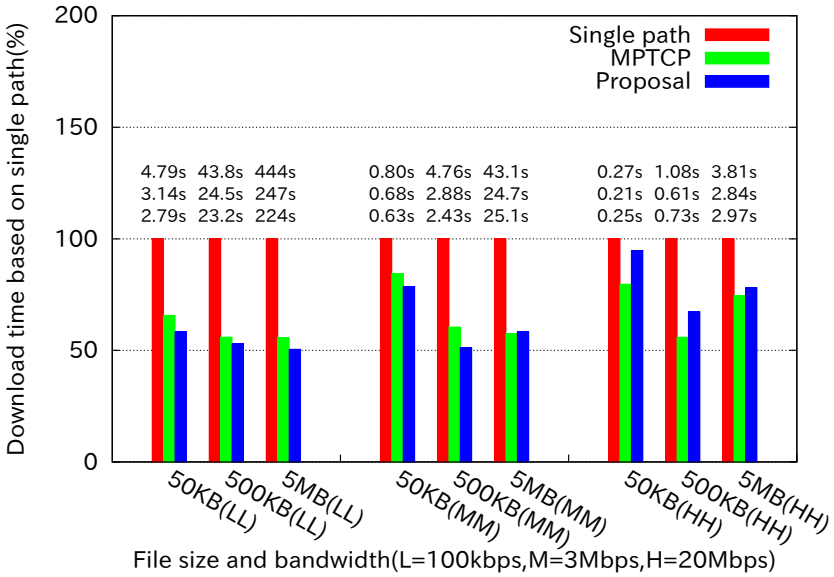


図 6.5　ホモジニアスな環境における測定 (同一アプリケーション)

ヘテロジニアスな環境における測定は図 6.6 である．MPTCP の効率の良い場合は (無線 LAN,3G 回線,500KB) でありダウンロード時間は 64%，効率の悪い場合は (3G 回線, 品質の悪い 3G 回線,50KB) でありダウンロード時間は 202% となった．提案手法の効率の良い場合は (無線 LAN, 品質の悪い 3G 回線,500KB) でありダウンロード時間は 82%，効率の悪い場合は (3G 回線, 品質の悪い 3G 回線,50KB) でありダウンロード時間は 194% となった．

6.4　異なるアプリケーションにおける既存研究との比較

HTTP プロキシを使用した場合，第 5.5 で述べたようオーバーヘッドが発生する．第 6.3 章では同一のアプリケーションを想定して測定を行なったが，シングルパス通信の場合はプロキシサーバを通す必要は必ずしもない．そこで本章では実際に使用する環境を想定したアプリケーションによって測定を行う．

まず，シングルパス通信ではプロキシサーバを通さず WEB サーバから直接ダウンロードすることにする．次に，MPTCP では HTTP プロキシではなく SOCKS プロキシを使用することにする．なぜなら MPTCP に対応しているサーバが少ない場合はマルチパス通信を行うプロキシサーバが必要になり，また HTTP プロキシではなく SOCKS プロキシを用いることで WEB サーバからプロキシサーバへのダウンロード時間が削減されるためである．SOCKS プロキシを行うソフトウェアとして OpenSSH を使用し，wget コマンドが SOCKS に対応していなかったため SOCKS プロキシを使用する測定のみ curl コマンドを使用した．提案手法においても第 5.5 章で述べた改良を行なったプログラムを使用する．分割数は 20 で固定とする．

ホモジニアスな環境における測定結果は図 6.7 である．MPTCP の効率の良い場合は (3G 回線,3G 回

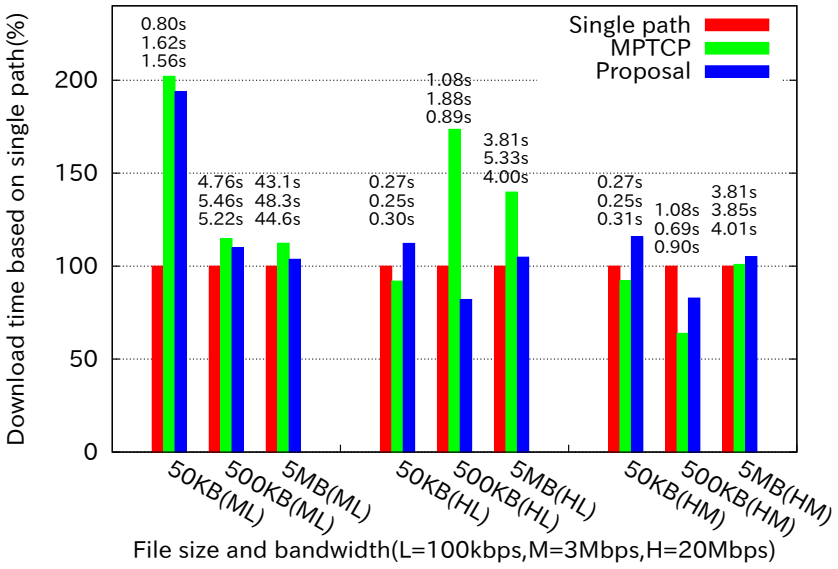


図 6.6　ヘテロジニアスな環境における測定 (同一アプリケーション)

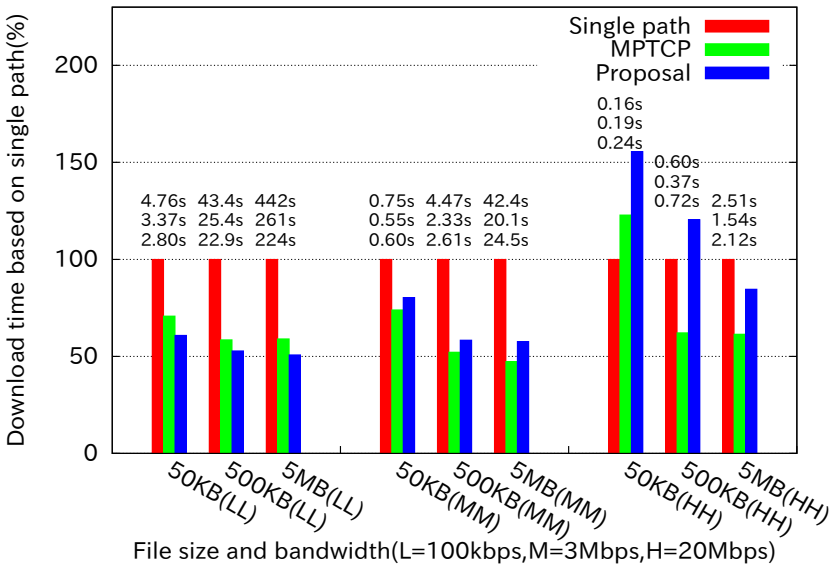


図 6.7　ホモジニアスな環境における測定 (異なるアプリケーション)

線,5MB) でありダウンロード時間は 47%，効率の悪い場合は (無線 LAN, 無線 LAN,50KB) でありダウンロード時間は 123% となった．提案手法の効率の良い場合は (品質の悪い 3G 回線, 品質の悪い 3G 回線,5MB) でありダウンロード時間は 51%，効率の悪い場合は (無線 LAN, 無線 LAN,50KB) でありダウンロード時間は 156% となった．ダウンロード時間が理想値であるとされる 50% を下回る理由は第 7 章にて考察する．

ヘテロジニアスな環境における測定は図 6.8 である．MPTCP の効率の良い場合は (無線 LAN,3G 回線,500KB) でありダウンロード時間は 80%，効率の悪い場合は (無線 LAN, 品質の悪い 3G 回線,5MB)

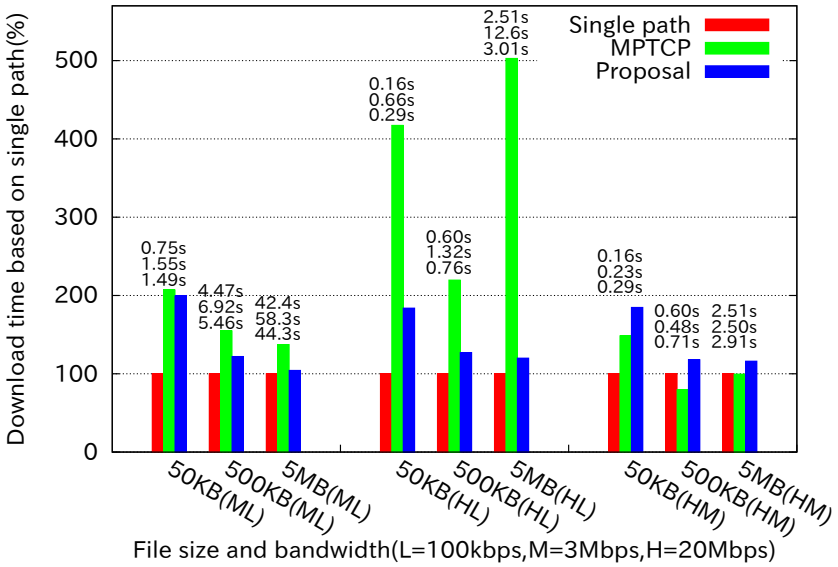


図 6.8　ヘテロジニアスな環境における測定 (異なるアプリケーション)

でありダウンロード時間は 503% となった．無線 LAN だけで直接ダウンロードした時と比べて 10 秒以上遅くなっている．提案手法の効率の良い場合は (3G 回線, 品質の悪い 3G 回線,5MB) でありダウンロード時間は 104%，効率の悪い場合は (3G 回線, 品質の悪い 3G 回線,50KB) でありダウンロード時間は 199% となった．提案手法ではいずれの環境であってもダウンロードの時間が増加する結果となった．

6.5　3 回線を集約した場合の比較

同一アプリケーションおよび異なるアプリケーションという条件下で 3 回線を集約した測定結果が図 6.9 である．図において左側が同一のアプリケーションを用いた場合で右側が異なるアプリケーションを用いた場合である．MPTCP を SOCKS プロキシを通して使用した場合，5MB のファイルのダウンロードに 11.7 秒かかっており，無線 LAN だけで直接ダウンロードした場合に比べて 9 秒以上遅くなっている．提案手法においては，同一アプリケーションにおいて 0.1 秒ダウンロード時間が伸び，異なるアプリケーションでは 0.5 秒時間が伸びた．

6.6　実機による測定

スマートフォンの多くは一般的なパソコンに比べてハードウェアの性能が低い．CPU の駆動周波数や主記憶装置に制限がある環境下で実装を行なったプログラムが想定通りの動作を行うか検証する．また，使用する回線においても帯域などを模したネットワークではなく，実際の無線回線を使用することで検証を行う．図 6.10 のようにクライアントを接続して測定を行った．モバイル端末は 3G(3.5G) と Wi-Fi の無線モジュールを内蔵し，同時に二つの回線に接続が可能であるため，これを集約する．クライアントはインターネットを通してプロキシサーバや WEB サーバに接続を行う．測定は第 6.4 章と同様に行う．測

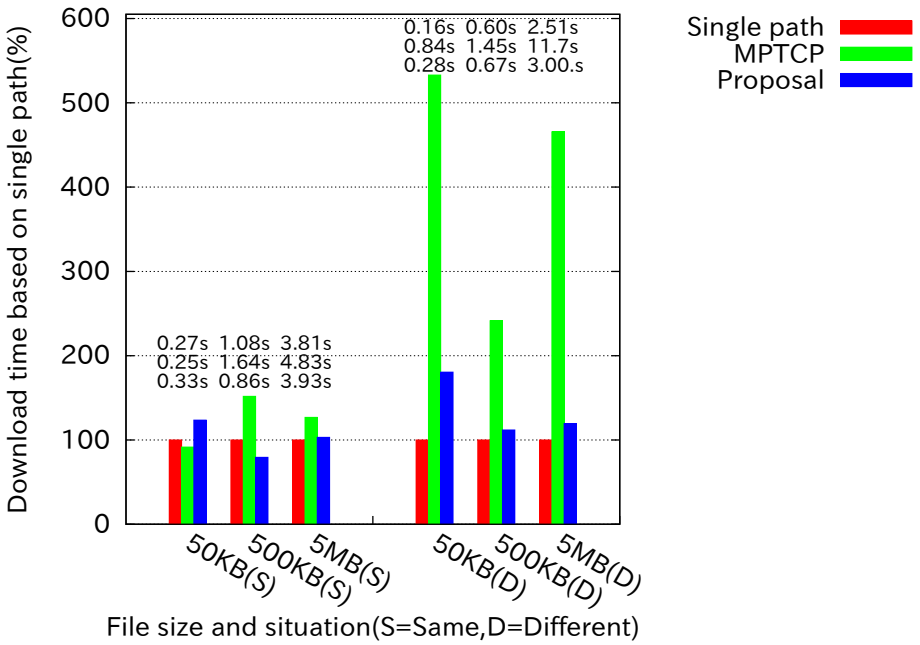


図 6.9　3 回線の帯域集約 (同一アプリケーションおよび異なるアプリケーション)

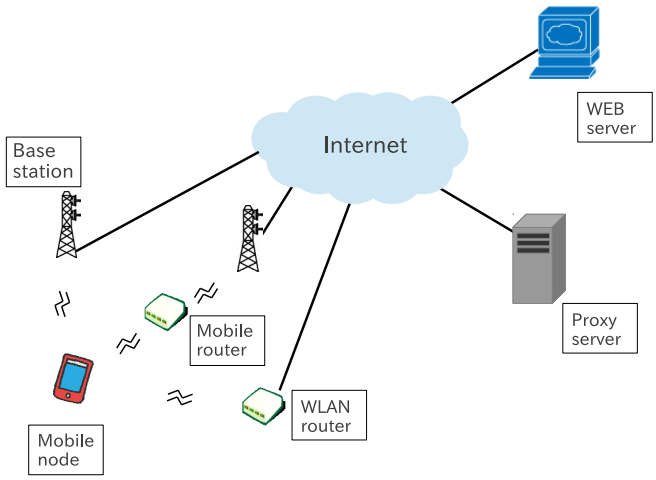


図 6.10　実機による測定の概要

定環境は以下の通りである.

- モバイル端末
  - L-07C
- 無線 LAN を構築するルータ
  - FON2405E
  - D22HW

– L-04D

ルータの WAN 側は，FON2405E では有線，D22HW では HSPA の無線，L-04D では LTE または HSPA の無線が接続されている．L-04D では上下の帯域が最大 128Kbps に制限される SIM(Subscriber Identity Module) カードを使用した．L-07C は HSPA 対応の無線モジュールを内蔵し，HSPA の無線で接続される．通信量に応じて L-04D と同様の帯域制限がされる SIM カードを使用した．帯域制限は測定中に意図せず発生しないよう通信量を確認し，帯域が制限された環境での測定の際にはあらかじめ通信量を意図的に増加させることで切り替えを行なった．測定中における帯域制限は発生しないよう管理しているが，電波環境によって帯域は変動する．その変動の特性を不変とするため，測定場所は同一とし測定は連続して行う．測定を連続して行うとは，シングルパス通信として Wi-Fi による通信を行い，次に 3G 回線による通信を行い，マルチパス通信として Wi-Fi と 3G 回線による通信を行う際に時間的間隔が極力発生しないよう行うということを指す．ダウンロードは一つのファイルに対して 10 回行う．ファイルサイズが大きく通信帯域が小さい場合にダウンロード時間が長くなるため，その場合において測定回数を 5 回に少なくして測定を行なった．パソコンを用いた測定ではソケット API である $recv()$ 関数にて 8192 バイトずつ読み込みを行なっていたが，事前動作検証においてモバイル端末ではこの部分がボトルネックとなり帯域が低下する現象が発生したため今回の測定では 819200 バイトずつ読み込みを行うよう変更を加えた．

通信品質である帯域 · 往復遅延 · パケットロス率という情報がダウンロード時間に反映されているが，参考のため往復遅延およびパケットロス率に関して以下に概算を示す．WEB サーバに ping パケットを 100 回送信した所，FON2405E では往復遅延が 14ms，パケットロス率が 0%，D25HW では往復遅延が 97ms，パケットロス率が 1%，L-04D と L-07C では往復遅延が 175ms，パケットロス率が 0% であった．

モバイル端末に 3G 回線の SIM カードを使用し，Wi-Fi とマルチパス通信を行なった際のダウンロード時間は表 6.1 であり，モバイル端末の使用する 3G 回線に帯域制限が掛けられた環境で Wi-Fi とマルチパス通信を行なった際のダウンロード時間は表 6.2 と表 6.3 である．Wi-Fi として FON2405E を使用したものが表 6.1(a) と表 6.2(a) であり，D22HW を使用したものが表 6.1(b) と表 6.2(b) である．表 6.3 は Wi-Fi に L-04D を使用したものである．Wi-Fi の帯域は FON2405E， D22HW， L-04D の順に大きいため，表中ではそれぞれを H，M，L として表記している．また 3G 回線に関しても帯域制限のかかっていないものを M とし，帯域制限がかかっているものを L と表記している．本文においても同様の表記を用いる．D22HW と L-07C で使用しているキャリアは異なるが，ここでは同じ M として表記している．L-04D と L-07C で使用しているキャリアは同一のものである．表中にてダウンロード時間に対してアンダーバーが引かれているものは測定回数を 5 回に減らしたことを表している．

表 6.1(a) と表 6.2(a) によると，50KB と 500KB のダウンロード時間は，Wi-Fi(H) のみを用いて直接 WEB サーバからファイルをダウンロードした場合よりも 0.1 秒遅くなっている．その他にシングルパス通信よりダウンロード時間が増加したものは，表 6.2(a) における 5MB のファイルをダウンロードした場合で 1.4 秒遅くなり，表 6.2(b) における 50KB のファイルをダウンロードした場合で 0.5 秒遅くなっている．

表 6.1　3G 回線と Wi-Fi のマルチパス通信におけるダウンロード時間 (秒)

	50KB	500KB	5MB
Wi-Fi(H)	0.05	0.43	4.43
3G(M)	0.60	2.02	25.47
Wi-Fi(H) + 3G(M)	0.18	0.56	4.00

(a) 3G 回線と通信帯域の大きい Wi-Fi

	50KB	500KB	5MB
Wi-Fi(M)	1.17	3.30	26.20
3G(M)	0.86	2.73	28.12
Wi-Fi(M) + 3G(M)	0.64	2.05	18.14

(b) 3G 回線と通信帯域が 3G 回線程度の Wi-Fi

表 6.2　通信帯域の小さい 3G 回線と Wi-Fi のマルチパス通信におけるダウンロード時間 (秒)

	50KB	500KB	5MB
Wi-Fi(H)	0.05	0.43	4.44
3G(L)	2.64	26.70	273.04
Wi-Fi(H) + 3G(L)	0.13	0.52	5.80

(a) 通信帯域の小さい 3G 回線と通信帯域の大きいWi-Fi

	50KB	500KB	5MB
Wi-Fi(M)	0.41	8.29	15.81
3G(L)	2.68	26.62	272.29
Wi-Fi(M) + 3G(L)	0.93	8.09	15.22

(b) 通信帯域の小さい 3G 回線と通信帯域が 3G 回線程度の Wi-Fi

表 6.3　通信帯域の小さい 3G 回線と通信帯域の小さい Wi-Fi のマルチパス通信におけるダウンロード時間 (秒)

	50KB	500KB	5MB
Wi-Fi(L)	2.67	26.72	272.95
3G(L)	2.73	28.00	271.84
Wi-Fi(L) + 3G(L)	1.63	13.76	141.95

6.7　おわりに

本章では，マルチパス通信の性能評価にあたり，まずは無線回線の通信品質を測定した．測定した値に基づいて仮想ネットワークを構築し，シングルパス通信と既存研究である MPTCP と提案手法を実装したアプリケーションを用いて，ファイルのダウンロード時間の測定を行なった．ホモジニアスな環境において帯域は集約できており，ヘテロジニアスな環境では一部帯域が小さくなった．また，実際のスマートフォンと実際の無線回線を用いた測定を行ない，仮想ネットワークと同様にホモジニアスな環境では帯域集約が行えており，ヘテロジニアスな環境では一部帯域が小さくなった．

第 7 章

考察

7.1 仮想ネットワーク上の測定に対する考察

通信帯域が大きい場合や小さなファイルをダウンロードする場合，図 7.1 で示すようなスロースタートの影響により帯域が大きくならない．スロースタートによる影響と帯域集約を行うために発生した提案手法における二つのオーバーヘッド (プログラムの処理とソケットを閉じる操作にかかる時間) による影響について考察を行う．本論文で述べるオーバーヘッドとはダウンロード時間に対するコストを指す．

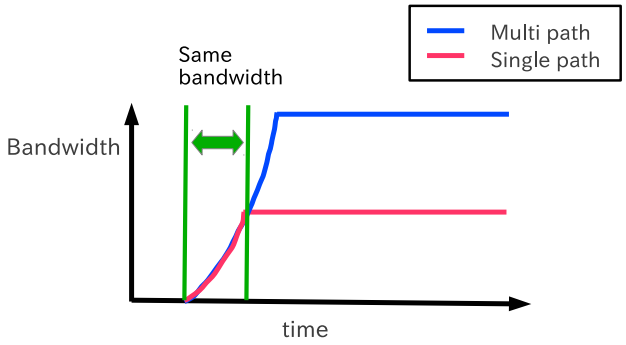


図 7.1 スロースタートの問題

7.1.1 シングルパス通信との比較

シングルパス通信よりダウンロード時間が伸びた環境に着目すると，まずヘテロジニアスな環境が主となっている．帯域のオーダーが異なる回線を組み合わせているため帯域集約のメリットは少ない．提案手法を用いた場合にファイルを 20 分割しているため，品質の悪い 3G 回線は分割されたデータを一つ転送している間に他方の回線が全てのデータをダウンロードしてしまう．アプリケーションが同一の場合，品質の悪い回線を組み合わせた場合を除くとダウンロード時間は 0.2 秒の増加にとどまった．また，異なるアプリケーションの場合は (無線 LAN, 品質の悪い 3G 回線,5MB) や (無線 LAN,3G 回線,5MB) におい

て 0.6 から 0.7 秒ダウンロード時間が増加した．図 6.6 と比較すると WEB サーバからプロキシサーバへのダウンロード時間を削減できているが，シングルパス通信で直接ダウンロードする場合と比較するとダウンロード時間が伸びてしまった．同様にホモジニアスな環境である (無線 LAN, 無線 LAN) の場合に 0.1 秒ダウンロード時間が増加した．提案手法では再送処理が分割されたデータごととなっており，帯域が 1 桁異なる回線を組み合わせた場合は通信品質の良い回線と同一の帯域となり，マルチパス通信を行うための処理にかかる時間が加算されるためダウンロード時間は増加する．加算された時間が何に起因しているのか後述する．

ダウンロード時間が減少した環境に着目すると，ホモジニアスな環境ではダウンロード時間が 51% とほぼ理論値となっている．プロキシサーバを使わないシングルパス通信と比較した場合でも 51% となった．これより，ホモジニアスな環境で，且つ帯域に対してファイルサイズが大きい場合は帯域が約 2 倍に集約できたといえる．

7.1.2　MPTCP との比較

まず同一アプリケーションを作成した場合の比較を行う（図 6.5，図 6.6）．MPTCP は [25] にて述べられているようにヘテロジニアスな環境では帯域の低下がみられた．しかし一方の回線が無線 LAN でかつファイルサイズが小さい場合はダウンロード時間が削減されている．これは，他方の回線を使用する前にダウンロードが完了したのではないかと推測している．一方で，使用している通信帯域が小さい場合は，ほとんどの場合において提案手法の性能が良い．提案手法の方がダウンロード時間が長い場合であっても，その差は最大で (無線 LAN,3G 回線,500KB) の 0.21 秒である．そのため，アプリケーションにおいてマルチパス通信を行なった場合に MPTCP と同程度の性能が出せると言える．

次に，異なるアプリケーションにて比較を行う.MPTCP では (3G 回線,3G 回線) においてダウンロード時間が半分以下になった．これは SOCKS プロキシとして用いた OpenSSH の影響である．OpenSSH ではクライアントとプロキシサーバ間では ssh 用の一つの TCP コネクションが張られ，その TCP コネクションの中で新たに仮想的な TCP コネクションを張ることで WEB アクセスを行なっている．そのためスロースタートの影響が削減されている．しかしながら，無線 LAN と品質の悪い 3G 回線を組み合わせた場合や 3 回線を組み合わせた場合は無線 LAN だけの場合に比べて 10 秒のオーバーヘッドがかかっている．この現象は同一アプリケーションの際には発生していないことから，プロキシサーバとクライアント間の通信品質の低下がプロキシサーバと WEB サーバ間の通信品質を低下させているのではないかと推測する．品質の良い回線を組み合わせた場合は MPTCP の方が帯域集約を実現しているが，一方で通信品質の悪い回線が混在している場合は提案手法の方が帯域の集約が行えている．

7.1.3　ダウンロード時間が増加した要因

プログラムの処理にかかる時間はソケットを閉じるためにかかる時間に比べると大きくないと推測した．図 6.5 において (無線 LAN, 無線 LAN,50KB) の場合にダウンロード時間は 0.25 秒であり，発生したオーバーヘッドも 0.25 秒以下である．しかし，図 6.6 において (3G 回線, 品質の悪い 3G 回線,5MB) の場合にダウンロード時間が 1.5 秒増加している．プログラムの処理にかかる時間がほぼ同一であると仮

定すると，このオーバーヘッドはソケットを閉じるためにかかる時間に起因すると考えられる．しかしながら，8KB の転送に品質の悪い回線を使用したとしても 1.5 秒はかからない．往復遅延が 200ms であるため，パケットロスが発生したと仮定しても 0.4 秒または 0.6 秒であると推測できる．そこでサーバから送信が完了した時点からのクライアントの動作を見てみると，一方の回線で通信が完了後も他方の回線でデータを受信していた．つまり，サーバ側では分割したデータの送信が完了していても，受信側では受信が完了していなかった．これは送信者側において送信完了のフラグをソケット API である *send*() が完了した時点で立てていたことに起因する．分割したデータが送信バッファ内に収まった場合は送信が完了したと見なして再送処理が行われていない．そのため図 6.6 における (3G 回線, 品質の悪い 3G 回線,5MB) のダウンロード時間 44.6 秒は，通信品質の悪い 3G 回線で 5MB のデータを 20 分割したもののうち二つをダウンロードするためにかかる時間に近い値となる (図 6.5 において通信品質の悪い 3G 回線にて 5MB のファイルをダウンロードするためにかかる時間は 444 秒であり，20 分割したデータのうち二つのデータをダウンロードするためにかかる時間は 44.4 秒)．また，図 6.6 において (3G 回線, 品質の悪い 3G 回線,50KB) の場合も 0.8 秒から 1.56 秒にダウンロード時間が増加したが，品質の悪い 3G 回線で転送したデータにおいて送信完了フラグを立てずに他方の回線で再送を行わせた所，ダウンロード時間は 0.9 秒とダウンロード時間は減少した．シングルパス通信に比べて増加した 0.1 秒がプログラムの処理にかかった時間である．

クライアント側で受信できていないデータをサーバ側と同期するには 2 通りの手法が挙げられる．送信バッファのサイズを小さくすることでソケットの *send*() と送信バッファの状態とを同一となるようにする手法と，クライアント側でデータを受信した際にアプリケーション層で ACK を送信する手法である．いずれの手法もそれぞれ問題点が存在する．前者の問題点は通信帯域が大きい場合に帯域を十分に活用することが困難になるということである．後者の問題点は，無線回線では遅延が大きいため ACK が遅れることによって不必要な再送が増えてしまうことである．送信バッファや MSS(Maximum Segment Size) に比べて分割されたデータが小さい場合に未受信のデータのみを再送することが困難となる．今回行なった測定のようにダウンロード時間に対してパケットの往復時間が影響を及ぼす環境では，サーバとクライアント間でパケットのやり取りが少ない同期を行う必要がある．

その他にダウンロード時間を増加させる要因としてプロキシを経由することが挙げられる．今回の実装では，プロキシサーバに送信可能なデータが溜まるまで待たなければならない．これを測定した所 5MB のファイルで 0.2 秒であったため，図 6.8 における (無線 LAN, 品質の悪い 3G 回線,5MB) と (無線 LAN,3G 回線,5MB) で発生した 0.5 秒のオーバーヘッドのうち 0.2 秒がプロキシによるオーバーヘッドであると言える．このオーバーヘッドを削減するためにプロキシサーバ上でより柔軟な転送を実装する必要がある．

7.2　実機を用いた測定に対する考察

モバイル端末にスマートフォンを使用し，実際の 3G 回線と Wi-Fi を組み合わせた場合でも仮想ネットワークを使用した場合とほぼ同様の結果となった．表 6.1(a) と表 6.2(a) では，50KB と 500KB のファイルを直接 WEB サーバからダウンロードする際に Wi-Fi のみで接続する場合と比べて 0.1 秒オーバー

ヘッドが発生した．また，表 6.2(b) において，50KB のファイルに対するダウンロードが 0.5 秒増加したが，パソコンを用いた場合と同様に 3G 回線に割り当てているデータを再送できていないことに起因する．

しかしながら，パソコンを使用した測定では発生しなかった現象も発生した．CPU 負荷が高い場合にダウンロード時間が増加するというもので，5MB のファイルを Wi-Fi(H) と 3G(L) でダウンロードした場合 (表 6.2(a)) に，図 7.2 のように 10 回の測定中にダウンロード時間が増加した．また表 6.3 において，5MB のファイルのダウンロード時間はシングルパス通信の半分である 136 秒に比べて 141.95 秒と 6 秒のオーバーヘッドが発生している．実装を行なったプログラムにて一方の回線のみ使用する測定を再度行なった所，278.59 秒と，プログラムを使用したことによるダウンロード時間の増加が 6 秒であったため，マルチパス通信を行なった際に発生したオーバーヘッドはこれに起因していると推測している．スマートフォンで使用する場合はプログラムの軽量化を行う必要がある．

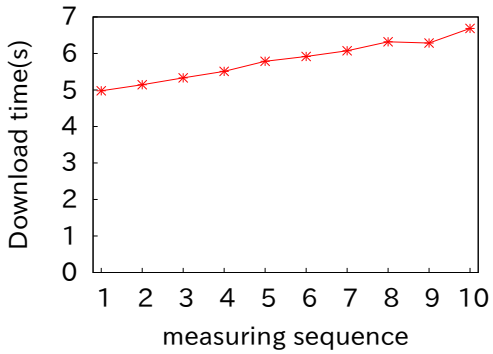


図 7.2　Wi-Fi(H) と 3G(L) のマルチパス通信下で
5MB のファイルをダウンロードするためにかかった時間

7.3　議論と今後の課題

既存研究ではヘテロジニアスな環境においてマルチパス通信を行うことにより通信帯域が小さくなる場合があった．そこで，通信帯域が大きく異なる場合においても複数回線のうち最も良い回線の帯域が保証される手法を提案し評価を行なった．提案手法では事前に帯域を測定する必要がないため，ダウンロード時間が短い場合であってもマルチパス通信が可能である．通信帯域が等しい回線を組み合わせた場合はトランスポート層によるマルチパス通信と同様に帯域の集約が実現された．ヘテロジニアスな環境下では，再送処理が行われた場合に通信品質の良い回線で通信を行い，加算されたダウンロード時間は 0.1 秒であった．しかしながら，サーバとクライアント間での送受信が同期されておらず，送信バッファ内にあるデータに関しては再送が行えていなかった．サーバとクライアントの同期を行う必要があるが，パケットの往復時間が影響を及ぼすため，パケットのやり取りの少ない同期手法の検討が今後の課題となる．また，今回作成したアプリケーションではプロキシサーバにファイルをダウンロードするためにかかる時間が加算され，そして，スマートフォンによる動作時において CPU 負荷が高い場合にダウンロード時間が伸びてしまった．そこで，本アプリケーションの配布にあたりプログラムの最適化を行う必要がある．

実装を行なったプログラムを使用することで携帯向けの WEB サイトへのアクセスを高速化できないか検討している．評価の際に使用したアプリケーションはファイルを 1 つずつダウンロードするものであり，ファイルサイズも小さい場合で 50KB であった．いくつかの WEB サイトの調査を行なった所，ファイルのダウンロードは並列的に行われ，またファイルサイズは最大で 50KB というサイトが存在する．ファイルサイズが 1KB 以下のものが多数含まれるサイトも存在した．ファイルサイズが MSS より小さい場合，そのファイルを分割し，複数の回線に割り振るメリットは少ない．実際に本プログラムを使用した所，シングルパス通信に比べて 3 倍以上時間がかかる場合があった．これは，ファイルのダウンロードが並列で行われるため CPU への負荷がより高くなったことに起因すると推測している．そして，1 つのサイトに含まれるファイル (コンテンツ) の数も 10 個以上であることが多く，今回述べたオーバーヘッドが 1 サイトあたりファイルの数だけ加算される．つまり，本プログラムを使用すると 1 秒以上のオーバーヘッドが加算されると推測できる．そのため，プログラムの軽量化だけではなく，WEB サイトへのアクセスを考慮した仕組みの導入が必要となる．

第 8 章

まとめ

本研究では，スマートフォンの持つ複数の無線回線に対して帯域集約を行う手法を提案した．スマートフォンでは OS の改変が困難であり，また標準で SCTP というプロトコルをサポートしていないため既存のマルチパス通信はあまり利用されていない．そこで，提案手法では，一般的な OS やアプリケーションに用いられている TCP というプロトコルを使用し，アプリケーション層においてマルチパス通信を実現した．また帯域の集約を行う回線の通信品質が異なる環境や，無線回線のように帯域が変動する環境下では帯域を集約したことによって帯域が元の回線より小さくなることがある．そこで，通信品質の良い回線にて積極的に再送処理を行うアルゴリズムを提案した．また，これは帯域の測定を必要としないためファイルサイズが小さくデータの転送時間が短い場合であっても使用可能である．

提案手法を実装したアプリケーションを用いて評価を行い，再送処理が動作した場合に，通信帯域が異なる環境やファイルサイズの小さい環境下においてトランスポート層における既存の手法と同様に帯域の集約が行えていることを示した．また，実際のスマートフォンにおいても動作の確認を行い，実際の無線回線を使用し帯域の集約が行えることを示した．

謝辞

本論文を執筆するに当たり，大変多くの方からご指導，ご協力を頂きました．ここに心より感謝の意を表します．まず，２年という研究生活において幅広い分野に渡る知識により時には厳しく時には優しく指導して頂いた江崎浩博士に深く感謝致します．色々な失敗を経験しましたが，その度に励まし助言して頂き，それらを次の成功に活かせるよう努めることができました．

研究に関する指導だけでなく，運用の技術を指導して頂いた山本成一博士および土本康生博士に深く感謝致します．運用の技術は単なる独学では足りておらず，実運用を通して多くを学ばせて頂きました．また，様々な開発デバイスを紹介して頂いた落合秀也博士に深く感謝致します．実際に自分で手を動かし，物作りをする楽しさを教えて頂きました．

測定実験を行う際に白井俊宏氏に大変お世話になりました．実際の現場で働いて得た知見によって助言をして頂き，また実験機器を提供して頂き，円滑に測定実験を行うことができました．気軽に相談をしても良いとおっしゃって頂き，研究の方向性に迷った際には何度も相談させて頂きました．浅井大史氏から自身の研究からの知見によって具体的で理解しやすい助言を頂き，研究を進めて行く上で大変参考になりました．両先輩に感謝致します．

英語の苦手な私でしたが Romain Fontugne 氏，Badertscher Stefan Jurg 氏，Luciano Aparicio 氏と研究の相談をし英語の力も身につきました．Trinh Minh Tri 氏は週に何度も一緒に水泳に行き，研究で行き詰まった時のリフレッシュを共に行いました．国境を超えたお付き合いに感謝致します．

江崎研究室に配属になった際に右も左も分からない私を導いて頂いた呉和賢氏，本舘拓也氏，川口紘典氏に感謝致します．発表の仕方や論文の書き方を指導して頂き参考になりました．同期として一緒に研究をし一緒に苦労を共にした石橋尚武君，正原竜太君，李聖年君，林東權君に感謝致します．研究室は研究するだけの場所ではなく娯楽を楽しむ場所ともなりました．後輩として一緒に研究やワークショップで作業したくれた高成源君，池上洋行君，川守田光昭君，美嶋勇太朗君，西田綾佑君，中村哲也君，中村遼君，木下僚君，沼田進君，小林諭君，田中晋太朗君，福田鉄平君に感謝致します．既に卒業して社会人として活躍している東浦成良君，朴成軍君に感謝致します．また，短い間でしたが色々と面白い話を話してくださった松田貴成氏に感謝致します．学生が研究に集中できるよう日々研究室の整備に尽力して頂いた江崎研究室の秘書である高橋富美さん，岩井愛映子さんに感謝致します．

私が以前に所属していた森川研究室の皆さんにも大変お世話になりました．森川研究室の企画に卒業生として参加させて頂き，充実した学生生活を送ることができました．奥井寛樹君，田代諭拡君，下城拓也君，David Gonzalez 君とはそれぞれの研究の議論をしました．彼らの頑張る姿を見ることで私の研究に対するモチベーションが向上しました．共に研究してくれたことを感謝致します．また，森川研究室に所

属していた時の同期である大和田裕亮君，李睿智君も既に研究分野が変わってしまいましたが，それでも色々と議論をしました．分野が違う話が聞けて僕の知見も広がったように感じます．両氏に感謝致します．

最後に，私の学生生活を支え続けてくれた家族と友人にこの場をお借りして御礼申し上げます．

2013 年 2 月 6 日

小坂 良太

参考文献

[1] H.S. Abdel-Ghaffar, H.A.E.A. Ebrahim, and A.A.M. Khalifa. Performance evaluation of real time applications for vertical handover between wlan 802.11 g and umts. In *Proceedings of the 3rd international conference on Mobile technology, applications & systems*, p. 1. ACM, 2006.

[2] C. Ahlund, R. Brannstrom, K. Andersson, and O. Tjernstrom. Port-based multihomed mobile ipv6 for heterogeneous networks. In *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*, pp. 567–568. IEEE, 2006.

[3] K. Argyraki, P. Maniatis, and A. Singla. Verifiable network-performance measurements. In *Proceedings of the 6th International Conference*, p. 1. ACM, 2010.

[4] H. Balakrishnan, M. Stemm, S. Seshan, and R.H. Katz. Analyzing stability in wide-area network performance. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 25, pp. 2–12. ACM, 1997.

[5] S. Barré, O. Bonaventure, C. Raiciu, and M. Handley. Experimenting with multipath tcp. *ACM SIGCOMM Computer Communication Review*, Vol. 40, No. 4, pp. 443–444, 2010.

[6] K. Chebrolu, B. Raman, and R.R. Rao. A network layer approach to enable tcp over multiple interfaces. *Wireless Networks*, Vol. 11, No. 5, pp. 637–650, 2005.

[7] K. Chebrolu and R.R. Rao. Bandwidth aggregation for real-time applications in heterogeneous wireless networks. *Mobile Computing, IEEE Transactions on*, Vol. 5, No. 4, pp. 388–403, 2006.

[8] D. Crocker. STANDARD FOR THE FORMAT OF ARPA INTERNET TEXT MESSAGES. RFC 822 (Standard), August 1982. Obsoleted by RFC 2822, updated by RFCs 1123, 2156, 1327, 1138, 1148.

[9] A.B. Downey. Using pathchar to estimate internet link characteristics. In *ACM SIGCOMM Computer Communication Review*, Vol. 29, pp. 241–250. ACM, 1999.

[10] K. Evensen, D. Kaspar, P. Engelstad, A.F. Hansen, C. Griwodz, and P. Halvorsen. A network-layer proxy for bandwidth aggregation and reduction of ip packet reordering. In *Local Computer Networks, 2009. LCN 2009. IEEE 34th Conference on*, pp. 585–592. IEEE, 2009.

[11] K. Evensen, D. Kaspar, A.F. Hansen, C. Griwodz, and P. Halvorsen. Using multiple links to increase the performance of bandwidth-intensive udp-based applications. In *Computers and Communications (ISCC), 2011 IEEE Symposium on*, pp. 1117–1122. IEEE, 2011.

[12] F. Fitzek, A. Kopsel, A. Wolisz, M. Krishnam, and M. Reisslein. Providing application-level

qos in 3g/4g wireless systems: A comprehensive framework based on multirate cdma. *Wireless Communications, IEEE*, Vol. 9, No. 2, pp. 42–47, 2002.

[13] A. Ford, C. Raiciu, and M. Handley. TCP Extensions for Multipath Operation with Multiple Addresses. Internet draft, draft-ietf-mptcp-multiaddressed-12.txt, Work in progress, October 2012.

[14] B.D. Higgins, A. Reda, T. Alperovich, J. Flinn, T.J. Giuli, B. Noble, and D. Watson. Intentional networking: Opportunistic exploitation of mobile network diversity. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking*, pp. 73–84. ACM, 2010.

[15] H.Y. Hsieh, K.H. Kim, and R. Sivakumar. An end-to-end approach for transparent mobility across heterogeneous wireless networks. *Mobile Networks and Applications*, Vol. 9, No. 4, pp. 363–378, 2004.

[16] H.Y. Hsieh and R. Sivakumar. On transport layer support for peer-to-peer networks. *Peer-to-Peer Systems III*, pp. 44–53, 2005.

[17] J. Iyengar, K. Shah, P. Amer, and R. Stewart. Concurrent multipath transfer using sctp multihoming. *SPECTS 2004*, 2004.

[18] J.R. Iyengar, P.D. Amer, and R. Stewart. Concurrent multipath transfer using sctp multihoming over independent end-to-end paths. *Networking, IEEE/ACM Transactions on*, Vol. 14, No. 5, pp. 951–964, 2006.

[19] E. Katz-Bassett, H.V. Madhyastha, V.K. Adhikari, C. Scott, J. Sherry, P. Van Wesep, T. Anderson, and A. Krishnamurthy. Reverse traceroute. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, pp. 15–15. USENIX Association, 2010.

[20] D. Kim and A. Ganz. Architecture for 3g and 802.16 wireless networks integration with qos support. In *Quality of Service in Heterogeneous Wired/Wireless Networks, 2005. Second International Conference on*, pp. 8–pp. IEEE, 2005.

[21] J.I. Kim and S.J. Koh. Extension of proxy mobile ipv6 with bicasting for support of multihoming and mobility in wireless networks. In *Advanced Information Networking and Applications (WAINA), 2011 IEEE Workshops of International Conference on*, pp. 86–89. IEEE, 2011.

[22] J.M. Márquez-Barja, C.T. Calafate, J.C. Cano, and P. Manzoni. Evaluating the performance boundaries of wi-fi, wimax and umts using the network simulator (ns-2). In *Proceedings of the 5th ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*, pp. 25–30. ACM, 2010.

[23] T.N. Minhas, M. Fiedler, and P. Arlos. Quantification of packet delay variation through the coefficient of throughput variation. In *Proceedings of the 6th International Wireless Communications and Mobile Computing Conference*, pp. 336–340. ACM, 2010.

[24] V. Nelson, J. Tania, and H. Yezekael. Performance of sctp in wi-fi and wimax networks with

multi-homed mobiles. In *Proceedings of the 3rd International Conference on Performance Evaluation Methodologies and Tools*, p. 71. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008.

[25] S.C. Nguyen, X. Zhang, T.M.T. Nguyen, and G. Pujolle. Evaluation of throughput optimization and load sharing of multipath tcp in heterogeneous networks. In *Wireless and Optical Communications Networks (WOCN), 2011 Eighth International Conference on*, pp. 1–5. IEEE, 2011.

[26] C. Perkins. IP Mobility Support for IPv4, Revised. RFC 5944 (Proposed Standard), November 2010.

[27] C. Perkins, D. Johnson, and J. Arkko. Mobility Support in IPv6. RFC 6275 (Proposed Standard), July 2011.

[28] L. Popa, A. Ghodsi, and I. Stoica. Http as the narrow waist of the future internet. In *Proceedings of the Ninth ACM SIGCOMM Workshop on Hot Topics in Networks*, p. 6. ACM, 2010.

[29] C. Raiciu, C. Paasch, S. Barre, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M. Handley. How hard can it be? designing and implementing a deployable multipath tcp. *Networked Systems Design and Implementation (NSDI'12)*, 2012.

[30] H. Sakakibara, M. Saito, and H. Tokuda. Design and implementation of a socket-level bandwidth aggregation mechanism for wireless networks. In *Proceedings of the 2nd annual international workshop on Wireless internet*, p. 11. ACM, 2006.

[31] A. Scott. Connectify dispatch. `http://www.connectify.me/dispatch/`,visited on 2013-02-01.

[32] H. Soliman, C. Castelluccia, K. ElMalki, and L. Bellier. Hierarchical Mobile IPv6 (HMIPv6) Mobility Management. RFC 5380 (Proposed Standard), October 2008.

[33] R. Stewart. Stream Control Transmission Protocol. RFC 4960 (Proposed Standard), September 2007. Updated by RFCs 6096, 6335.

[34] O. Teyeb, T.B. Sørensen, P. Mogensen, and J. Wigard. Subjective evaluation of packet service performance in umts and heterogeneous networks. In *Proceedings of the 2nd ACM international workshop on Quality of service & security for wireless and mobile networks*, pp. 95–102. ACM, 2006.

[35] 伊藤陽介, 小山健二, 太田賢, 石原進. Mobile ip を用いた通信回線共有方式の実装. マルチメディア, 分散, 協調とモバイル (DICOMO2003) シンポジウム論文集, 情報処理学会シンポジウムシリーズ, Vol. 2003, No. 9, pp. 97–100, 2003.

[36] 荻野秀岳, 石原進. Mobile ip shake における tcp の輻輳ウィンドウ縮小回避方法 (携帯端末, モバイルアプリケーション, モバイルコンピューティング). 電子情報通信学会技術研究報告. MoMuC, モバイルマルチメディア通信, Vol. 107, No. 39, pp. 51–56, 2007.

[37] 玉井森彦, 酒井憲吾, 山本俊明, 長谷川晃朗, 植田哲郎, 小花貞夫. 多様な無線システムの同時利用を考慮した階層化モバイル ipv6 による移動通信方式の提案 (一般講演, コグニティブ無線, 招待講演, 一般). 電子情報通信学会技術研究報告. SR, ソフトウェア無線, Vol. 108, No. 400, pp. 75–82, 2009.

[38] 園部修. au Wi-Fi SPOT の環境改善を進める KDDI(ITmedia Mobile). http://www.itmedia.co.jp/mobile/articles/1211/02/news131.html,visited on 2013-02-01.

[39] 野澤高弘. アプリケーション層における複数パスの同時利用による高速通信機構. Master's thesis, 慶応義塾大学, 2010.