DEIM Forum 2011 F10-2

無線通信の帯域幅集約ミドルウェアにおける複数経路データ配分手法とバッファ量に関する評価

宮崎　悦子†　　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
E-mail: †etsuko@ogl.is.ocha.ac.jp, ††oguchi@computer.org

あらまし　無線通信で使用できる帯域は有線での通信と比較していまだ乏しいものが多いのが現状である．そこで使用可能な無線通信を複数同時に使用する帯域幅集約が提案されているがトランスポート層より下の制御で集約を実現しようとするとパケットロスが起こった際に TCP の輻輳制御により必要以上に通信の品質が下がってしまうのが問題である．そこで本研究では帯域幅集約ミドルウェアを提案し，複数経路データ配分アルゴリズムと受信側のバッファサイズについての評価と考察を行った．

キーワード　複数帯域, ミドルウェア, 帯域幅集約, バッファサイズ

Evaluation of Buffer Size and Data Distribution Method for Bandwidth Aggregation on Middleware Layer in Wireless Communication

Etsuko MIYAZAKI† and Masato OGUCHI†

† Department of Computer Science, Ochanomizu University
2-1-1 Ohtsuka, Bunkyo-ku, Tokyo, 112–8610 Japan
E-mail: †etsuko@ogl.is.ocha.ac.jp, ††oguchi@computer.org

1. はじめに

近年，モバイルインターネットは急速に発展しており，この基盤となる様々な無線通信技術が開発されている．例えば，無線 LAN に用いる IEEE802.11，近距離無線通信の Bluetooth や無線 MAN の WiMAX などである．しかしこれらの無線技術は広帯域なものであっても，Ethernet などの有線での通信と比較するとネットワークそれぞれの帯域幅が未だに乏しいものが多い．また，比較的広帯域を確保している無線は利用可能なシーンが限定されているのが現状である．そこで同時に複数の無線技術の適用範囲がオーバーラップしたとき，ユーザが 1 つの無線技術だけでなく複数の無線技術を選択して利用できれば帯域の制限が緩和され，より充実したモバイルサービスを受けることが可能になると考えられる．そこで提案されているのが適用範囲にある無線技術の帯域幅を複数同時に使用することでより大きいスループットを得ようとする帯域幅集約（Bandwidth Aggregation）である．

これまでの研究により，1 つのインタフェースから別のインタフェースへのスムーズなハンドオフを行うことについては議論されているが [1]，同時に複数の無線技術を用いて通信することについては未だに実用化されていない．複数の無線技術を同時に使用することで，帯域幅集約の他に可動性のサポート，信頼性の向上やリソース共有などのメリットを受けることができると考えられる．

2. 研究背景

2.1 各層における帯域幅集約手法

帯域幅集約は様々な層において実現することが考えられるが，それぞれに利点と欠点が存在する．

データリンク層でのアプローチ [2] では，効率よく帯域を集約することができ，それより上位の層が集約していることを意識せずに済むのが最大の利点である．しかし，同じデータリンク層のプロトコルを使用しているネットワークの範囲内でしか使用できず，すべてのノードに専用のハードウェアを導入しなければならないという欠点を持つ．

ネットワーク層での制御では，手法によって専用機器も必要なく帯域幅集約を実現することができる [3] [4]．上位層に現在広く使用されている TCP・UDP に対して透過的に振る舞うことができるのが利点であるが，トランスポート層のプロトコルとして TCP を用いて集約を行った場合，本来の順序とは異な

る順序で到着するパケットが多くなる可能性が高い．それにより TCP は不必要な再転送を要求し，全てのコネクションのスループットを下げてしまうことが欠点となっている [5]．また，この問題点を解決するためのトランスポート層のプロトコルとして RR-TCP [6] が提案されているが，パケットロスが一定より多い環境だと通常の TCP を使う場合と同等の結果しか得ることができないということが分かっている．

トランスポート層で集約を行うと輻輳制御や再転送制御を経路ごとに持たせることができる [7]．これにより各経路に応じた効率的なパケットの分配や再転送を行うことができるので効率のよい通信を実現可能であるが，エンド間でのオペレーティングシステム内の実装が不可欠である．

アプリケーション層での実装では 1 つのエンドノードのみへ集約の制御を加えることで実現できる [8]．現行のオペレーティングシステム内を書き換える必要も無いということも大きな利点である．しかし様々な種類のアプリケーションが存在する中で，それぞれに適した集約の実装を行うのは大変困難である．また経路の数だけトランスポート層プロトコルのコネクションを張ってもどのコネクションへどの程度トラフィックを分散させるかを見積もることが重要となる．

2.2 ネットワーク層における帯域幅集約

性質の異なる無線インタフェースを複数同時に用いて通信を行った場合，異なる経路からのパケットは送信された順序以外でパケットが到着する可能性がある．パケット受信側の TCP は，次に期待していたパケットより後のパケットを先に受信してしまうと，パケットが抜けたと認識してしまう．それにより TCP は本来不要であるはずの再転送を要求し，期待した集約結果を得ることができない．

そこで提案されているのが EDPF(Earliest Delivery Path First) [3] である．EDPF は送信側からパケットを送信する際，経路が分岐するノードにおいて実装される．分岐ノードは分岐する先の経路の帯域幅や遅延，使用率を元に，パケットがどの経路を通れば最短で受信側ノードへ届くかを計算する．パケットは常に最短で届くと見積もられた経路へ配信されるので受信側 TCP は常に送信側で送信された順にパケットを受け取ることができる．この手法により，パケットロスの無い環境では期待通りの帯域幅集約の性能を発揮できることが確認されている．

2.3 ネットワーク層における問題解決のアプローチ

無線環境で通信を行った場合，有線の場合と比較してパケットロスが多く起こることが知られている．ネットワーク層以下で帯域幅集約を行う際にパケットロスが起こると，TCP は特定の経路のパケットロスであることを認識できず輻輳制御を行ってしまう．その結果ウィンドウサイズが下がるのでスループットを下げる原因となり，期待した帯域幅集約の性能を得ることができない．

この問題を解決するために提案されているネットワーク層の帯域幅集約方式が PET (Packet-Pair based Earliest-Delivery-Path-First algorithm for TCP applications) と BMP (Buffer Management Policy) である [5]．PET は EDPF で行っていた経路の見積もりを，より厳格に，動的に行う．BMP はパケット受信側のノードへ実装され，受信したパケットを並び替えが必要なものか，それともパケットロスなのかを判断する役割を果たす．次に到着する筈のパケットのシークエンス番号より大きいシークエンス番号のパケットがすべての経路から到着すれば，それは確かにパケットロスであることを BMP が TCP へ報告する．そう出ない場合には，順番どおりに並び替えたパットを TCP へ渡す役割を果たす．

PET と BMP を実装した場合，EDPF を用いた場合と比較すると，パケットロスが起こった際の複数コネクションのスループットの低下を防ぐことができている．しかし，パケットロスが集中した場合，複数コネクションのスループットが下がってしまうことが問題となっている．既存研究では，この問題に対して，ほかの仕組みで無線のロスを減らすことができれば期待した性能を発揮することができるとしているが，現実には無線通信でのパケットロスを減らすことは大変困難である．

3. 提 案 手 法

3.1 概　　要

先に述べたようにトランスポート層以下やアプリケーションにおいて帯域幅集約を行う際には様々な制約や問題が存在する．そこで本研究ではトランスポート層とアプリケーション層の間にミドルウェアとして帯域を集約する層を新たに挿入することでより効率の良い帯域幅集約を行う手法を提案する．ネットワーク層での帯域幅集約と提案手法を比較したネットワーク各層の概念図を図 1 に示した．

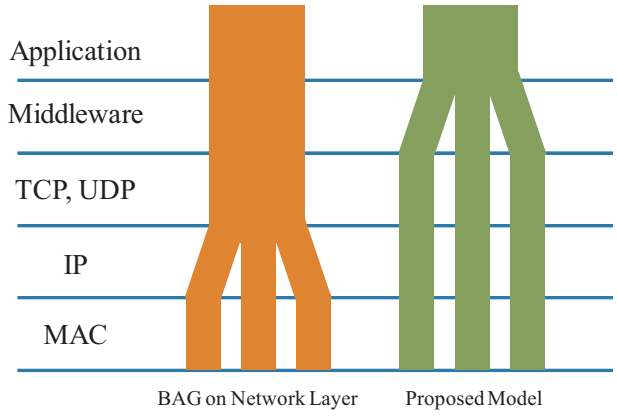

図 1　提 案 手 法

提案手法では経路ごとに独立した TCP のコネクションを持ち，それら複数のコネクションをミドルウェアで統合することで帯域を集約していることをアプリケーションが意識することなく通信可能であるというトランスポート層以下での制御でのメリットを残すことができる．さらにパケットロスが起こった場合に数経路のスループットが下がってしまう原因となっている輻輳ウィンドウを経路ごとに持たせることで，既存研究で指摘されている問題点を回避する．ネットワーク層で帯域幅集約を行った場合，上位層のプロトコルである TCP は集約された結果のデータしか受け取ることができないのでどの経路を経由

して届くはずであったパケットがパケットロスを起こしたか知覚することができない．提案手法によりすべての経路へ TCP のコネクションを確立し，輻輳制御もそれぞれの TCP コネクションが行うことでパケットロスが起こった際に発生する帯域幅集約の問題点を解決する．

この手法は，既存の TCP を作り変えて，複数のコネクションをトランスポート層で 1 本に束ねるアプローチでも実現可能である．しかしミドルウェアで統合することを考えれば既存の TCP をそのまま利用して目的を達成することが可能であると考えられる．

3.2 提案ミドルウェアの構造と動作

送信側の提案手法ミドルウェアは接続可能な経路すべてへそれぞれ TCP コネクションを張る．アプリケーションからデータを受け取るとパケットの大きさ分にデータを分割し，配信順の番号をデータへ付与してから確立されたコネクションへデータを送信していく．データを配信する経路を選択する際のアルゴリズムとして EDPF for Middleware (Earliest Delivery Path First for Middleware) を採用した．EDPF はそれぞれの経路でパケットを配信した場合，送信ノードから受信ノードまでどれだけ時間がかかるかを計算し，その見積もられた時間が一番短いと判断された経路へパケットを配信する．パケット配信時間の見積もりには経路の有線・無線部分の帯域幅，遅延と混雑状況を使用した．次の時間の見積もりを計算するために，パケットを配信するたびに配信した経路の混雑状況を更新する．

受信側のミドルウェアはそれぞれの TCP コネクションから受信されたパケットから付与されているパケット番号を読み出し，その時点でミドルウェアが期待していた番号のパケットだった場合にはそのままデータをアプリケーションへ渡す．期待した番号ではなかったらミドルウェアが持つバッファへパケットを一時貯めておき，次のパケットを待つ．それぞれの TCP コネクションから受信されるパケットは順番が正しくない可能性があるので，それを並び替えてからアプリケーションへ渡す役割を受信側ミドルウェアが果たす．

4. シミュレーションソフトウェアによる評価

本実験では帯域幅集約を行う際の明らかになっている問題の解決のために新しいモデルを考案することにより，帯域幅集約の性能を上げることを目標とする．実験で使用したのは無線ネットワークのシミュレーションソフトウェアである QualNet [9] である．

第 5 章では予備実験として提案手法のモデルを設計するために必要な受信側ミドルウェアのバッファサイズを調べるために，様々な状況での実験を行なった．

第 6 章では提案手法である帯域幅集約アルゴリズムのデータ配信経路決定アルゴリズムとして EDPF for Middleware を用いることの有効性を確認するために，より単純な手法である WRR (Weighted Round Robin) との比較を行った．既存研究で実験されているネットワーク層での帯域幅集約について追実験を行った後，本研究での提案手法であるミドルウェアにおける帯域幅集約の性能を評価し，両者の結果を比較することで，提案手法の有効性を検討した．

5. 帯域幅集約ミドルウェアの受信側バッファサイズの評価

提案手法である帯域幅集約ミドルウェアを設計するために必要となる値の一つとして，受信側ミドルウェアのバッファサイズが考えられる．同時に使用可能な経路の本数や，帯域幅や遅延などのさまざまな状況に適応した大きさのバッファサイズを確保することを可能にするために様々な状況での実験を行い，必要になる受信側ミドルウェアのバッファの量を考察した．

使用したパケット分配アルゴリズムは単純な手法である WRR (Weighted Round Robin) である．WRR とはそれぞれの経路のボトルネックとなる無線部分の幅の比によりパケットを配信する経路を決定するアルゴリズムである．例えば使用できる経路の無線部分の帯域幅が 200kbps, 100kbps と 50kbps であった場合 4:2:1 の割合でそれぞれの経路へパケットを配信する．

5.1 全体的に低帯域幅の回線におけるシナリオ

図 2 に示されるような 2 本の無線経路を利用できるモバイル端末（ノード 2）がノード 1 から送信されたデータを 2 本の経路を同時に使用して受信するシナリオを実行した．これをシナリオ 1 と呼ぶ．

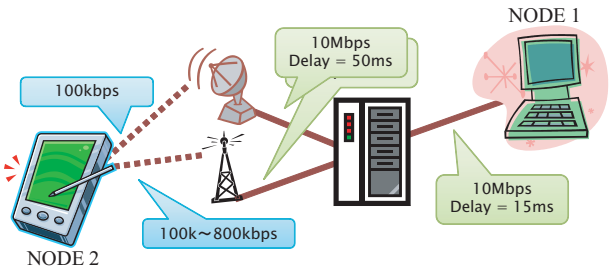

図 2 シナリオ 1（全体的に低帯域幅の回線を持つ環境）

有線部分の帯域幅は 10Mbps で遅延は図のとおりであり，無線部分の帯域幅は片方を 100kbps に固定し，もう片方を 100～800kbps まで 100kbps づつ変化させ，二つの無線の幅の比が 1：1 から 1：8 になるように設定して実験した．使用したトランスポート層のプロトコルは TCP new Reno でパラメータの値は表 1 のように設定した．

表 1 TCP パラメータ

MSS	1460Bytes
Send buffer	65535Bytes
Receive buffer	65535Bytes

5.2 全体的に高帯域幅の回線におけるシナリオ

図 3 に示されるような 2 本の無線経路を利用できるモバイル端末（ノード 2）がノード 1 から送信されたデータを 2 本の経路を同時に使用して受信するシナリオを実行した．これをシナリオ 2 と呼ぶ．

有線部分の帯域幅は 100Mbps で遅延は図のとおりであり，無

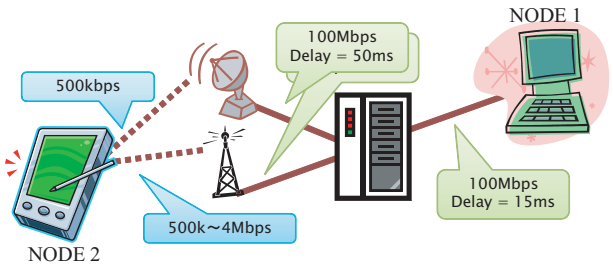

図 3　シナリオ 2（全体的に高帯域幅の回線を持つ環境）

線部分の帯域幅は片方を 500kbps に固定し，もう片方を 500k～4Mbps まで 500kbps づつ変化させ，二つの無線の幅の比が 1：1 から 1：8 になるように設定して実験をした．使用したトランスポート層のプロトコルは TCP new Reno でパラメータの値はシナリオ 1 と同様に設定した．

5.3　様々な状況での実験

さらに図 4 と図 5 に示されるような 2 本の無線経路を利用できるモバイル端末（ノード 2）がノード 1 から送信されたデータを 2 本の経路を同時に使用して受信するシナリオを実行した．

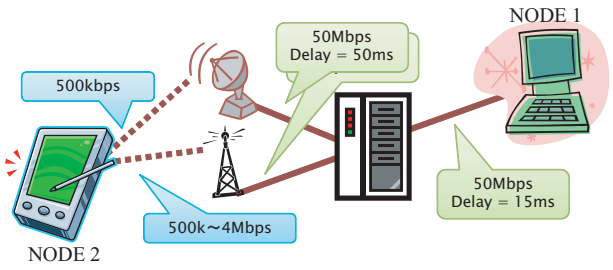

図 4　シナリオ 3（無線部分の帯域幅のみ広い環境）

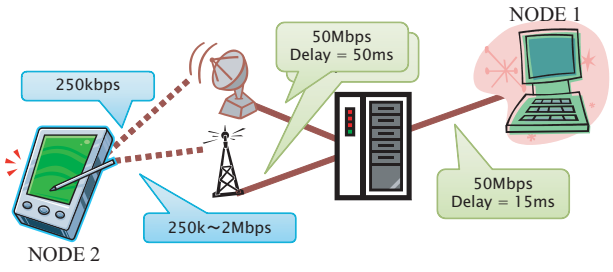

図 5　シナリオ 4（シナリオ 4 と比較して無線部分が狭い帯域幅を持つ環境）

シナリオ 1 やシナリオ 2 と同様に無線部分の帯域幅は片方を一定の値に固定し，もう片方を一定の値ずつ変化させ，二つの無線の幅の比が 1：1 から 1：8 になるように設定して実験をした．使用したトランスポート層のプロトコルは TCP new Reno でパラメータの値はシナリオ 1 と同様に設定した．

5.4　定常状態と定常状態になるまでの時間

シナリオ 1 で無線の帯域幅が 100kbps と 300kbps での実行時のそれぞれの経路のスループットと受信側ミドルウェアのバッファの大きさを図 6 へ表した．通信が開始されしばらく経つと，2 本の無線のスループットはそれぞれ約 100kbps と 300kbps 程度の速度で通信しており，効率の良い通信を行えていることがわかる．受信側ミドルウェアのバッファのサイズは初めは小さいものが徐々に大きくなり，一定の大きさで安定することが分かった．このバッファのサイズが安定した状態を定常状態と呼び，定常状態の際のバッファの大きさと，定常状態にになるまでの時間，2 つの値に注目した．

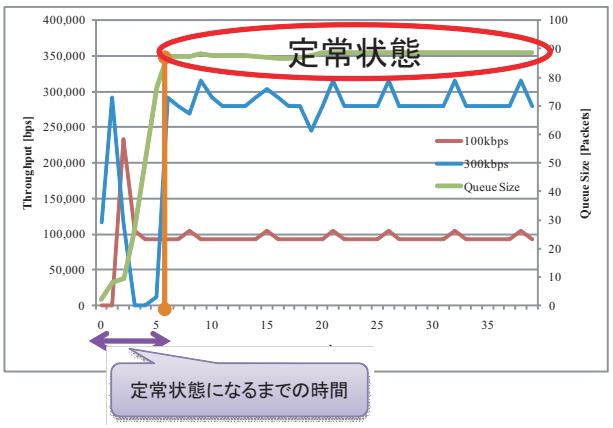

図 6　定常状態と定常状態になるまでの時間

5.5　帯域幅の比とバッファサイズの関係

シナリオ 1 で 2 つの無線の幅を変化させた際の定常状態のバッファサイズがいくらになるか示したグラフを図 7 へ表した．

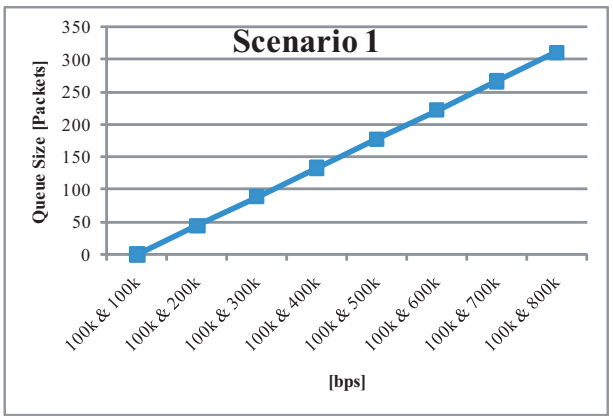

図 7　シナリオ 1 での帯域幅とバッファサイズの関係

無線の幅が同じ時にはゼロであった定常状態のキューの大きさが片方の無線の幅を広くするとそれに比例して大きくなっていくことが分かった．

シナリオ 2 で 2 つの無線の幅を変化させた際の定常状態のバッファサイズがいくらになるか示したグラフを図 8 へ表した．

シナリオ 1 のときと同様に片方の無線の幅が大きくなるにしたがってそれに比例してキューの大きさも大きくなっているこ

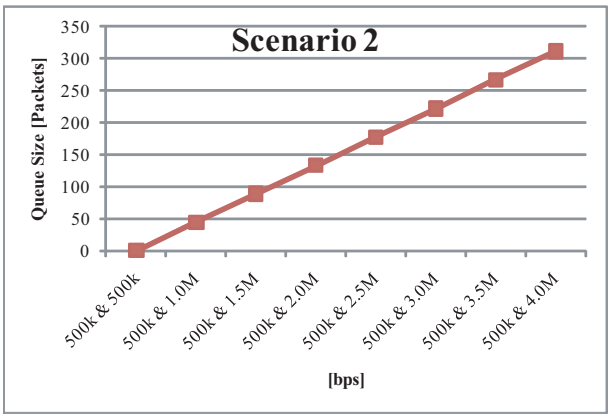

図 8　シナリオ 2 での帯域幅とバッファサイズの関係

とがわかった．さらに，シナリオ 1 とシナリオ 2 では有線部分や無線部分の速度，有線と無線の速度の比が違うにも関わらず，2 つの無線部分の速度の比が同じだと，キューにたまるパケットの数も完全に同じであることが分かった．

さらにシナリオ 3 とシナリオ 4 での 2 つの無線の幅を変化させた際の定常状態のバッファサイズがいくらになるか示したグラフをそれぞれ図 9 と図 10 へ表した．

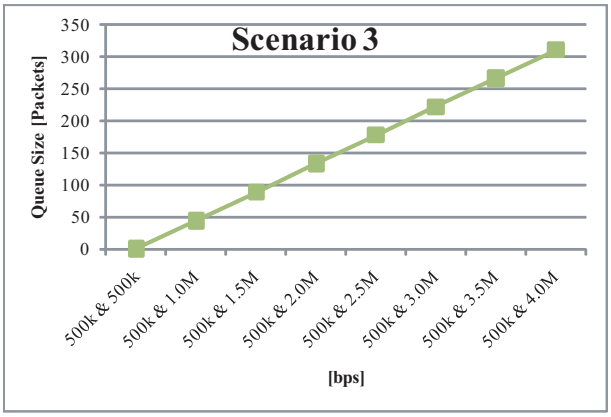

図 9　シナリオ 3 での帯域幅とバッファサイズの関係

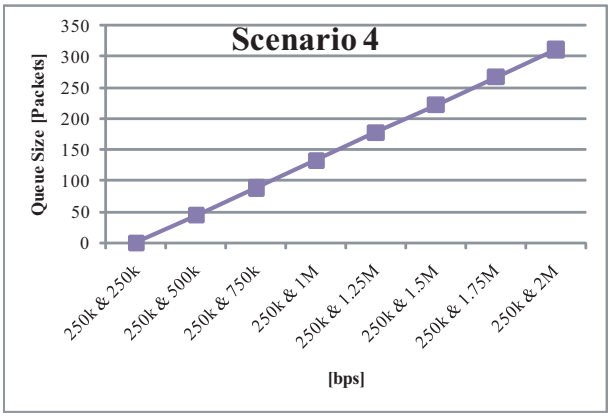

図 10　シナリオ 4 での帯域幅とバッファサイズの関係

5.6　帯域幅の比と定常状態になるまでの時間の関係

すべてのシナリオおける 2 つの経路の幅の比を変化させた場合の定常状態になるまでの時間を調べたグラフを図 11 へ示した．

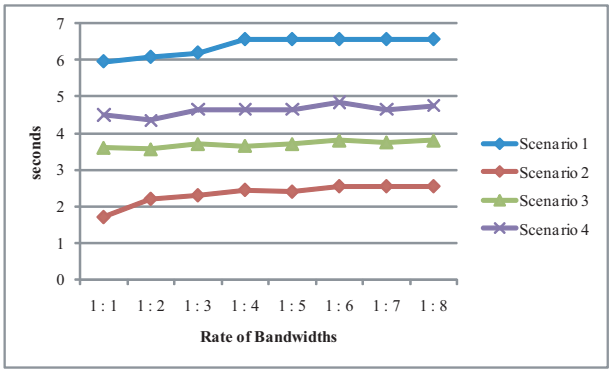

図 11　帯域幅の比と定常状態になるまでの時間の関係

全体的に速度の遅い経路を使うシナリオ 1 と全体的に速度の速い経路を使うシナリオ 2 を比較すると，全体を通してシナリオ 2 の方が定常状態になるまでの時間が短いことが分かった．また，2 本の経路の幅の比が変化しても，すべてのシナリオにおいてそれぞれ定常状態になるまでの時間はあまり変化しないことが分かった．

6.　ネットワーク層での手法とミドルウェアによる手法の比較による評価

提案手法である帯域幅集約アルゴリズムのデータ配信経路決定アルゴリズムとして EDPF for Middleware を用いることの有効性を確認するために，より単純な手法である WRR (Weighted Round Robin) との比較を行った．

既存研究で実験されているネットワーク層での帯域幅集約について追実験を行った後，本研究での提案手法であるミドルウェアにおける帯域幅集約の性能を評価し，両者の結果を比較することで，提案手法の有効性を検討した．

6.1　実験シナリオ

図 12 に示されるような 3 本の無線経路を利用できるモバイル端末（ノード 2）がノード 1 から送信されたデータを 3 本の経路を同時に使用して受信するシナリオを実行した．

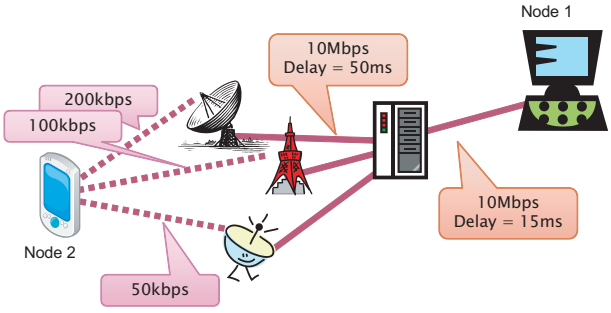

図 12　実験シナリオ

直線であらわした有線部分の帯域幅は 10Mbps で遅延は図のとおりであり，点線であらわした無線部分の帯域幅はそれぞれ 200kbps, 100kbps と 50kbps である．使用したトランスポート層のプロトコルは TCP new Reno でパラメータの値は表 1 のように設定した．

表 2　TCP パラメータ

MSS	1460Bytes
Send buffer	65535Bytes
Receive buffer	65535Bytes

6.2　パケットロスのない環境

ネットワーク層における帯域幅集約と，ミドルウェア層における帯域幅集約を行うにあたり，データ配信経路選択のアルゴリズムとして EDPF と WRR を使用した際にパケットロスのない環境でのそれぞれの経路でのスループットと受信側ミドルウェアのバッファにたまるパケットの量を観察した．

6.2.1　ネットワーク層における帯域幅集約の実験結果

表 3　実験結果 1

アルゴリズム	Thr(kbps)	Dup ACKs	Retransmitted
EDPF	329	0	0
WRR	265	522	96

3 つの経路の帯域幅を単純に合計すると 350kbps なので，WRR ではそれと比較して 75.7 %の性能しか出ていないのに対し EDPF では 94.0 %の性能を発揮することができている．これはパケットロスの際に発生する Duplicate Ack が発生していないので不必要な再転送が行われていないためである．この実験により，既存研究と同様に EDPF が高い性能を発揮することを確認した．

6.2.2　ミドルウェア層における帯域幅集約の実験結果

経路選択アルゴリズムとして EDPF for Middleware を使用した場合のそれぞれの経路でのスループットを図 13 へ，WRR を使用した場合の物を図 14 へ表した．

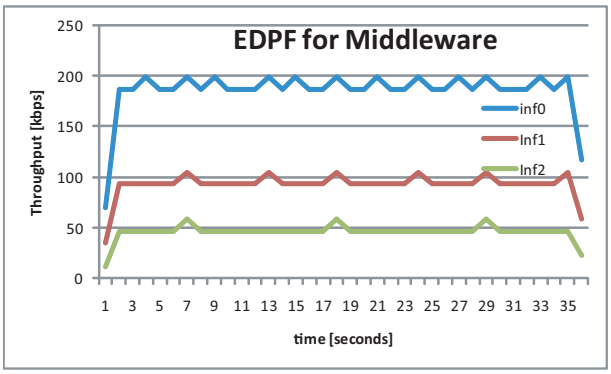

図 13　EDPF for Middleware を用いた場合のスループット

より単純な手法である WRR と比較すると，経路の状況を踏まえてデータを配信する EDPF ではそれぞれの瞬間で安定した通信を行えているということがわかる．

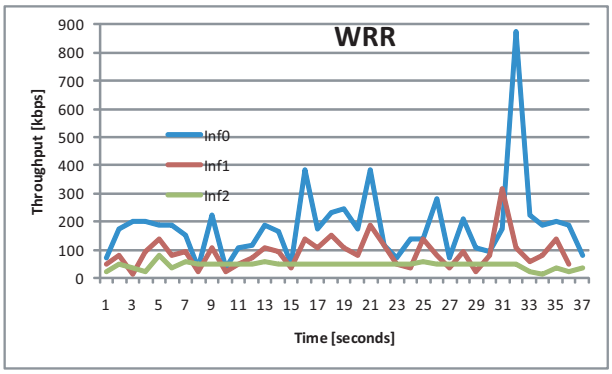

図 14　WRR を用いた場合のスループット

また，受信側ミドルウェアのバッファにたまるパケットの量が EDPF を採用した場合と WRR で採用した場合でそれぞれの瞬間どの程度になっているかを表したグラフを図 15 へ表した．

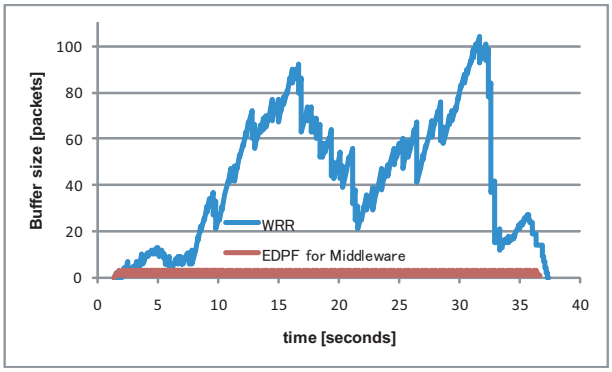

図 15　受信側バッファへ溜まったパケット量

EDPF を採用した場合は最大でも受信側ミドルウェアのバッファへ 3 つのパケット分しかデータが溜まらないのに対して，WRR を用いた場合では最大で 100 パケット分程度バッファにデータが溜まってしまっている．これにより図 14 のような安定しないスループットが見られ，送信アプリケーションから受信アプリケーションまでの遅延も大きくなってしまう．

6.3　パケットロスがひとつ起こる環境

ネットワーク層における帯域幅集約と，ミドルウェア層における帯域幅集約を行うにあたり，データ配信経路選択のアルゴリズムとして EDPF を使用した際にシナリオ開始から 18 秒たったところで一つのパケットロスを起こした際のそれぞれの経路でのスループットを観察した．

6.3.1　ネットワーク層における帯域幅集約の実験結果

14 秒に一つのパケットロスを起こした際に，すべてのインタフェースのスループットが下がってしまっていることを確認した．これはどれか 1 つの経路で起こったパケットロスを，TCP はどこの経路で起こったものか理解できず，全体のスループットを下げてしまっているためである．この実験により，既存研究と同様に EDPF を実装してもパケットロスのある環境では

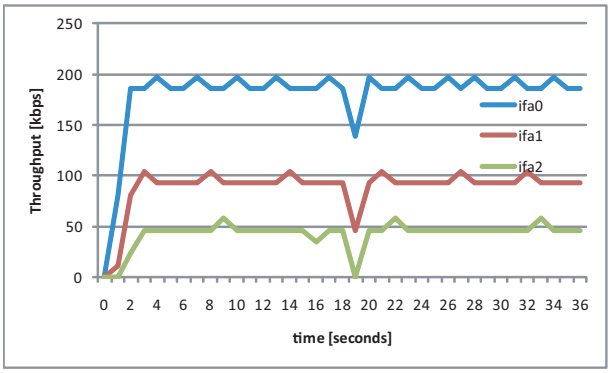

図 16　ネットワーク層での制御でパケットロスが起こった際のスループット

期待した性能を発揮しないということを確認した．

6.3.2　ミドルウェア層における帯域幅集約の実験結果

それぞれの経路でのスループットを図 17 へ表した．

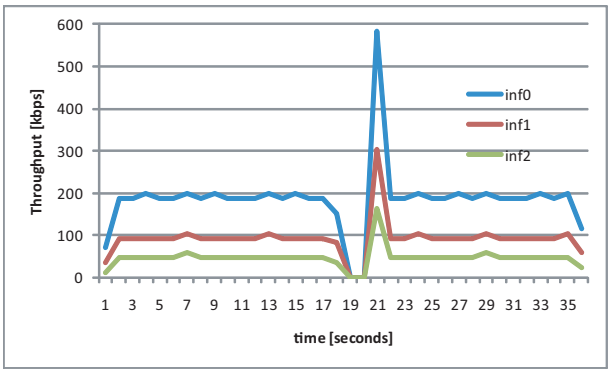

図 17　パケットロスが起こった際の EDPF のスループット

また，受信側ミドルウェアのバッファにたまるパケットの量がどの程度になっているかを表したグラフを図 18 へ表した．

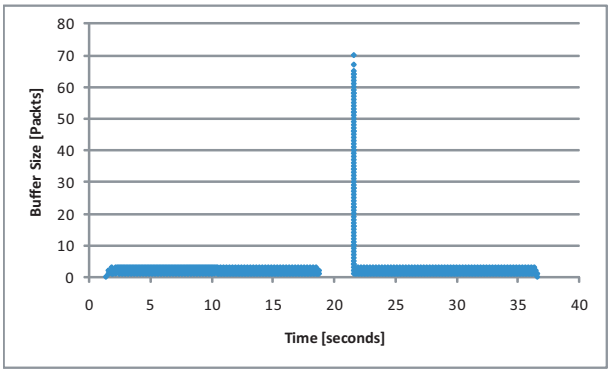

図 18　パケットロスが起こった際の受信側バッファへ溜まったパケット量

図 17 を見るとひとつだけ発生させたパケットロスがすべての経路のスループットを下げてしまっているように見える．しかしこの図のスループットはミドルウェアからアプリケーションへデータを渡せるようになった瞬間のデータ量を表しており，図 18 で見れる通り，パケットロスを起こしたパケットを待つために受信側ミドルウェアのバッファが大きく成長している．このためロスしたパケットを待つためだけにすべての経路のスループットが下がってしまっていると考えることができ，ネットワーク層での制御で起こってしまっていた輻輳ウィンドウの低下による問題ではない．ミドルウェアで観測されるスループットが下がってしまっている間もパケットロスが起こらなかった経路ではデータの転送が続けられ，その間に受信されているパケットはバッファへ貯められている．

7.　結論と今後の課題

本研究では複数の無線インタフェースを同時に用いた通信に関する評価を行うためにネットワークシミュレータを用いて実験を行った．既存研究で提案されている，ネットワーク層で帯域幅集約を行う手法についての検証を行い，パケットロスが発生するとすべてのインタフェースのスループットが下がってしまうという問題点を解決するためにミドルウェアにおける帯域幅集約を提案した．既存研究と同様に，それぞれの経路を使って通信を行ったスループットを束ねた値と同程度の性能を発揮することを確認した．

受信側のミドルウェアへ到着したデータは順番どおり並び替えてからアプリケーションへ渡される必要があるため，ある程度のバッファを持つことが想定される．そのバッファがどのような環境でどれだけ必要になるかを推測するために 2 本の無線経路を持つモバイルノードの経路の幅を変化させた際のバッファの大きさを調べた．2 本の幅の比が大きくなると，それに比例して必要なキューの大きさも大きくなることが分かった．

送信側の帯域幅集約ミドルウェアのデータ配信経路決定アルゴリズムとして EDPF for Middleware と，より単純な手法である WRR を採用してデータの配信をおこなったところ，EDPF for Middleware の方がより効率の良い通信を行うことができるということを確認した．また，パケットロスが起こった際の帯域幅集約の手法による振舞の違いについて，ミドルウェアによる手法の方がネットワーク層での手法より優れた通信を行えるということを確認した．

今後は，パケットロスが起こった際に，より迅速にミドルウェア層のスループットも回復するようにより状況を考慮したデータ配信アルゴリズムを考案する．状況を考慮した高度なアルゴリズム実装のためには様々な TCP パラメータを参考にする必要がでてくると考えられるので，本実験での結果を参考にした制御を帯域幅集約トランスポート層プロトコルへ実装することも検討する．また，本実験では同時に使用可能な無線インタフェースが 3 本であるシナリオのみを実行したが，より使用できる経路の本数が多い場合の検討も行う．さらにパケットロスの頻出頻度が異なる環境や，下位層のプロトコルが異なる場合，有線や無線部分の帯域幅が様々に変化した場合のシナリオを実行し，より適応性の高い帯域幅集約を実現する．

8.　謝　　辞

本研究を行うにあたり，株式会社トヨタ IT 開発センターの

Onur Altintas さんに多大なるご指導，ご鞭撻いただきました．
ありがとうございました．

文　　献

[1] M. Stemm and R. Katz：“Vertical handoffs in wireless overlay networks,” Mobile Networks and Applications Vol.3, No4, pp.335-350, Jan. 1998.
[2] IEEE P802.3ad Link Aggregation Task Force：
http://grouper.ieee.org/groups/802/3/ad/
[3] K. Chebrolu and B. Raman：“Bandwidth Aggregation for Real-Time Applications in Heterogeneous Wireless Networks,” IEEE Transactions on Mobile Computing, Vol.5, No4, pp.388-403, April 2006.
[4] K. Koyama, Y Ito, H. MINENO and S. Ishihara: “Evaluation of Performance of TCP on Mobile IP SHAKE”, Transactions of Information Processing Society of Japan, 2004.
[5] K. Chebrolu, B. Raman, and R.R. Rao：“A Network Layer Approach to Enable TCP over Multiple Interfaces,” J. Wireless Networks (WINET), Vol.11, No5, pp.637-650, Sept. 2005.
[6] M. Zhang, B. Karp, S. Floyd and L. Peterson: “RR-TCP: A Reordering-Robust TCP with DSACK”, IEEE ICNP, 2003.
[7] M. Zhang, J. Lai, A. Krishnamurthy, L. Peterson and R. Wang: “A transportlayer approach for improving end-to-end performance and robustness using redundant paths.” USENIX 2004 Annual Technical Conference, pages 99-112, 2004.
[8] H. Nozawa, N. Honda, K. Sakakibara, J. Nakazawa, and H. Tokuda: ARMS: “Application-level Concurrent Multipath Utilization on Reliable Communication” Internet Conference 2008，Oct. 2008.
[9] Scalable Network Technologies：
http://www.scalable-networks.com/