

次世代無線メッシュネットワークに向けた複数チャネル有効利用手法の高度化

代表研究者 妙中 雄三 東京大学 情報基盤センター 助教
共同研究者 塚本 和也 九州工業大学大学院 情報工学研究院 准教授

1 背景

スマート機器の急増に加えて IoT の普及が予測されており、それらが繋がる無線ネットワークの大容量化が急務となっている。現在の無線ネットワークは LTE を用いた広域網が主であるが、既にモバイル通信量の急激な増加が始まっており、LTE 技術とその高度化のみでは対応が困難となりつつあるため、様々な周波数を扱う複数の無線ネットワークの広域での併用と個々の技術の大容量化が求められている。無線 LAN もそれらの技術の一つに挙げられ、既に物理層・データリンク層では 802.11ac が標準/実用化され、今後 802.11vht も標準化が行われる見込みであり、ネットワーク容量の大容量化が進んでいる。しかし、無線 LAN は物理的に極めて狭い範囲を対象とした無線ネットワーク技術である上に設置できる場所が限定されてしまうため、無線資源の地理的な再利用性が乏しい。従って、無線 LAN は本来利用できる無線資源を広い範囲で有効に利用できていない。

このような背景から、広い範囲に無線 LAN を展開する技術として、無線メッシュネットワーク (WMN) が再注目されている。無線メッシュネットワークは、全てのアクセスポイント (AP) が無線マルチホップネットワークを用いたバックボーン網を構築し、有線接続を持つ一部の AP (GW) がバックボーン網を経由したインターネットへの接続性を WMN 全域に供給する構造を持つ。しかし、WMN は無線 LAN を広域に展開できる一方で、マルチホップ転送によって複数の AP が共有する無線資源 (同一のチャネル資源) を繰り返し利用するため、ネットワーク容量が減少してしまう。インターネットアクセスが主の環境では、GW までのマルチホップ経路上において、経路途中の AP に接続する端末発/宛の通信によって GW へ近づくに連れて転送するトラフィック量が累積するため、GW がボトルネックとなりネットワーク容量が制限される。このような理由から、これまでは WMN の用途が限定されていた。

無線通信では、チャネル単位の容量に理論上限があるため、ネットワーク容量は併用するチャネル数と、それらチャネル資源の利用量 (利用率) に依存する。しかし、WMN では利用するチャネル全ての資源を有効利用できず、ネットワーク容量の最大化が困難である。なぜなら、WMN では AP 毎に端末が接続し、端末発/宛の通信とその量は時々刻々と変化するため、WMN 上のボトルネックが一意の場所とはならず、チャネル資源が未使用が発生してしまうためである。従って、様々な状況に柔軟に適応してネットワーク容量を常に最大化するためには、必要な場所で必要な量だけチャネル資源を活用する柔軟なチャネル利用制御を行う必要がある。

既存研究の殆どがホップ毎に異なるチャネルを事前に割り当て、トラフィックの転送経路を分散させる WMN を用いたネットワーク容量の大容量化技術に着目している。ホップ毎に異なるチャネルを利用することから、マルチホップによるネットワーク容量の減少回避には寄与するが、ホップ毎にチャネルを固定することから、トラフィック状況が変動する場合には変動に適応できず、ボトルネックが発生して全チャネル資源を有効活用できない。同様に、GW のボトルネックも GW の利用チャネル数に依存し、ネットワーク容量の最大化が困難である。

我々はこれまでに、複数のチャネルを全てのホップで並列に利用する WMN アーキテクチャ (SD-WMN: Software Defined WMN) を提案した [1]。このアーキテクチャでは、OpenFlow を用いてフロー単位でホップ毎に利用するチャネルを全チャネルから柔軟に選択・変更できるため、ホップ毎に特定のチャネル利用とはならず、トラフィック状況に応じてチャネル資源を適材適所に有効利用する制御が可能となる。

このアーキテクチャを用いて先行研究 [2] では、複数チャネル間の負荷を平滑化するチャネル有効利用手法を提案した。この平滑化手法では、フロー単位のチャネル利用制御でチャネル負荷を常に均等とすることで、チャネル資源利用の最大化による大容量化を実現した。しかし、実環境ではフロー毎の帯域は様々であるため、平滑化することでパケットロスが発生させてしまう。必要帯域の大きなフローを転送する場合には、平滑化により全てのチャネルに空容量 (残余帯域) があり、合計値では該当のフローを転送できるにも関わらず、どの単一チャネルでも転送できない状況が生じてしまう。

そこで本研究では、様々な帯域のフローに対応するために、負荷の平滑化は行わず、フローを可能な限り特定のチャンネルへ集約するチャンネル有効利用手法（フロー集約手法）を提案する。具体的には、各フローの到着時点でのパケットロス为了避免するために可能な限り残余帯域の多いチャンネルを用いて該当フローを転送した後に特定のチャンネルでの転送に切り替える（集約する）。その結果、様々なフローを転送する場合においても、パケットロスを低減することが可能となる。また、試作機を用いて実環境で提案手法の評価を行い、提案手法の有効性を示す。

2 関連研究

複数チャンネルを用いた WMN は、これまでに数多く研究されている [3-10]。主な研究対象は複数チャンネル構成の WMN での経路制御と複数チャンネルを扱う MAC プロトコルの 2 種類である [6]。

経路制御の研究では、ホップ毎に異なるチャンネルを利用し、経路制御によってトラフィックを複数チャンネルに分散させる手法が提案されている [3-8]。この経路制御では、近接リンクとの干渉を避けるチャンネル割当てが必要となるため、チャンネル割当てと経路制御を連動する手法が研究されている [7, 8]。トラフィックに応じて経路を制御する事で、同一のチャンネル資源を繰り返し利用する事による帯域低下を防ぐ事ができる。しかし、利用するチャンネルが各ホップで固定されてしまうため、一部ホップにトラフィックが集中する場合には、ボトルネックが生じてしまう。また、インターネットアクセスでは、GW にトラフィックが集中してしまい、その部分がボトルネックとなる。従って、チャンネル利用を柔軟に行い、必要な場所で必要なチャンネルを柔軟に利用できる技術が必要である。

そこで MAC プロトコルの研究では、複数のチャンネルを任意に切替えながらトラフィック転送を実施する事で、トラフィックを複数チャンネルに分散させる手法が研究されている [9, 10]。文献 [10] では、1 つの無線インタフェースを用いて、パケット送受信時に使用チャンネルを適切に切替える事により複数チャンネルでのデータ送受信を実現している。この手法によって 1 つのインタフェースしか持たない市販の AP において、空きチャンネル全てを時分割で利用する事が可能となる一方で、チャンネル切替えのための時間の増加 [11] や複数チャンネルの同時利用の困難により、ネットワーク容量の拡張への貢献は薄い。以上の様に、複数チャンネルを使う WMN であっても、必要なチャンネル資源を必要な場所で必要なだけ適材適所で活用する技術の確立が望まれる。

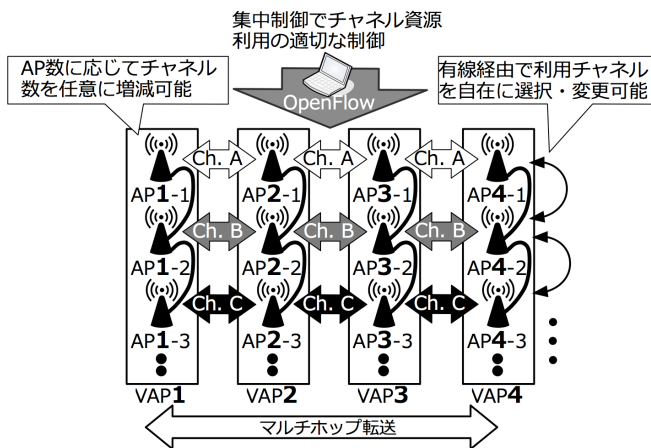


図 1: SD-WMN アーキテクチャ

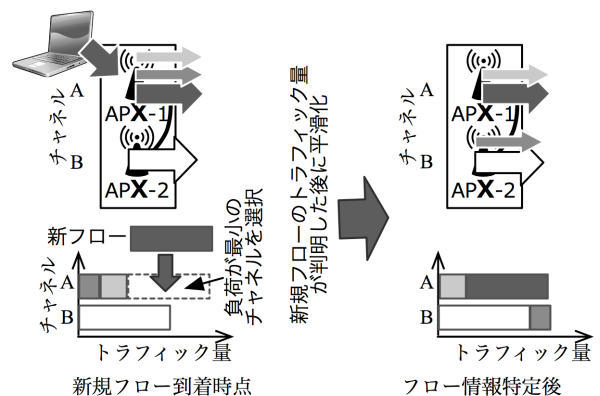


図 2: 平滑化手法の概念図

3 Software Defined WMN (SD-WMN)

本節では、先行研究で提案した SD-WMN のアーキテクチャ並びにチャンネル有効利用手法を説明する。その後、チャンネル有効利用の適用範囲と問題点を明らかにする。先行研究の概要は、3-1 節に、適用範囲と問題点は 3-2 節に示す。

3-1 アーキテクチャとチャンネル有効利用手法

先行研究では、図 1 に示す仮想 AP (VAP) と VAP を用いた WMN を提案した。複数の AP を有線接続し、個々の AP は異なるチャンネルを利用する。この有線接続された AP 群を、複数チャンネルを持つ 1 つの VAP として扱

い、それらで WMN を構築する。その結果、理想的には各 VAP に AP を制限なく追加でき、追加する AP 数に応じて WMN 上で併用するチャンネル数を柔軟に増やす事ができる。

WMN のネットワーク容量を最大化するためには、全ホップで併用する複数チャンネルを適切に利用する必要がある。先行研究では、上記構成の WMN に中央制御型の OpenFlow 技術を導入した Software Defined WMN (SD-WMN) を提案した。SD-WMN では、OpenFlow コントローラ (OFC) が OpenFlow プロトコルを用いて (送信元・先の IP アドレスとポート番号の組で定義する) フロー単位で VAP 毎に任意の制御パスを設定・変更が可能となる。また、OpenFlow は SD-WMN 全域の情報を集約できることから、各ホップの各チャンネルでのトラフィック転送状況も収集が可能である。これらの特徴を用いて、SD-WMN は潜在的に様々なトラフィック状況に対応して、適切なチャンネル利用が可能となる。

先行研究で提案した手法では、SD-WMN 上で各 VAP の電波範囲内において、図 2 の様に全チャンネル間でトラフィック負荷を平滑化することでチャンネル有効利用を行い、ネットワーク容量の最大化を実現するチャンネル有効利用手法 (平滑化手法) を提案した。具体的には、OFC は新規フローに対して、到着時点では帯域を収集できないため、一時的に負荷の少ないチャンネルを用いて転送する (図 2 左)。その後、OFC が該当フローの帯域を把握した段階でチャンネル間での負荷を均等とする様にフローの転送チャンネル変更を行う (図 2 右)。帯域の収集は OpenFlow の FlowStats を用いて定期的に取得し、その取得毎に平滑化処理を実施する。なお、FlowStats で得られる結果は累積の転送量であるため、2 回送信した差分をフローの帯域として用いた。

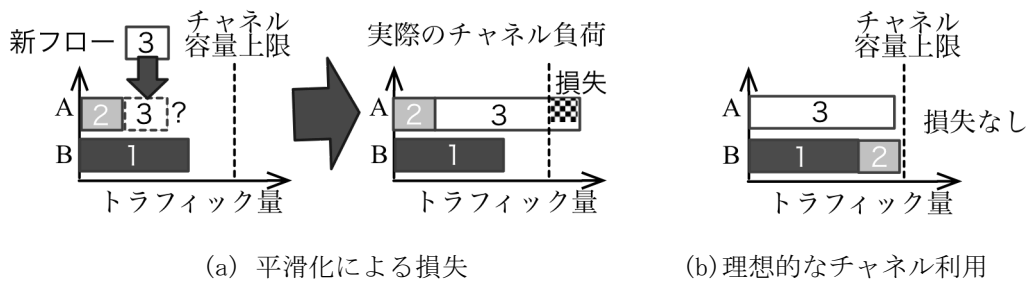


図 3: 平滑化手法の問題点と理想状態

3-2 先行研究の適用範囲と問題点

平滑化手法では、チャンネルを有効利用できる状況が限定される。具体的には、平滑化手法は新規フローの帯域と到着順に制限がある。3-1 節で説明した通り、新規フローは、到着時点で帯域が不明であるため、トラフィック負荷が少ないチャンネルを用いて一時的に転送される。平滑化手法を用いてチャンネル利用を制御していると、図 3(a) の様に、新規フローの帯域が転送するチャンネルの空き容量 (残余帯域) を超過し、パケットロスが発生する可能性がある。一方で、フローが 1、3、2 の順で到着する場合には図 3(b) の様に損失が発生しない。従って、平滑化手法が有効に働くのは、単一のチャンネルの残余帯域を超えない範囲の帯域のフローが順番に到着した場合に限定される。

この様に到着順によっては適切なチャンネル割り当てとなることから、平滑化手法がフローの到着順に依存する点が問題である。これは、平滑化することで、フローの増加に応じて新規フローが利用できる残余帯域がどのチャンネルでも減ってしまうことに起因する。よって、この問題を解決するためには、一部のチャンネルの残余帯域を可能な限り最大化して、新規フローを転送する際のパケットロスを回避する必要がある。

4 フロー集約手法

本手法では、3-2 節で述べた通り、一部チャンネルの残余帯域を最大化し、新規フロー到着時点でのパケットロスを予防するチャンネル有効利用を提案する。新規フローについて、到着時点では残余帯域が最大となるチャンネルを用いて一時的に転送し、帯域が判明した時点で特定のチャンネルに集約する。また、転送中のフローが終了した場合には、その終了によって増加した残余帯域を活用するために、新規フローに関わらず他のチャンネルからフローを集約する。これらを繰り返すことで、一部のチャンネルで十分な残余帯域が確保され、新規フローの到着時点におけるパケットロスを可能な限り回避できる。ただし、フローを特定のチャンネルに集約する際には、チャンネルの残余帯域にフローが収まることを正確に確認する手法も必要となる。以上より、本研究のポイントである残余帯域の計算方法を 4-1 節、新規フロー到着時のチャンネル選択手法を 4-2 節、フローの集約方法を 4-3 節、フロー終了時のフローの再集約手法を 4-4 節でそれぞれ説明する。

4-1 チャンネルの残余帯域の算出方法

特定のチャンネルにフロー集約を行うためには、フローの帯域とチャンネルの残余帯域を適切に管理する必要がある。本節では、チャンネルの残余帯域の算出方法を説明する。無線ネットワークでは、トラフィックの転送量は容易に計測できる一方で、残余帯域の正確な計測は困難である。本研究では転送量から残余帯域を推定する。

まず、OFC が各 AP から各チャンネル上の全てのフローの統計情報を取得する。この統計情報には、フロー毎の転送パケット数と転送バイト数の累積値が含まれる。累積値からフローの帯域を取得するために、0.5 秒間隔で 2 度統計情報を取得し、差分を取得する。つまり、0.5 秒間に転送されたパケット数とバイト数を計測する。また同時に、OFC が OpenFlow とは異なる外部プログラムを用いて、全 AP から各ホップの無線リンク速度を取得する。

次に上記で収集したフロー毎の統計情報と無線リンク速度からチャンネルを利用した合計時間 (Airtime) を計算する。Airtime は、フレーム送信毎のオーバーヘッド (プリアンブルやヘッダ、FEC、ACK 等) と転送バイト数の和を無線リンク速度で除算し、その結果にフレーム毎に発生する DIFS/SIFS の制御時間を加えることで計算する。この結果は、上記のフロー統計情報の計測間隔 0.5 秒間の間で、データ送信に利用された合計時間となる。この Airtime を用いて、チャンネルの未使用時間を計算 ($0.5 - \text{Airtime}$) し、無線リンク速度を乗算することで残余帯域を予測する。

4-2 フロー到着時の転送チャンネル決定方法

新規フローの到着時には、そのフローの帯域は不明であるため、一時的に可能な限り残余帯域の多いチャンネルを用いて転送することで、チャンネル容量の超過によるパケットロスを回避する。具体的には図 4 に示す様に、新規フローが獲得し得るチャンネル容量 (獲得期待容量) を用いて、一時的な転送チャンネルを決定する。帯域が判明しているフローのみが存在する場合 (図 4 左) には、獲得期待容量は残余帯域と等しいため残余帯域が最も多いチャンネルを選択する。

一方、新規フローが複数到着し、帯域が不明のフローが既に存在する場合 (図 4 右) には、全チャンネルを対象に新規フローの獲得期待容量を計算して比較する。まず、OFC はフローの詳細情報はなくとも既に転送中のフローの数は保持している。図 4 右の様に未知のフロー 2 をチャンネル A で転送している場合にはチャンネル A の残余帯域を $2 (= [\text{既存の未知のフロー数}] + 1)$ で割った値をフロー 3 の獲得期待容量として算出する。この例では、チャンネル A 上では未知のフローのみが転送されているため、チャンネル容量をフロー数で除算する。一方、チャンネル B 上では既知のフロー 1 のみが転送されている事から、フロー 3 の獲得容量期待値はチャンネル容量からフロー 1 の消費分を減算した値となる。これらチャンネル A/B での獲得期待容量を比較し、図 4 右の例ではその値が大きいチャンネル B が選択される。こうする事で、新規フローが頻繁に流入する場合においても、可能な限り多くの容量を個々のフローが利用できるチャンネルを用いて転送することでパケットロスの可能性を最小限に抑える。

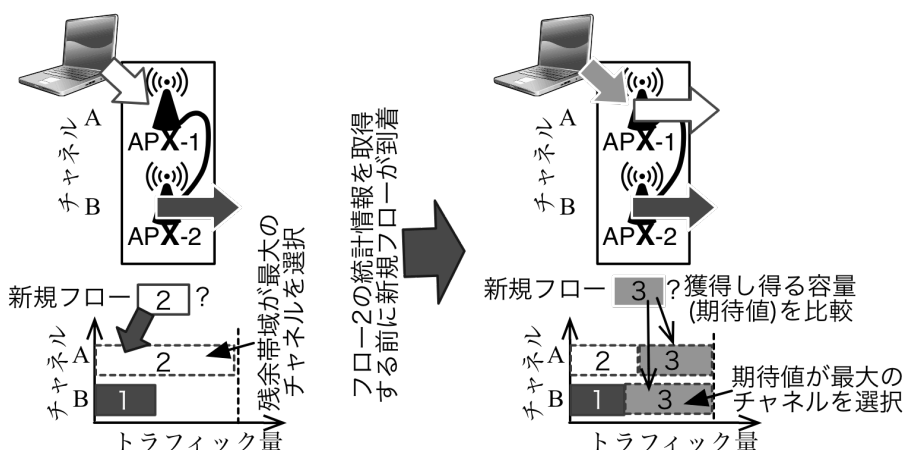


図 4: 一時的な転送チャンネル決定の実施例

4-3 必要帯域判明時のフロー集約方法

本手法は 4-2 節の新規フローの処理と並行して定期的にフロー情報を収集し、未知のフローの情報が判明した場合に該当フローを特定のチャンネルでの転送に切り替える。集約先のチャンネルには、十分な残余帯域が

確保できないものから順番に選択する。フロー情報には、OFC が定期的に（3 秒毎に）フロー単位の統計情報を（0.5 秒間隔で）2 度取得してその差分を用いる。フロー情報の収集で全てのフロー情報が集まることから、全フローの情報から 4-1 節で説明した方法によってチャンネル毎の残余帯域を計算する。ここで計算した残余帯域は 4-2 節の処理でも活用する。

OFC はフロー情報を取得し、直前まで未知であったフローの情報を特定できた場合にフロー集約を実行する。OFC はフローの帯域を計算し、特定のチャンネルに集約可能であるかを確認する。具体的には、フローの帯域を用いて残余帯域の少ないチャンネルから順に収容可能かを比較し、収容可能なチャンネルが存在した場合には、フローを該当チャンネル上での転送に切り替える。フローの移動先が無い場合には、転送チャンネルの切替えを行わない。なお、上記の処理はホップ単位で行う。

この様に残余帯域の少ない一部のチャンネルにフローを集約し続けることで、残りのチャンネルでは残余帯域が最大限確保される。その結果、4-2 節の新規フローに対する一時的なチャンネル決定が有効に働き、流入時のチャンネル容量超過によるパケットロスを予防することができる。

4-4 フロー終了時のフローの再集約手法

フロー終了時には、該当フローの帯域分だけフローを転送していたチャンネル（終了チャンネル）の残余帯域が増加する。この時、新規フローの到着と集約を待たずに、既存フローの再集約を行う。OpenFlow では、フローの終了時に各 AP から OFC へ通知が届くため、この通知を契機に再集約を実行する。

OFC はフロー終了の通知を受け取ると同時にフロー終了前に取得したフロー情報から終了したフローの帯域と終了チャンネルの残余帯域を計算し、フローの帯域分だけ残余帯域を加算することでフロー終了時点での終了チャンネルの残余帯域を予測する。最も残余帯域の大きいチャンネルから順番に、そのチャンネル上で転送中のフローについて、集約が可能か確認する。具体的には、フローの帯域と終了チャンネルの残余帯域から収容可能かどうかを判断し、残余帯域がフロー帯域より大きい場合には該当のフローの転送を終了チャンネル上に切り替える。残余帯域が終了チャンネルより少ないチャンネルについては、上記の処理は行わず、条件を満たすフローが見つからない場合には再集約は実施しない。

5 実環境での評価実験

5-1 実験環境

本実験では図 5 に示す通り、VAP および各 AP をそれぞれ 0.7m 離して設置した。実装に使用する AP は Buffalo 製 WZR-HP-AG300H を使い、OpenWrt ファームウェアをインストールした。また、OpenFlow スイッチとして、OpenWrt 上に OpenvSwitch をインストールした。なお、無線干渉が無い環境を調査した上で 100 と 112、124、136 チャンネルの 4 チャンネルを WMN を構成するために用意し、54Mbps 固定の IEEE802.11a で構成した。実験では、Linux をインストールした PC1 と PC2 を用意し、それぞれ VAP1 と VAP4 に接した。その上で PC1 と PC2 の間で iperf を用いたトラフィック送受信を行う。なお、本研究はデータトラフィックに対するチャンネル有効利用が研究対象であるため、端末と AP 間や OFC と AP 間は全て有線接続とする。

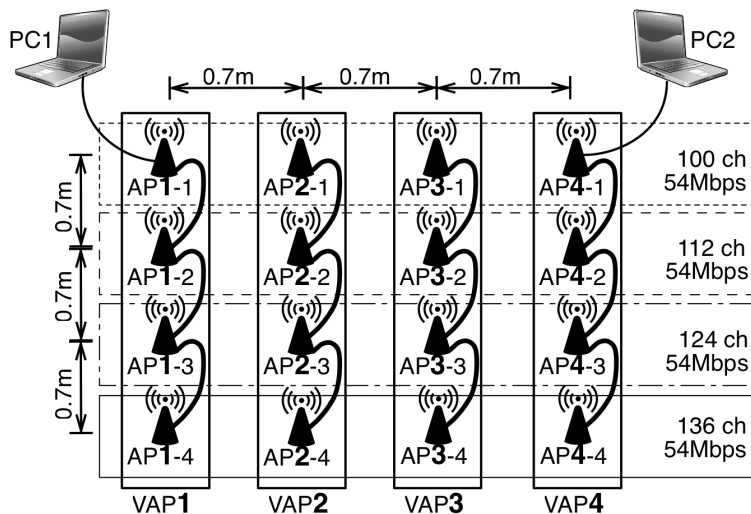


図 5: 実験環境

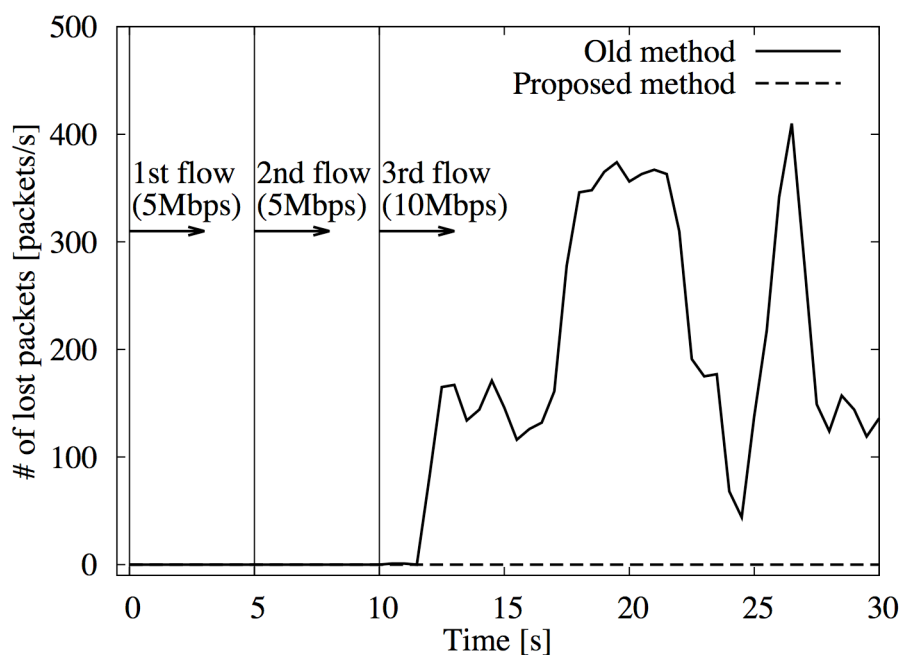


図 6: パケットロス数の時間変化

5-2 基礎性能評価

本節では、提案するフロー集約手法が先行研究の平滑化手法に比べてチャンネル有効利用を実現できる適用範囲が広いことを示すために、3-2 節であげた適応範囲外の例を元にした実験シナリオで性能評価を行う。利用する SD-WMN は、図 5 に示す 100 と 112 チャンネルの 2 チャンネルのみを使うトポロジ構成を取る。実験シナリオは、1,500 バイトの packets を用いた 5Mbps の UDP フローを 2 つと 10Mbps の UDP フロー 1 つを 5 秒間隔で順番に PC1 から PC2 へ送信する。送信開始したフローは、30 秒間の実験終了まで継続してトラフィック送信を続ける。

図 6 に実験期間中の 1 秒ごとのパケットロス数を示す。この図から提案手法では、実験を通して常にパケットロスが 0 であることがわかる。一方、平滑化手法では、3 つ目のフローが送信された直後からパケットロスが発生している。2 本目のフローが送信された後、平滑化手法では 2 つのチャンネルのトラフィック負荷を均等にするために、5Mbps のフローがそれぞれのチャンネルで 1 つずつ送信される様に、フローの転送チャンネルの切替えを行う。しかし、この平滑化によって両方のチャンネルの残余帯域が減少してしまう。本実験環境では、54Mbps 固定の 802.11a を用いている事から 1 チャンネルを 3 ホップする場合の可用帯域は約 10Mbps である。それに対して、どちらのチャンネルも既に 5Mbps のフローを転送しているため、3 つ目のフローが到着してどちらのチャンネルで転送してもチャンネル容量超過によるパケットロスが発生する。その後、平滑化手法はチャンネル負荷を平滑化するために、FlowStats を収集することでフローの転送量から帯域を計測するが、既にパケットロスが生じているため正確な帯域計測が行えずに誤ったチャンネル切替えを繰り返してしまい、パケットロス数が大きく変化している。

一方で提案手法は、2 フロー目が到着した後に全てのフローを片方のチャンネルに集約するため、もう一方のチャンネルは未使用状態となり、そのチャンネルの残余帯域が十分に確保される。3 本目のフローが到着した時点では、フローの帯域は不明であるが、残余帯域の大きなチャンネルで一時的な転送を開始するため、パケットロスが発生しない。また、フローの帯域を特定した後はフローの集約を試行するが、既に 1/2 つ目のフローでチャンネルが埋まっているため、チャンネル切替えは行わない。その結果、全てのチャンネルが最大限有効利用され、ネットワーク容量を最大化できる。つまり、2 チャンネル・3 ホップの最大容量である 20Mbps を損失なく転送できている。

以上の結果より、平滑化手法に存在したフローの帯域と到着順に対する手法の適用範囲がフロー集約手法には存在しないことがわかる。従って、提案手法ではネットワーク容量に空きがある限り、到着フローに依存せずに最大限チャンネルを有効利用が可能となる。

表 1: 実験施行毎のペケットロス数

手法	最大値	中央値	最小値
平滑化手法	6826	4229	1196
フロー集約手法	8	7	4

5-3 実践的なトラフィックによる評価

本節では、様々な到着タイミングと様々な帯域のフローを模擬した環境で実験を行い、フロー集約手法の有効性を示す。実験では、あらかじめランダムに生成したトラフィックを1パターン準備する。フローの生成は、(1)フローの到着間隔は1秒以下、(2)フローの継続時間は1秒以上10秒以下、(3)フローの転送レートは固定で9Mbps以下、(4)同時に送信する合計スループットは36Mbps以下の4つの条件を満たした範囲でランダムに100フロー生成する。なお本実験では、図5の4チャンネル3ホップの実験トポロジで、生成した100フローをPC1からPC2へ送信する。同一フローパターンで9回の実験を行う。

表1に実験試行単位でのペケットロス数の計測結果を示す。本実験では、同時に転送するトラフィックを必ず36Mbps以下となる様にフローを生成しているため、チャンネルを有効利用できていれば、全てのトラフィックを転送できるはずである。しかし、平滑化手法では多くのペケットロスが生じている。3-2節で説明した通り、平滑化する事によりフロー数とそれらの帯域によってチャンネル毎の残余帯域が低下してしまうため、新規フロー到着にチャンネル容量超過が発生してペケットロスが増加する。その際、OFCが実測するフローの帯域にも誤差が生じてしまい、誤ったフローのチャンネル切替えを実施してしまう。この差分は、実験毎に差異が生じるため、ペケットロス数も実験毎に差異がある。

一方、フロー集約手法ではペケットロス数が全ての実験において数ペケットに抑えられている。本手法では、フローの到着順序や帯域に依存せず、新規フローについて到着時点では十分な残余帯域のあるチャンネル上で一時的に転送し、フローの帯域が判明した段階で特定のチャンネルに集約するため、チャンネル容量超過を可能な限り回避できている。したがって、ランダムに生成したフローに対しても、フロー集約手法が良好な性能を示した事から、本手法は広い適応範囲の元でチャンネル有効利用を実現したと言える。

4 まとめ

本論文では、フローを特定のチャンネルに集約して一部のチャンネルで十分に残余帯域を確保する事により、新規フロー到着時のチャンネル超過によるペケットロスを大幅に削減した。残余帯域は、フローの統計情報からチャンネルの合計占有時間を算出し、その空き時間を用いて予測する手法を提案した。新規フロー到着時とフロー集約、フロー終了時の再集約手法を提案し、これらを組み合わせる事で十分な残余帯域を確保できる手法を実現した。フロー集約手法を実環境で評価し、先行研究に比べて大幅にペケットロスが削減できる事を示した。今後は、複数チャンネル間で無線送信レートに差がある場合や、AP間の距離を適切に調整した上で詳細な調査を行う必要がある。

【参考文献】

- [1] Y. Taenaka and K. Tsukamoto, "An efficient traffic management framework for multi-channel wireless backbone networks," IEICE Communications Express, vol.3, no.3, pp.98–103, March 2014.
- [2] Y. Taenaka and K. Tsukamoto, "A radio interference aware dynamic channel utilization method on software defined WMN," proceedings of In Proceeding of the 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC2015), p.736-741, Nov. 2015.
- [3] S. Mustafa, S.A. Madani, K. Bilal, K. Hayat, and S.U. Khan, "Stable-path multi-channel routing with extended level channel assignment," International Journal of Communication Systems, vol.25, no.7, pp.887–902, 2012.
- [4] E. Alotaibi and B. Mukherjee, "Survey paper: A survey on routing algorithms for wireless ad-hoc and mesh networks," Computer Network, vol.56, no.2, pp.940–965, Feb. 2012.

- [5] P.H. Pathak and R. Dutta, "A Survey of Network Design Problems and Joint Design Approaches in Wireless Mesh Networks," IEEE Communications Surveys and Tutorials, vol.13, no.3, pp.396–428, 2011.
- [6] D. Benyamina, A. Hafid, and M. Gendreau, "Wireless Mesh Networks Design - A Survey," IEEE Communications Surveys and Tutorials, vol.14, no.2, pp.299–310, 2012.
- [7] A.A. Franklin, A. Balachandran, and C.S.R. Murthy, "Online reconfiguration of channel assignment in Multi-Channel Multi-Radio wireless mesh networks," Elsevier Computer Communications, vol.35, no.16, pp.2004–2013, 2012.
- [8] W. Si, S. Selvakennedy, and A.Y. Zomaya, "An Overview of Channel Assignment Methods for Multi-radio Multi-channel Wireless Mesh Networks," Journal of Parallel and Distributed Computing, vol.70, no.5, pp.505–524, May 2010.
- [9] K.S. Vijayalayan, A. Harwood, and S. Karunasekera, "Distributed Scheduling Schemes for Wireless Mesh Networks: A Survey," ACM Computing Survey, vol.46, no.1, pp.14:1–14:34, July 2013.
- [10] J. So and N.H. Vaidya, "Multi-channel Mac for Ad Hoc Networks: Handling Multi-channel Hidden Terminals Using a Single Transceiver," Proceedings of the 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), pp.222–233, 2004.
- [11] A. Raniwala and T. Chiueh, "Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network," Proceedings of the 24th Annual IEEE International Conference on Computer Communications (INFOCOM), pp.2223–2234, 2005.

〈発表資料〉

題名	掲載誌・学会名等	発表年月
An Experimental Approach to Examine a Multi-Channel Multi-Hop Wireless Backbone Network	An International Journal of Computing and Informatics	2015年
A Channel Utilization Method for Flow Admission Control with Maximum Network Capacity toward Loss-free Software Defined WMNs	Proc. of the fourteenth International Conference on Networks (ICN2015)	2016年2月
A radio interference aware dynamic channel utilization method on software defined WMN,	Proc. of the 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC2015)	2015年11月
Software define WMN のパケットロス低減を実現するフロー集約手法	電子情報通信学会	2016年1月
〔奨励講演〕 Software Defined WMN における一括フロー制御による制御トラヒック削減手法	電子情報通信学会	2015年12月
無線メッシュネットワークのパケットロスを低減する遅延バインディング型フロー割当による複数チャネル有効利用手法	電子情報通信学会	2015年5月