OpenFlowを用いたTCPトラフィック動的経路分散ネットワークの提案

吉村 悠[1] 佐藤 健哉[1]

概要：インターネット上のトラフィック量は年々増加しており, 中でもインターネットビデオの TCP トラフィックはその大きな要因となっている. また既存のルーティングプロトコルは最適経路上のリンクやノードにトラフィックが集中する欠点があり輻輳の要因となる. そこで TCP トラフィックを最適経路だけでなく複数の経路に分散させることが輻輳の抑制に有効であるが，適切な経路分散を行うにはネットワークの帯域状態から動的に経路設定をする必要があり，既存の IP 通信網では複雑な実装となり困難である. そこで本研究では OpenFlow を用いてネットワークのトラフィック状態に合わせて動的な経路分散を行う.OpenFlow は近年，注目を集めている SDN(Software-Defined Networking) と呼ばれるネットワークをソフトウェアで制御する仕組みの代表的な技術である．OpenFlow によってネットワーク状態を監視し，新たに発生した TCP フローに対し経路を動的に設定する．また短時間にトラフィックが急増するバーストトラフィック発生時にも適切に経路分散を行えるよう設計を行う．結果として提案手法を既存手法とを比較し提案手法はスループットが向上，再送率が低下し輻輳の抑制に効果があることを示した.

Dynamic TCP Traffic Route Distribution Networking Using OpenFlow

HARUKA YOSHIMURA[1] KENYA SATO[1]

1. はじめに

1.1 背景

インターネットを利用したアプリケーションの普及により，ネットワーク上のトラフィック量は年々増加しており，今後も増加していくことが予想されている. 中でも動画共有サイトにおいて増加，高品質化しているインターネットビデオの TCP トラフィックはその大きな要因となっている [1]. そのため膨大な量の TCP トラフィックがある状況下でもアプリケーションが利用できるようネットワークの輻輳を抑制する仕組みが必要となっている．対策として挙げられるのがルーティングプロトコルの改善である．RIP[2] や OSPF[3] といった既存のルーティングプロトコルは特定の経路を最適な経路として選択し利用する最適経路ルーティングプロトコルである．これらのルーティングプロトコルは様々な場所で利用されているが，最適経路上のリンクやノードにトラフィックが集中する傾向があり輻輳のリスクが高まってしまう欠点がある．インターネットには最適経路以外にも宛先ノードに通じる冗長経路が存在しており最適経路だけでなく複数の経路を伝送に使用しネットワーク全体の使用率を上昇させることが輻輳の抑制に有効である．しかしインターネットは特定の時間帯にアクセスが集中し [4] 短期間にトラフィックが急増するバーストトラフィック [5] を発生させる特徴があり，複数経路伝送で輻輳の抑制を行うにはトラフィックが急激に増加している場合においても適切に複数経路にトラフィックを分散できる手法を考案する必要がある.

1.2 目的

本研究では複数経路を使用した TCP トラフィック伝送を実現し，ネットワーク使用率を向上させ輻輳を抑制することを目的とする．そのために OpenFlow[6] を用いてネットワークに必要な機能を実装する．またバーストトラフィックに対しても経路分散できるフロー管理を検討する.

1.3 本論文の構成

本論文では第 2 章で複数経路伝送の課題と OpenFlow の

[1] 同志社大学理工学院研究科情報工学専攻

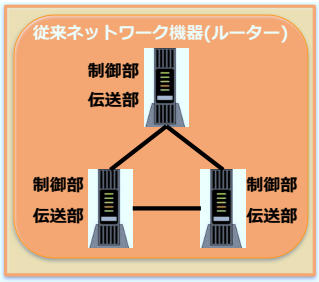

図 1　従来ネットワーク

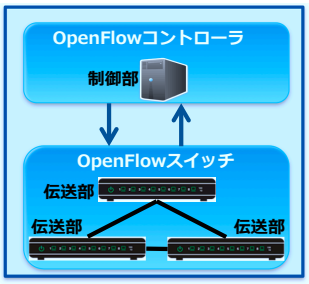

図 2　OpenFlow ネットワーク

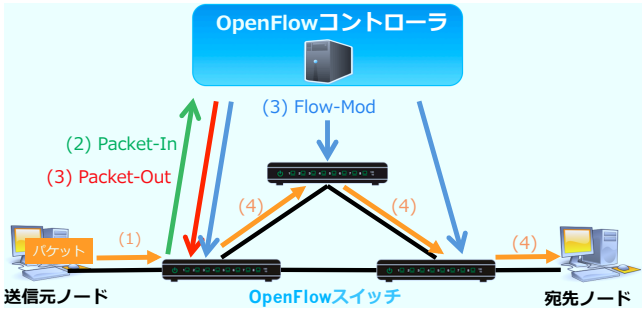

図 3　パケット伝送の基本動作

特徴について解説し第 3 章で提案手法の構成や動作にていて説明する．第 4 章で提案手法の実装環境と評価環境，評価結果について説明し第 5 章でその評価結果に対する考察を行う．最後に第 6 章で本研究のまとめを行う．

2.　複数経路伝送と OpenFlow の特徴

2.1　複数経路伝送の課題

本研究での複数経路伝送とは，TCP トラフィックを最適経路のみでなく冗長経路も同時に利用し分散，伝送することである．これを実現するためには主に，以下の 4 つの課題の解決が必要となる．

複数の経路の発見

経路を分散するためには伝送元のノードからある宛先ノードに対して複数の経路を用意しなければならない．

ネットワークの空き帯域の把握

動的に伝送経路を選択するためには現在の各リンクの負荷情報を収集し，各経路の空いている帯域幅を把握しなければならない．

動的な経路の選択と伝送

各経路の負荷情報と伝送に必要とする帯域から経路を動的に選択し，その経路にそってトラフィックを伝送しなければならない．

パケット順序逆転問題

パケット単位で経路を分散し伝送すると，同一の TCP フローのパケットが異なる経路で伝送されパケットの到達順序の逆転が頻発してしまう．TCP 通信においてこのパケット順序の逆転が頻発すると，TCP の再送処理などが発生しかえってスループットが減少してしまう．よって TCP 通信の場合，同一の TCP コネクションのトラフィックは同じ経路で伝送し，TCP コネクション毎に経路を分散する必要がある．

これらの課題の解決は各々のネットワーク機器が独立して動く既存の IP 通信網ではそれぞれの機器に機能を追加する複雑な実装となり困難である．そこで本研究では柔軟な伝送制御が可能な OpenFlow ネットワークを利用する．

2.2　OpenFlow のパケット伝送動作

近年，SDN(Software-Defined Networking) と呼ばれる，ネットワークをソフトウェアによって管理，制御する考え方が広まってきており，OpenFlow はその代表的な技術である．**図 1** のように，ルータ等の従来のネットワーク機器は，パケットの操作内容を制御をする機能と実際にパケットを転送する物理的な機能が同一機器内に結合しているが OpenFlow では，**図 2** のように OpenFlow コントローラと呼ばれる制御部と OpenFlow スイッチと呼ばれる転送部が別機器として分離している構造となっており，伝送用ネットワークを構成する各 OpenFlow スイッチと OpenFlow コントローラが相互に通信しパケットを伝送する．各スイッチはフローテーブルを持ち，コントローラがフローテーブルに伝送ルールであるフローエントリを書き込み伝送制御を行う．ここで OpenFlow ネットワークにおけるパケット伝送の基本動作手順を以下に示す．また，動作の全体図を**図 3** に示す．

(1) 送信元ホストからパケットが隣接スイッチに届く．

(2) ホストからパケットを受け取ったスイッチは，フローテーブルからパケットに合うマッチングルールを持つフローエントリを探索する．該当するフローエントリがあれば，そのフローエントリに従いパケットを伝送する．該当するフローエントリがなければ，コントローラへ Packet-In メッセージを送信する．

(3) Packet-In メッセージにはスイッチが受け取ったパケットの情報が記載されており OpenFlow コントローラは，その情報を基にパケットに対するフローエントリを作成し，各 OpenFlow スイッチに Flow-Mod メッセージによって追加する．以後同様のフローのパケットは追加したフローエントリによって処理されるが Packet-in メッセージを送った OpenFlow スイッチはフローエントリ検索を終えてしまっているため，最初のパケットはコントローラが Packet-Out メッセージによってアクションを明示的に実行させる．

(4)　以降パケットは各スイッチのフローエントリに従い伝送されていく．

2.3　関連研究

宮本らの研究 [7] では複数経路による TCP 通信を行う上での課題を示し，既存 IP 網に対し機能を追加する形で実

装している．しかしトポロジの制限や経路選択を行うノードの限定などパケットループ防ぐための都合上の制限がある．川島らの研究 [8] では移動端末が外部ネットワークと通信を行う際に近隣の移動端末同士が協力ネットワークを構築し，自身の外部リンクだけでなく協力ネットワーク内の他端末の外部リンクも利用し複数経路伝送を行う SHAKE（SHAring multiple paths procedure for cluster networK Environment）の検討を行っている．その中で経路選択には事前に静的なポリシーを設定するのではなく各経路のパケットロス率，遅延ゆらぎなどの帯域情報から動的に経路選択ポリシーを変更していく必要性を示している．土屋らの研究 [9] では OpenFlow を用いた MPLS（Multi-Protocol Label Switching）ネットワークにおける迂回経路制御を提案している．経路選択に Relay Node アルゴリズムを用いてトラフィックを均等に分散させネットワーク全体の最適な負荷バランスを保ち性能を向上させている．迂回経路は最適経路の使用率が 80 %を超えた場合選択される．一方でバーストトラフィックについての考察はなされておらず，トラフィックがゆるやかに上昇している場合と急に上昇している場合とで閾値や動作に変わりはない．

2.4 問題点

既存のルーティングプロトコルでは特定のノードやリンクにトラフィックが集中し輻輳のリスクが高まってしまう．対策として OpenFlow を利用し最適経路以外も伝送に使用する複数経路伝送が考えられる．しかし短時間にフローが大量に増加するバーストトラフィック発生時においては，フローの経路選択時とフローの設定後でネットワークの状態が大きく変化し意図した負荷分散が行われない場合がある．例えば，フロー発生時にネットワークの帯域情報から帯域により余裕のある経路を選択する場合，同時に発生した大量のフローの経路選択に使用する帯域情報は同じ時点のものなのでそれらのフローに同じ経路を設定してしまい，意図する負荷分散が行えない．そのため複数経路伝送で輻輳の抑制を行うにはフローが急激に増加した場合においても複数経路にトラフィックを適切に分散できる仕組みが必要になる．

3. 提案手法

本章では TCP トラフィックの複数経路伝送と動的な負荷分散を実現する提案手法について解説する．また，提案手法の構成や動作手順を説明する．なおこれ以降では断りのない限りコントローラは OpenFlow コントローラ，スイッチは OpenFlow スイッチを意味する．

3.1 概要

OpenFlow によって複数経路伝送を実現し，ネットワークの使用率を向上し輻輳を抑制する．図 4 のように，本来

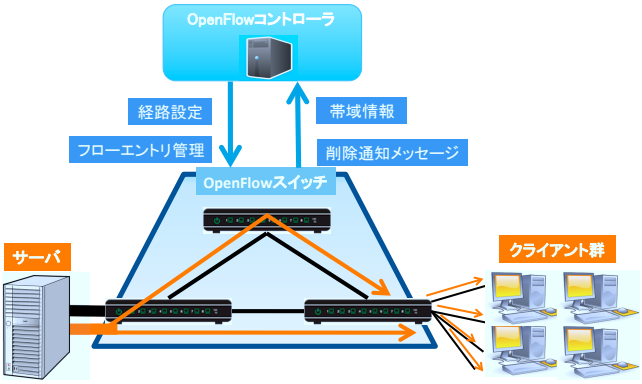

図 4 提案手法概要

一つの経路に集中するフロー群を複数経路に分散する．各スイッチから帯域情報を収集しコントローラが適切な経路設定やフローの管理を行う．経路選択の基準には各経路のボトルネックとなるリンクの空き帯域幅を使用する．また再設定時間をフローエントリにセットすることで，再設定時間毎に最新の帯域情報からフローエントリを再設定し経路の再選択を行う．再設定時間をネットワーク状態に合わせて増減させることで動的な経路再選択を行う．

3.2 提案手法の構成

提案手法に必要な機能はコントローラをプログラミングし実装する．実装内容は大きく分けて，以下の二つである．

複数経路伝送の基本機能

第 2.1 節で挙げた課題を解決するための機能群，提案手法の複数経路伝送動作を担う．

動的経路再選択機能

一度設定した各フローエントリを最新の帯域情報から再設定する機能．再設定時間ごとに書くフローが適切な経路へ分散し直される．再設定するまでの間隔を帯域情報から動的に決定することでバーストトラフィックなどのトラフィックの急激な増加に対応できるよう設計する．

以下で各機能の動作について解説する．

3.3 複数経路伝送の基本機能

複数の経路の発見

複数の経路の発見のために，まず LLDP(Link Layer Discovery Protocol) パケットを用いてネットワークトポロジを把握する．LLDP パケットはあるスイッチから隣接スイッチへ，次にその隣接スイッチからコントローラへと伝送される．LLDP パケットには経由してきた経路情報が保存されておりこれを読み取ることでコントローラは経由してきたスイッチ間の接続関係を把握する．この LLDP パケットを各スイッチに発信させ，ネットワーク全体のトポロジを把握する．把握したネットワークトポロジを元に経路探索を行い，ノー

ド間の複数の経路を発見する．

ネットワークの空き帯域の把握

コントローラはスイッチへ Stats-Request メッセージを送信することで，スイッチからフローテーブルに保存された統計情報を受け取ることができる．Stats-Request メッセージは数種類あり，必要な統計情報に応じて使い分ける．今回は各スイッチのポートごとの使用帯域を把握するため，PortStats-Request メッセージを送信し，統計情報を収集する．この動作を 5 秒毎に繰り返し，トポロジ情報と合わせて各リンクの空き帯域を把握する．

動的な経路の選択と伝送

新たな TCP コネクション形成時，第 2.2 節で示したようにフローエントリを作成し経路上の各スイッチに追加する．伝送経路には宛先までの経路の中で最もボトルネックとなるリンクの帯域幅が大きい経路を使用する．事前に上記の動作によって把握した宛先への複数経路，各リンクの空き帯域の情報を基に経路計算を行い使用する経路を選択しておき，その経路に合わせてフローエントリの作成，追加を行いこの TCP コネクションのフローを伝送する．経路の選択は 1 秒毎の定期的な帯域情報の更新時と新たなフローエントリが設定された後のタイミングに行われ，動的な経路の選択と伝送を実現する．同一フローのパケットは同じフローエントリによって伝送されるのでフロー単位に経路分散が行われパケット順序逆転問題も解決される．

これらの動作により複数経路伝送が実現され，フロー単位の経路分散が行われる．

3.4 動的経路再選択機能

ネットワークの状態変化に対応するため，再設定時間を決定しフローエントリが設定されてから再設定時間が経過すると最新の帯域情報を基にフローエントリを再設定し経路の再選択を行う．動的経路再選択にはフローエントリのオプション値を利用する．使用するオプション値の種類を表 1 に示す．これらのオプション値のうち hard-timeout,cookie,priority を利用し必要な機能を実現する．packet-in メッセージを受けて経路上の各スイッチに各フローエントリを設定する際に packet-in メッセージを送信したスイッチのフローエントリの hard-timeout オプションに再設定時間をセットし，hard-timeout オプションによってフローエントリが削除された際に経路を再選択し各フローエントリを再設定する．再設定時間はフローエントリ作成時の帯域使用率やフローエントリの増加速度から動的に決定し，バーストトラフィック発生時など大量のフローが作成されている状況では再設定時間を短かくすることで急激なネットワーク状態の変化に対応できるようにする．cookie オプションは削除されたフローエントリを再設定する必要性があるか (使用中であるか) 判断するのに使用する．また動的経路再選択の間にパケット伝送を行う priority 値を下げた予備フローエントリを予め設定しておく. 具体的な動作を以下で解説する．

表 1 使用するオプション値

オプション名	デフォルト値	内容
hard timeout	なし	設定後フローエントリが削除される時間
cookie	-	自由に数値設定可能なクッキー
priority	0xffff	フローエントリの優先度.

予備フローエントリの設定

フローエントリが削除されてから再設定されるまでの間，伝送されてくるパケットを伝送する予備フローエントリを削除されるフローエントリと同時に設定しておく. 予備フローエントリはフローエントリの再設定終了時に削除，再設定される．

再設定時間の決定

式 1 より再設定時間 T を計算する．ここで Blink は経路上でボトルネックとなるリンク，Dswitch は宛先となるスイッチ，U(Blink) はボトルネックとなるリンクの帯域使用率を表し，Iflow(Dswitch) は宛先スイッチへのフロー増加速度が上昇中の時には 2，低下または同じ場合は 1 となるパラメータである．α，β は T の最大値，最小値を決定する定数であり今回は最小値を 1，最大値を 30 とするため $\alpha = 29$，$\beta = 1$ としている．これによりボトルネックリンクの空き帯域幅とフロー増加状況に応じて再設定時間 T を計算する．例えば使用する経路のボトルネックリンクの帯域使用率が 90%，フローエントリ数の増加速度が増加中の場合再設定時間 T は 2.45 秒となる．

$$T = \alpha(1 - U(Blink))/Iflow(Dswitch) + \beta \quad (1)$$

フローエントリの再設定

決定した再設定時間をフローエントリ作成時に hard-timeout オプションに設定する．hard-timeout が設定されたフローエントリは，フローテーブルに追加後設定した時間経過すると自動的に削除されスイッチからコントローラに削除通知メッセージが送信される. 通知メッセージに含まれる情報を表 2 に示す．通知メッセージ内のマッチングルール情報をもとにフローエントリを再設定するが，その際にこのフローエントリのフローが継続中かどうか判断しフローが終了している場合，フローエントリの再設定は行わない．

フローの継続判定

今回は Windows OS で，TCP/IP 通信のデフォルトのタイムアウト時間である 21 秒間通信が行われなかったフローは終了したものと判断する．よって削除通知メッセージの packet_count 値が 0 かつ duration_sec

表 2　削除通知メッセージ内容

項目	内容
datapath_id	メッセージ送信元スイッチの ID
byte_count	フローエントリが処理したバイト数合計
packet_count	フローエントリが処理したパケット数合計
match	フローエントリのマッチングルール
cookie	フローエントリのクッキー値
duration_sec	フローエントリが有効であった秒数
duration_nsec	フローエントリが有効であったナノ秒数
idle_timeout	フローエントリの idle_timeout 値
5 秒 priority	フローエントリの priority 値
reason	フローエントリの削除理由

値が 21 以上の場合フローが終了たものとしてフローエントリの再設定は行わないということになる．しかしフローエントリは再設定されると統計情報もリセットされるため 21 秒より短い間隔で繰り返し再設定が行われると duration_sec 値が 21 以上になることがなくフローエントリが再設定され続けてしまう．そのため提案手法ではフローエントリを再設定する際にオプションの一つである cookie に duration_sec 値を格納し判定に用いる．判定動作の手順を以下に示す．

(1) 削除通知メッセージの packet_count を確認し 0 の場合 cookie 値に duration_sec 値を足す．0 以外の場合は cookie 値に 0 をセットする
(2) cookie 値が 21 以上になった場合，このフローエントリは使用されていないと判断し，経路再選択動作を中断する
(3) cookie 値が 21 未満の場合，最新の帯域情報からフローエントリを再設定する．その際にこの cookie 値を再作成するフローエントリの cookie 値としてセットする

3.5　動作手順

提案手法の複数経路伝送と動的経路再選択の動作手順を以下に示す．またシーケンス図を**図 5** に示す．

(1) 各スイッチに LLDP パケットを発信させ，トポロジ情報を収集する．トポロジ情報から経路探索を行う
(2) 各スイッチから統計情報を収集し，統計情報から各リンクの帯域使用率とフロー増加速度平均を割り出す
(3) 探索した経路の中から各リンクの帯域使用率から伝送経路を選択しておく
(4) （2）～（3）を周期的に繰り返す
(5) フローが発生し，パケットがスイッチに届くとスイッチのフローテーブルからパケットに合うフローエントリを検索する

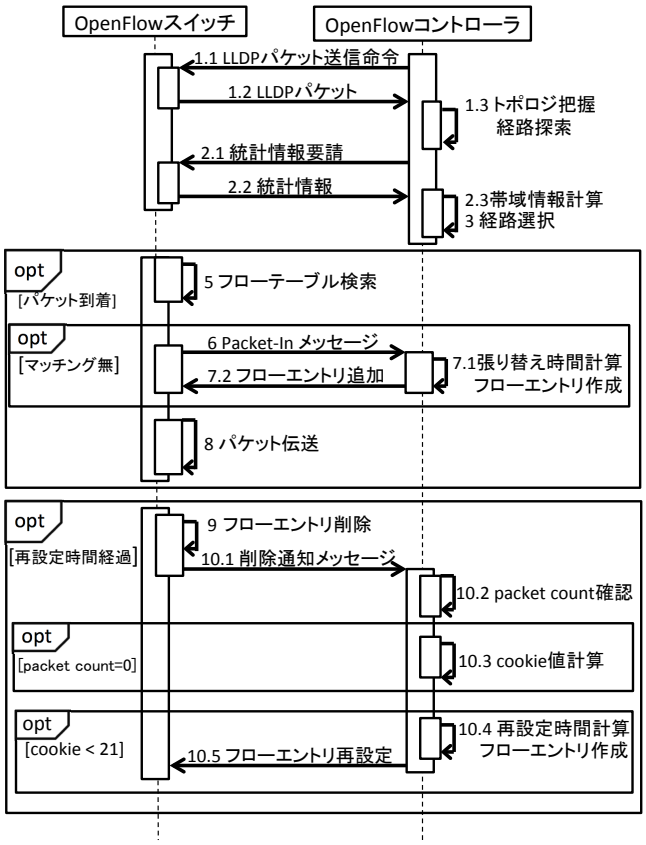

図 5　動作シーケンス図

(6) マッチするフローエントリが無い場合，コントローラへ Packet-In メッセージを送信する
(7) コントローラは（3）で決定した経路に従ってフローエントリを作成する．また再設定時間を計算し hard-timeout 値に設定し経路上のスイッチにフローエントリと予備フローエントリを設定する
(8) フローエントリに従って各スイッチがフローを伝送する
(9) 再設定時間が経過したフローエントリをスイッチが削除し，通知メッセージをコントローラへ送信する
(10) フローエントリのフローが継続中か判定を行い，継続中の場合フローエントリを再設定し経路再選択を行う

4.　実装・評価

本章では，提案手法の実装環境，評価環境について解説する．

4.1　実装環境

本研究ではコントローラのフレームワークである Trema を使用して実装を行う．開発言語は Ruby を使用した．Trema で提案手法の機能をコントローラに実装し，linux の仮想 OpenFlow ネットワーク内で

表 3　実装環境

OS	Ubuntu14.04LTS
CPU	Intel Core i5 CPU@3.00GHz × 4
メモリ	5.8GB
開発言語	Ruby
コントローラ	Trema Version 0.4.7
スイッチ	Open vSwitch Version 2.0.2

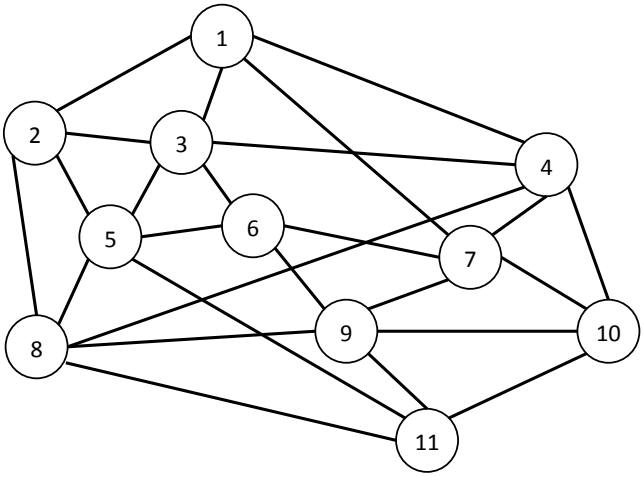

図 6　Cost239 トポロジー

動作させる．なお仮想のスイッチにはソフトウェア OpenFlow スイッチである Open vSwitch を使用する．実装環境を表 3 に示す．

4.2　評価ネットワークトポロジ

評価には欧州都市間ネットワークモデルである Cost239 トポロジー (図 **6**) を使用し，隣接ノード間のホップ数は全て 1 としている．また各ノード間のリンクはアウトバンド帯域幅を 2Mbyte/s に制限した．

4.3　評価手法

提案手法の評価は iperf，wireshark による TCP トラフィック測定によって行った．ノード 1，2，4，8 からノード 9 に対して TCP フローを生成しその平均スループットと再送率を測定した．それぞれのノードからのコネクション数を 4 から 24 まで 4 つずつ増やし，合計コネクション数が 16，32，48，64，80，96 となるそれぞれの場合で測定を行った．図 **7** にノード 1，2，4，8 からノード 9 への最短経路を示す．ノード 8 からノード 9 へのリンクがフローの集中するボトルネックリンクとなっている．提案手法の経路制御の妥当性を評価するためこのボトルネックリンクの使用帯域変動を測定した．また比較のため RIP ルーティングプロトコルにより最短経路のみで伝送を行った場合についても同様に測定した．

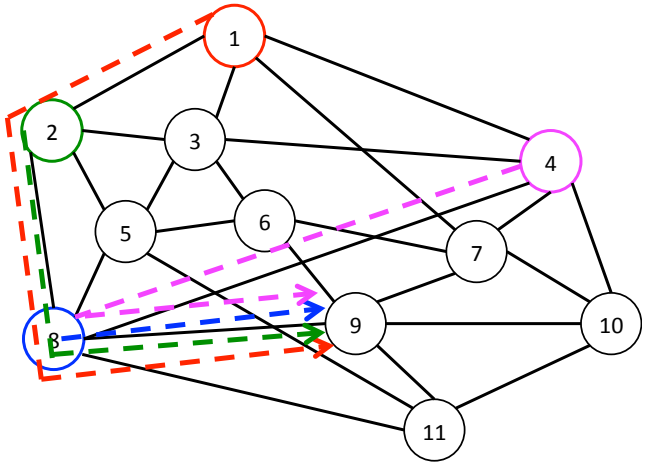

図 7　ノード 1，2，4，8 からノード 9 への最短経路

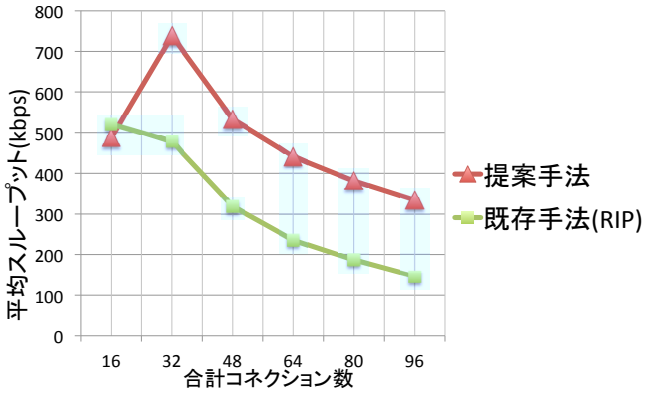

図 8　平均スループット

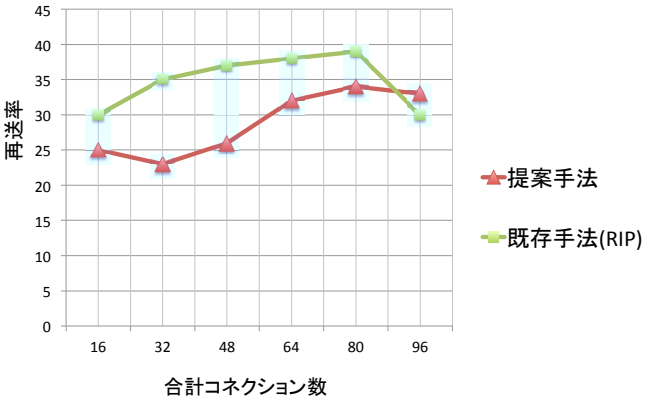

図 9　平均再送率

4.4　評価結果

評価結果として提案手法と既存手法でのそれぞれの平均スループット，再送率をそれぞれ図 **8**，図 **9** に示す．またノード 8 からノード 9 へのボトルネックリンクの使用帯域変動を図 **10**，図 **11** に示す．

5.　考察

5.1　平均スループット，再送率の結果について

提案手法は既存手法と比べて最適経路以外の冗長経路を利用することで輻輳が抑制され平均スループットが

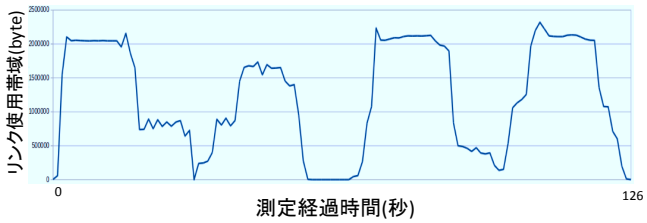

図 10　ボトルネックリンク使用帯域変動 (提案手法)

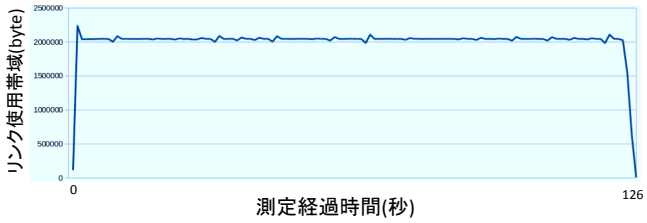

図 11　ボトルネックリンク使用帯域変動 (RIP)

最大約 128%上昇している．総コネクションが 96 本の時点で再送率が逆転しているがこれは既存手法で，フロー開始時から輻輳が頻発し TCP の輻輳制御により各フローのスループットが大きく下げられたからである．

5.2　使用帯域変動の結果について

既存手法のボトルネックリンク使用帯域はフロー開始時から終了するまで帯限界値である 2MByte を保ち続けこのリンクで輻輳が起こり続けていることが分かる．対して提案手法では輻輳状態になってから数秒から 10 数秒でにフローが張り替えられ輻輳状態から脱している事が分かる．しかし使用帯域がほぼゼロになるまでフローを張り替えてしまっていたり，帯域が空いてくると他の経路からフローを集めて再び輻輳状態に戻ってしまっている．これは全てのフローに大して現在最も帯域に余裕のある経路を選択するよう設定した為である．より効率よく伝送するためには使用帯域が常に 9 割程度を保つようフローを張り替える必要があり経路選択方法を改善していく必要がある．

6.　まとめ

インターネット上のトラフィック量は年々増加してきておりネットワークの輻輳を抑制する仕組みが必要となっている．しかし既存の RIP や OSPF のような最適経路ルーティングプロトコルでは特定リンクやノードに負荷が集中しやすい傾向がある. 対策として複数経路にトラフィックを分散する複数経路伝送が考えられるがバーストトラフィックのような急激に増加するトラフィックも適切に複数経路に分散する必要がある. 本研究では OpenFLow ネットワークを利用し提案手法を実装した.OpenFlow は近年，注目を集めている Software-Defined Networking の代表的な技術で，柔軟なネットワーク制御を可能とする．OpenFlow を用いて複数経路伝送における課題を解決し複数経路伝送を実現するとともにバーストトラフィックのような急激なネットワークの状態変化に対応するため動的経路再選択機能を考案した．提案手法ではリンクの帯域使用率やフロー増加速度に応じた間隔で経路再選択が行われバーストトラフィック発生時などにはより細く経路分散を行い輻輳を抑制する．OpenFlow コントローラフレームワークである Trema を用いて提案手法を実装し，既存手法と比較評価を行った. 提案手法は再送率の低下やスループットの向上が見られ，輻輳の抑制に効果があることを確認した．しかし今回はフローを最も帯域に余裕のある経路に割り振ったために最適経路に余裕があるのに冗長経路に割り振ってしまうなど非効率的な振る舞いが見られ改善が必要である．

謝辞

本研究の一部は JSPS 科研費（JP16H02814）の助成を受けたものである．

参考文献

[1] cisco: VNI Traffic and Service Adoption Forecast，入手先 ⟨https://www.ciscoknowledgenetwork.com/files/448_06-24-14-VNI_Traffic_and_Service_Adoption_Forecast_Presentation__CKN_MOBILITY_.pdf⟩（参照 2016-02-01）.

[2] IETF: RFC 1058 Routing Information Protocol，入手先 ⟨https://tools.ietf.org/html/rfc1058⟩（参照 2016-02-01）.

[3] IETF: RFC2328 OSPF Version 2，入手先 ⟨https://www.ietf.org/rfc/rfc2328⟩（参照 2016-02-01）.

[4] 総務省: 我が国のインターネットにおけるトラヒック総量の把握，入手先 ⟨http://www.soumu.go.jp/main_content/000244628.pdf⟩（参照 2016-02-01）.

[5] 森達哉，川原亮一，内藤昭三: インターネットトラフィックの統計解析 ～傾向と対策～，テレコムサービス協会技術研究会最新動向セミナー，入手先 ⟨http://nsl.cs.waseda.ac.jp/ mori/talks/telecom-200203.pdf⟩（参照 2016-02-01）.

[6] N.McKeown，T.Anderson，H.Balakrishnan，G.Parulkar，L.Peterson，J.Rexford，S.Shenker，J.Turner，OpenFlow: Enabling Innovation in Campus Networks，SIGCOMM，Volume:38，pp.69-74，2008.

[7] 宮本正和，家永憲人，動的負荷分散型 IP 網の構成法と経路制御アルゴリズムの性能に関する研究， 電子情報通信学会技術研究報告， 2002.

[8] 川島佑穀，峰野博史，石原進，水野忠敬，複数経路通信における動的トラフィック制御の検討，情報処理学会研究報告 ITS[高度交通システム]，2004.

[9] 土屋俊貴，川原崎雅敏，MPLS-OpenFlow 結合アーキテクチャを用いた迂回制御方式，信学技報，Volume:114，No.494，pp.25-30，2015.