効率的かつ公平なデータ転送実現のための帯域制御技術

平成 18 年度

秦野　智也

目 次

第1章　序論

1.1 まえがき

現在，日本のブロードバンドネットワークは低廉かつ広帯域なのものになり，ブロードバンドの契約数も増加している [1]．国民のネットワーク利用状況をみると，ネットワークの利用人口は 2005 年末で約 8,529 万人に達し，人口普及率は 66.8 %まで達するなど，ネットワークは国民生活に浸透している．ネットワークへのアクセスは，パソコンだけではなく様々な端末から可能となり，職場や家庭だけでなく移動中を含めたあらゆる状況で「いつでも，どこでも，誰でも」ネットワークを利用する状況になりつつある．それに伴い，ネットワークは国民に大きな利便，恩恵をもたらす生活の必需品になってきている．特に企業においては，ブロードバンド，モバイルネットワークの活用が進展し，新たなネットワーク活用方法が生まれつつある．さらに，法人向けネットワークにおいては，広帯域サービスばかりでなく，高信頼なネットワークサービスへの関心が高まってきている．このように，国民と情報通信ネットワークとの関わりはますます強くなりつつあり，高いサービス品質への期待も大きくなっている．

ネットワークの利用状況に着目してみると，ネットワークアプリケーション数が増加し，ネットワーク利用者が増加，多様化してきている．ネットワークを流れる通信の特性も多様化し，海外への通信，衛星回線を利用した通信，広帯域転送を必要とする通信が現れてきている．また，高いサービス品質の要素として，高いユーザ間公平性，優先制御を実現すること，効率よく帯域を使用することが求められている．このように，通信の範囲が広がることにより伝播遅延が増大するため，ネットワークの広帯域化が進んだ現在においても，ネットワークに対して高いサービス品質を維持してデータ転送をすることが重要視されている．

現在，帯域が広く，伝播遅延が大きいネットワーク，即ち，帯域遅延積の大きなネットワークでは以下の課題を抱えている．ネットワークの伝播遅延が大きくなることによって生じる，コネクションの伝播遅延時間が異なる場合，TCP 輻輳制御機構の性質上，伝播遅延時間の短いコネクションが長いコネクションよりも転送帯域の増加量が大きい．その結果，伝播遅延時間の短いコネクションの獲得帯域が多くなり，公平性が損なわれる．また，ネットワーク内の広帯域化が進み，伝播遅延が増大した場合，TCP 輻輳制御機構の性質上，転送帯域の増減の周期が長くなる．その結果，転送帯域の増加に時間がかかり，ネットワークの帯域を十分に活用できない時間が長くなり，結果的に帯域を効率よく使用できなくなる．

本研究の目的は，帯域遅延積の大きなネットワークにおいて，ネットワーク内のコネクションに対して高いサービス品質を実現する帯域制御技術を提案することである．本研究では，帯域遅延積の大きなネットワークが抱える帯域制御技術に関する課題に対して，伝播遅延時間が異なることによって生じるコネクション間の公平性の課題と，帯域が拡大し，伝播遅延が増大することによって生じる効率の課題を分けて考察を行い，課題を解決する提案を行う．伝播遅延の増大によって引き起こされる課題に対して，ネットワークの改善，エンドホストとネットワークの連携による改善に焦点を絞り，解決を図る．一方，帯域の拡大，伝播遅延の増大によって引き起こされる課題に対しては，エンドホストの改

善に焦点を絞り，解決を図る．

1.2 帯域制御に関する技術

効率的かつ公平なデータ通信サービスの提供を実現する技術として，ネットワークにおけるサービス品質の概念を述べる．次いで，それを実現する既存技術として，本研究で検討を行うネットワーク層の Diffserv 網およびトランスポート層のエンドエンドで帯域転送を行う技術の TCP について述べる．

1.2.1 ネットワークサービス品質

現在，ネットワークを用いてデータを転送するために，OSI (Open Systems Interconnection) 参照モデルで紹介されているように，物理層，データリンク層，ネットワーク層，トランスポート層，セッション層，プレゼンテーション層そしてアプリケーション層において，階層化されたプロトコルが機能している．OSI 参照モデルにおいて，ネットワークのサービス品質を実現する層として，IP (Internet Protocol)，Diffserv (Differenciated Services) 網に代表されるネットワーク層，TCP (Transmission Control Protocol) に代表されるトランスポート層が担っている．ネットワーク層は，エンドホストまでデータを届けるための通信経路の選択，通信経路内のアドレスの管理を行い，トランスポート層は，エンドポイントまで確実に効率よくデータを届けるためのデータ圧縮や誤り訂正，再送制御などを行う．これらの層における転送技術の進歩により，各コネクションのデータ転送の品質を上げること，コネクション間での品質の同一化および差別化が可能となる．

現在の TCP/IP を用いて，データ転送を行う場合のサービスの基本は，指定された 2 点間でベストエフォートによるサービスを提供することである．ベストエフォートサービスでは，所望の相手と通信するサービス（コネクティビティサービス）が基本サービスであり，他のサービスに関しては何の考慮もされていない．そのため，現在求められている高いサービス品質を提供するネットワークは，コネクションの転送帯域，誤り率，遅延などのサービス品質の要素を考慮し，サービスとして提供する必要がある [2]．ところで，ネットワークサービス品質とは，あるホスト上のアプリケーションが他のホスト上のアプリケーションに一連のパケットフロー*を送る時の平均レート，ピークレート，パケットの遅延時間，ジッター，パケット廃棄率などのことである．このように，パケットフローに対して指定された品質を保ちながら複数のフローを同時かつ経済的に転送する技術を QoS 技術といい，主にネットワーク層，トランスポート層におけるプロトコル，転送技術の性能の指標の一つとして，重要視されている．

QoS 制御によって実現可能なサービス品質のうち，平均レート，ピークレート，遅延時

*ネットワークに投入される時間的にまとまったパケットの集合でサービス品質的に同一の扱いを必要とするものという．例えばある送信ホストから受信ホストへの 1 つのコネクション上で送られるパケット全体は 1 つのフローを形成する．

間，ジッター，パケット廃棄率などがある．これらの指標は，ネットワークの構造だけで定まらずに，ネットワークに投入されるトラヒックにより左右されるという特徴がある．そのため，ネットワークの構造だけを与えてその品質を予測すること，また逆にネットワークのサービス品質を定量的に指定するだけでそれを満たすネットワークを設計することは困難とされてきた．しかし，最近の QoS 制御技術の進歩により，ネットワークの構造とネットワークに流入するトラヒックの双方に，ある種の制約をあたえると，ネットワークの動作は予測可能となり，サービス品質の保証ができるようになってきた．

QoS 制御によりサービス品質の提供が可能になることによる効果として，以下の点が挙げられる．

- サービス統合の実現

  今後ネットワーク上で多種多様なデータを転送するようになり，サービス統合のネットワークが求められる時代を迎えると，サービス品質を考慮しないベストエフォートサービスだけでは不十分である．このような局面において，QoS 制御できると，サービス品質を指定する，あるいは，数メニューの中から選択することによって，それぞれのネットワークアプリケーションに整合した品質を提供できるようになる．

- 新収益源の創出

  今後はキャリアや企業ネットワークは各種のサービス品質を準備してユーザに提供することになろう．このような場合は，ネットワーク内を QoS 制御することによって，ユーザの求めるサービス品質を要求に応じて提供可能となる．その結果，ネットワークサービスの新たな収益源，差別化要因となりうる．

- コスト削減

  ベストエフォートサービスを提供するネットワークでマルチメディア通信を行う場合には，過剰設計や選択的転送を採用せざるをえない．これに対して QoS 制御を導入すると，コネクション毎に指定されたサービス品質を提供すればよいため，ネットワーク全体の設備量を節減できる．

以上のように，サービス品質の制御は今後のネットワークの普及，展開に不可欠であり，QoS 制御の技術革新が進み，標準化も進められている [3].

ネットワークのサービス品質を利用する場合，ユーザはコネクションのサービス品質を指定できるが，そのサービスを受けるためには転送帯域やバースト長といったコネクションの特性をあらかじめ約束した範囲内に保つ義務がある．これは，ユーザの投入するコネクションのトラヒック負荷とネットワーク事業者の提供するサービス品質レベルは密接に関係しているからである．ネットワークサービスを利用する際に，ネットワークサービス提供者とユーザはその都度利用者が投入するトラヒック，利用者が希望するサービス品質レベル，事業者の提供できるサービス品質レベルをつき合わせて両者の納得いくサービス品質レベルを決定しなければならない．このようにしてユーザとネットワークサービス提供者が合意した内容をサービス品質レベル契約 (SLA: Service Level Agreement) と呼ぶ．SLA では少なくとも，サービス仕様とトラヒック仕様の双方を規定しなければならない．

1.2.2 Diffserv

近年のネットワークサービスの多様化，アプリケーションの多様化により，ユーザが要求する帯域の保証，遅延やジッターの上限値といったサービス品質に主題をおいた研究が数多くなされてきている．これまでにサービス品質を保証するフレームワークとして Intserv (Integrated Service) [4, 5] や Diffserv [6, 7, 8] などが提案されている．特に Diffserv を利用したサービス品質保証サービスはスケーラブルな転送サービスであるため注目されている．Diffserv の基本概念として，次の 3 点が挙げられる．

- IPv4 パケットの Type of Service フィールド (8 ビット)，または IPv6 パケットの Traffic Class フィールド (8 ビット) の上位 6 ビットに DS コードポイント (DiffServ codepoint) を記入しておき，ルータのトラヒック制御機構の動作をこのコードだけで決定する．
- エンドツーエンドのサービスを規定することはせずに，QoS を実現するフレームワークとその構成部品の一部を規定する．
- 性能向上のためのトラヒック調整機能をネットワークの入口，出口だけで行い，途中のコアルータはできるだけ簡単にして性能と拡張性の増大を図る．

Diffserv では，サービス品質を行う保証を行うコネクションに属するパケットには DS コードポイントが与えられ，同じ DS コードポイントを持つパケットは，コネクションにかかわらず同一のサービスを受ける．すなわち Diffserv の制御単位は必ずしもコネクションではなく，同一 DS コードポイントの持つフローの集合である．Diffserv では同一 DS コードポイントを持ち，同一リンクを流れるパケット全体を DS 動作集合 (DS behavior aggregate) と呼び，Diffserv の QoS 制御の単位となる．Diffserv 機能を持つルータが DS 動作集合に施すトラヒック制御操作が PHB (Per Hop Behavior) であり，EF (Expedited Forwarding) [9] と AF (Assured Forwarding) [10, 8] などが規定されている．

EF PHB[9] は専用線的サービスを実現するための PHB である．各ルータは，DS 動作集合に，必ず設定帯域以上のサービス帯域を割当てる．設定帯域は DS 動作集合ごとにオペレータが管理する．入力の DS 動作集合の入力帯域より設定帯域が大きい場合，パケットがこのルータで待たされることはほとんどないので，遅延，ジッター，パケットの廃棄率がごく小さい．したがって仮想専用線の実現などに使用可能となる．このルータでプライオリティスケジューリングなどを使用する場合には，ネットワークの入口において DS 動作集合のトラヒックのピークレートを厳密に制限する必要がある．オペレータは入口のピークレートおよび最低レート，バースト長を規定してもよい．

AF PHB[10] は Better than Best Effort の実現のための定性的 PHB である．あらかじめ，エッジルータでパケットに複数の優先度のマークを付けておき，ネットワークが輻輳した場合，優先度が低いパケットから順に廃棄して，優先度が高いパケットはできるだけそのまま転送する．すなわち AF PHB では，共通のサービス品質処理を受けるパケット

全体をクラスと呼び，1 個のクラスを複数のレベルに分類する．各レベルにそれぞれ 1 個の DS 動作集合を割当て，各レベルに応じた QoS 制御をネットワークで規定する．

上記の AF PHB を用いたサービスである Diffserv 帯域保証サービスは，一般的にホスト集合からの複数ホストへのデータ転送についてネットワーク内で締結され，転送帯域を平均的に保証する．エッジルータにおいて SLA を R Mbit/s とし，平均転送帯域が R Mbit/s 以下のときは適合パケットに，R Mbit/s を超えた場合には超えた分を非適合パケットにパケットがマーキングされる．またこのサービスは複数のドメインを通過する場合でも，各ドメインは一様に R Mbit/s の平均帯域を保証可能である．

1.2.3 TCP

TCP は，トランスポート層のプロトコルで，信頼あるデータ転送を行うサービス，つまり通信の開始から終了まで通信路の信頼性を保証したコネクション型のサービスを提供する [11, 12]．一方，TCP が下位層として使用する IP は，IP パケットを通信単位とするコネクションレス型のプロトコルであり，通信の信頼性，到達時間を保証する機構を持たない．そこで，TCP では，パケットの紛失，パケットの重複，パケット順序の逆転，パケット内のビット誤り，パケット遅延の変化などの問題に対処し，信頼性の高い通信サービスを提供する．

また，TCP は輻輳制御機構を実装したプロトコルである [13]．TCP の輻輳制御機構が提案された当時，ネットワークは輻輳崩壊[†]の被害を受けていた．TCP 輻輳制御機構は，TCP がネットワーク中の利用可能な帯域について調査し，確認応答 (ACK: Acknowledgement) を利用して新しいセグメントの送出帯域を調節するものであった．TCP は，パケットが廃棄されるまで，ネットワークに送出するセグメント数を増加させることによって，ネットワーク上で利用可能な帯域を調整する．そのため，パケットはネットワークの輻輳発生によって予告なく廃棄される可能性がある．パケットが廃棄された際には，利用可能帯域を越えたとして，TCP は適切な動作ポイントが未処理のセグメントの半分であることを決定し[‡]，送信セグメント数を半分に減少させ，利用可能な帯域の調節を行う．この TCP 輻輳制御機構は，確認応答ごとに送信セグメント数を 1 つ増加させ，パケット廃棄時には送信セグメント数を半分に減少させるため，AIMD (Additive Increase, Multiplicative Decrease: 加算的増加，指数的減少) アルゴリズムと呼ばれている．

その後，TCP の改良の提案 [14, 15] が行われ，現在は，TCP Reno[16] というバージョンが主に使われている．TCP Reno では，送信側 TCP は輻輳ウィンドウ ($cwnd$) とよばれる状態変数を維持する．輻輳ウィンドウは動的に調節され，所定の TCP のネットワークへ送信可能なセグメント数に反映される．TCP の輻輳ウィンドウ調節アルゴリズムに

[†]輻輳崩壊とは，ルータがバッファを使い果たしているのにもかかわらず，タイムアウトによってエンドホストがパケットを再送し，より深刻な輻輳を招いてしまう現象のことである．

[‡]輻輳ルータのバッファスペースが，その時点での”pipe”に存在するパケット数と同数であるとし，TCP の輻輳ウィンドウを半分にすることによって TCP の動作ポイントと pipe 内のパケット数を一致させ，ルータ中のキューサイズをゼロにすることを想定している．

は，指数的に輻輳ウィンドウを増加させるスロースタートフェーズと，線形的に輻輳ウィンドウを増加させる輻輳回避フェーズの，2つのフェーズがある．指数的増加フェーズにおいては，ACK が戻ってきた場合，*cwnd* は RTT (Round Trip Time: 往復遅延時間) 毎に倍増する[§]．線形的増加フェーズにおいては，ACK が戻ってきた場合，*cwnd* は線形的に RTT 毎に 1 MSS (Maximum Segment Size) 分だけ増加する．また，2つのスロースタートフェーズ，輻輳回避フェーズとも，パケットが廃棄された場合には，*cwnd* は半減する．

ネットワークでパケットが廃棄されたと判断する方法は以下の仕組みになっている．受信側ではパケットを受け取ると常に ACK を返す．そして，パケットが順序外に到着した場合，その前に来るべきパケットが到着するまで，送信側は最後に送った ACK を送り続ける．この 2 回以上送られる ACK を重複 ACK と呼ぶ．この重複 ACK を受け取った送信側では，3 つの重複 ACK を受け取るまでは，パケットの順序逆転とし，パケットの再送を行わない．その後，3 つ目の重複 ACK を受信した時点で，ネットワークでパケットが廃棄されたと判断して，パケットの再送を行う．

TCP は *ssthresh* (Slow-Start threshold) と呼ばれる状態変数も保持する．*ssthresh* は，TCP が指数的増加フェーズから線形的増加フェーズへ切り替える変換点を示す．*cwnd* が *ssthresh* を下回っている場合，TCP は *cwnd* を指数的に増加させるためにスロースタート機構を用い，*cwnd* が *ssthresh* を超えた場合，TCP は輻輳回避機構に切り替え，*cwnd* を線形的に増加させる．

TCP の動作範囲としては，TCP は輻輳回避機構のみを用い，パケット廃棄が生じるまで加算的増加フェーズを保ち，パケット廃棄の際には *cwnd* と *ssthresh* が以前のパケット廃棄時の *cwnd* の半分になるように再調整を行うのが理想的である．*cwnd* の新しい値が *ssthresh* の新しい値と等しいので，TCP は加算的増加フェーズに留まり続け，輻輳回避機構による制御を行う．

図 1.1 に TCP Reno の輻輳ウィンドウの時間変化を示す．TCP コネクションの開始時は，スロースタートフェーズであり，指数的に輻輳ウィンドウを増加させ，輻輳が生じパケットが廃棄されると，輻輳ウィンドウを半減させ，その後は，輻輳回避フェーズに遷移し，ACK 受信時は線形的な輻輳ウィンドウの増加を行い，パケット廃棄時は輻輳ウィンドウの半減させる．

1.3 研究の目的と本研究の位置付け

1.3.1 研究の目的

現在，Diffserv と TCP に関して様々な技術の提案され，その技術により高いサービス品質の提供が可能となってきている．しかし，近年のアプリケーション種数の増加，ユーザ数の増加に伴い，帯域が広く，伝播遅延が大きい帯域遅延積の大きなネットワークにな

[§]RTT はパケットの往復遅延時間を示しており，伝播遅延，ノードの処理遅延とバッファリング時のキューイング遅延の和で計算される．TCP では計測された rtt の値を用い，RTT の推定値 (RTT_{new}) を，$RTT_{new} = (\alpha \times RTT_{old}) + \{(1-\alpha) \times rtt\}$ の計算式から算出する．

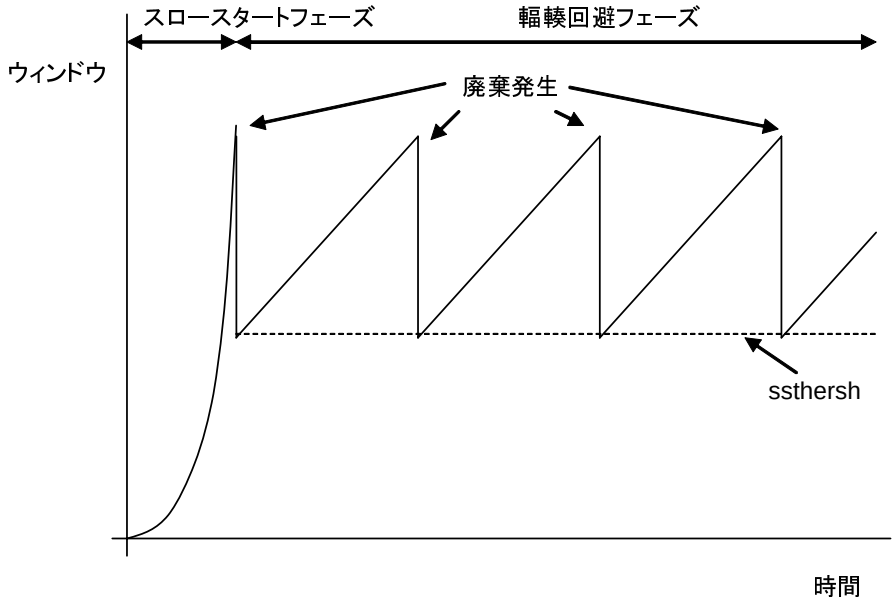

図 1.1: TCP Reno における輻輳ウィンドウの時間変化

りつつある．このように，ネットワークの使用帯域が拡大し，伝播遅延が増大した現在においても，高いサービス品質を維持してデータ転送をすることが重要であり，これが現在のネットワークが抱える課題である．

本研究の目的は，帯域遅延積の大きなネットワークにおいて，ネットワーク内のコネクションに対して高いサービス品質を提供可能な帯域制御技術を提供することである．本研究では，(1) 帯域遅延積の大きなネットワークが抱える帯域制御技術に関する課題に対して，RTT が異なることによって生じるコネクション間の公平性の課題と，(2) 帯域が拡大し，伝播遅延が増大することによって生じる効率の課題について考察を行い，課題を解決する提案を行う．伝播遅延の増大によって引き起こされる課題に対しては，ネットワークの改善，エンドホストとネットワークの連携による改善に焦点を絞り，解決を図る．帯域の拡大，伝播遅延の増大によって引き起こされる課題に対して，エンドホストの改善に焦点を絞り，解決を図る．

1.3.2 解決する課題

本研究では，以下の課題を解決する．

- コネクション間の公平性の課題

  ネットワーク内の伝播遅延が大きくなることによって，集約された各コネクションの RTT が異なる場合に，各コネクションに割当てられる帯域が不公平になる．

- 帯域使用効率の課題

  ネットワークの遅延の増大，広帯域化が進み，コネクションの輻輳ウィンドウ，RTT が増大すると，そのコネクションは帯域を効率よく使用できない．

コネクション間の公平性の課題に関して，Diffserv 帯域保証ネットワークにおける適用に焦点を絞り，考察を行う．Diffserv 帯域保証ネットワークでは，同一のクラスのコネクションの集合に対して同一のサービスを提供し，集合としてのコネクションに対して帯域を保証する．このとき，集約されたコネクションに対して保証された帯域は，それぞれのコネクションに公平に割当てられることが望ましい．また，契約帯域を超えて使用可能な帯域については，帯域を獲得可能なコネクション間で公平に帯域を分配しあうことが望ましい．しかし，集約されたコネクションのそれぞれのコネクションの RTT が異なっている場合，TCP Reno の輻輳制御機構の性質上，RTT の短いコネクションが長いコネクションよりも輻輳ウィンドウの増加量が大きいため，結果的に RTT の短いコネクションが帯域を多く獲得し，公平性が失われる [17]．その結果，Diffserv 帯域保証ネットワークで，集約されたコネクション間の帯域量を公平に割当てることが困難になる．

帯域使用効率の課題に関しては，一般的なネットワークでの適用での使用を前提とし，考察を行う．帯域遅延積の大きなネットワークで TCP 転送を行い，コネクションの輻輳ウィンドウおよび RTT が増大した場合，TCP Reno の輻輳制御機構の性質上，輻輳ウィンドウが増減する周期が長くなる．結果として，輻輳ウィンドウがネットワークの帯域に達しない状態が長く続き，帯域を効率よく使用できなくなる．

1.3.3 課題の解決方法

本研究では，ネットワークの遅延の増大，広帯域化に伴い発生する課題について，RTT の異なるコネクション間の公平性と帯域使用効率の課題を分けて議論する．RTT の異なるコネクション間の公平性の課題を解決するために，RTT の異なる TCP コネクションが生じやすい各コネクションの RTT が異なる状況になりやすい遅延が増大しているネットワークにおいて考察を行う．また，帯域使用効率の課題を解決するために，帯域が広く，伝播遅延が大きいネットワークにおいて考察を行う．これら 2 つの課題に対して，図 1.2 に示す 3 つの視点から考察を行う．(1) ネットワークの改善によるアプローチとは，ネットワークにおけるデータ転送ポリシーの改善を行うことで，主にデータリンク層，ネットワーク層のプロトコルが該当する．本研究では，Diffserv 網の帯域保証機能に関して，エッジルータのパケットマーキングポリシーを提案する．(2) エンドホストとネットワークルの連携による改善アプローチとは，ネットワークの使用状況に応じて，コネクションの使用帯域を決定し，エンド端末の帯域制御機能に反映することである．本研究では，Diffserv 網で算出した各コネクションの使用帯域を実現するエンド端末の TCP 輻輳制御機構を提案する．(3) エンドホストの改善によるアプローチとは，ネットワークの状況を考慮して，パケットの送出帯域，送出方法を改善を行うことで，主にトランスポート層とその上位層のプロトコルが該当する．本研究では，TCP 輻輳制御機構の提案を行い，パ

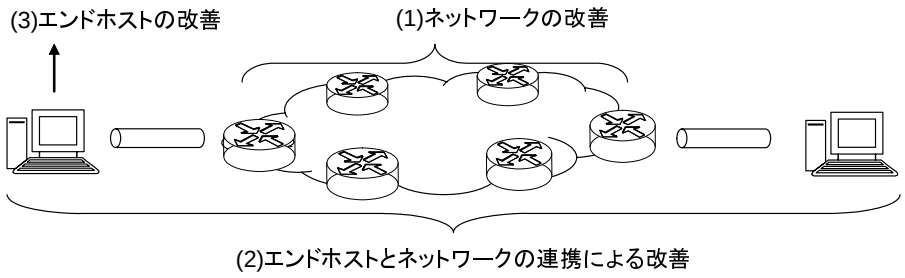

図 1.2: 課題に対する 3 つの解決アプローチ

表 1.1: 効率的かつ公平な帯域獲得を実現するネットワークおよびエンドホストにおけるアプローチ

	ネットワークの改善	エンドホストとネットワークの連携による改善	エンドホストの改善
公平性の課題	3 章において提案	4 章において提案	関連研究の紹介 (2.6.1 節)
帯域使用効率の課題	実現困難	関連研究の紹介 (2.6.3 節)	5 章において提案

ケットの到達確認，廃棄を元に，送出帯域の設定方法を決定する方法を提案する．

1.3.4 本研究の位置付け

本研究では，2 つの課題に対して 3 つの解決アプローチを用い，多面的に課題の解決を図る．また，公平性の課題，帯域使用効率の課題に対して，各アプローチの性質を述べる．

ネットワークの改善による解決アプローチでは，ネットワークのルータ，または，ポリシーサーバの機能向上を行い，課題を解決する．このアプローチは，エンドホストの対応が必要なく，エンドホストのプロトコルの違いによる不公平性を回避することができる．また，Diffserv 網のような帯域保証ネットワークでは，エッジノードでネットワーク内の帯域制御ポリシーを決定する方式であるため，エッジノードのみの変更で課題の解決が可能となる．しかし，コアルータの設備投資は回避できるものの，エッジルータの設備を置き換える必要があり，設備投資にコストがかかるという欠点がある．

エンドホストとネットワークが連携することによる解決アプローチでは，ネットワークのルータ，または，ポリシーサーバの機能向上だけでなく，エンドホストの帯域制御技術機能の向上を行い，課題を解決する．このアプローチは，エンドホストとネットワークの双方で改善を行うため，より公平で，より効率的に帯域制御を実現できると見込まれる．

一方で，ネットワークルータおよびエンドホストの設備を置き換える必要があり，ネットワークの改善によるアプローチ以上に，設備投資にコストがかかる．また，本アプローチは，エンドホストおよびネットワークのプロトコルを変更するため，既存プロトコルと共存可能な方式であることが望ましい．

エンドホストの改善による解決アプローチでは，エンドホストの帯域制御技術機能の向上のみで，課題を解決する．このアプローチは，ネットワークルータ等のネットワークの既存設備を使用可能であり，エンドホストの機能向上で実現できるため，スケーラビリティが高いという特徴がある．一方で，既存の TCP 輻輳制御機構との親和性を維持する必要がある．

表 1.1 に遅延の増大によって引き起こされる課題，広帯域化によって引き起こされる課題に関して，3 つの解決アプローチと，記述する章，または，節の番号を示す．

遅延の増大によって引き起こされる課題に対して，Diffserv 網の改善 (3 章) ，TCP 輻輳制御機構および Diffserv 網の連携による改善 (4 章) により，課題の解決を図る．コネクション間の公平性の TCP 輻輳制御機構による改善はすでに検討が行われているため，関連研究として紹介を行う (2.6.1 節) ．また，2.6.1 節において．関連研究と，ネットワークにおける Diffserv 網側の改善，TCP 輻輳制御機構および Diffserv 網の連携による改善との違いを考察する．

広帯域化によって引き起こされる課題は，TCP 輻輳制御機構と密接に関係しており，その性質上，TCP 輻輳制御機構と関係しないネットワークのみにおける改善では実現が困難である．効率的かつ公平な帯域獲得を実現するアプローチとして，TCP 輻輳制御機構における改善，TCP 輻輳制御機構およびネットワークルータの連携による改善が挙げられる．本論文では，ネットワークにおける TCP 輻輳制御機構の改善 (5 章) により，課題の解決を図る．広帯域ネットワークにおける TCP 輻輳制御機構かつネットワークの連携による改善は，関連研究として紹介を行う (2.6.3 節).

最後に，提案した解決アプローチ，関連研究を元に，公平性の実現，効率的な転送を実現する技術の適用方法について考察する．

1.4 本論文の構成

本論文の構成は以下の通りである．

2 章において，ネットワークにおける課題を明らかにし，その課題に対する関連研究を紹介する．ユーザ間公平性の課題に関して，3 章において，ネットワークにおける解決として Diffserv 網のエッジルータのマーキングポリシーの提案を行い，4 章において，ネットワークとエンドホストの連携による解決として，Diffserv 網と TCP 輻輳制御機構が連携して公平な帯域を実現する方式の提案を行う．また，広帯域化における効率の課題，ユーザ間公平性の課題に関して，5 章において，エンドホストのみの解決として，TCP 輻輳制御機構の輻輳回避フェーズの提案を行う．最後に，6 章で各解決アプローチの考察を行い，結論を述べる．

第2章　帯域制御技術が抱える課題

2.1 まえがき

本章では，効率的かつ公平なデータ転送を実現するために必要な帯域制御技術にかかわる課題を明らかにする．

2.2 帯域保証ネットワークにおける現状

2.2.1 集約されるコネクション

ネットワークで帯域保証サービスを実施する形態を考えた場合，その最も単純なものとしては，1 つの送信元と 1 つの受信先との間で帯域を保証する 1 対 1 帯域保証サービスが挙げられる [18]．この場合，契約に集約された各コネクションの RTT にそれほど差は生じないと考えられる．しかし，帯域保証サービスには 1 対 1 の接続だけでなく，1 つの送信元から複数の受信先へのデータ転送に対してその帯域を一括して保証するようなモデルや，1 つの送信元からのあらゆるデータ転送に対してその帯域を保証するといったモデルの 1 対 n 型接続サービスが提案されている．これらのサービスは 1 対 1 で帯域保証契約を結ぶサービスを複数適用する形態に似ているが，契約にかかるコストや帯域の効率利用を考えた場合，1 対 n 型接続サービスの方がより効率的であるといえる．より具体的にはこのような帯域保証サービスは,

- 個人が特定の複数の相手とのデータ転送に対して帯域保証を求める場合
- 企業が複数の支社間におけるデータ転送に対して帯域保証を求める場合
- 不特定多数のユーザに対してファイルを配布する FTP サイトがそのデータ転送に対して帯域保証を求める場合

といった場面で用いられると予想される．

このような帯域保証サービスの場合，RTT の異なる複数のコネクションがネットワークにおいて 1 つの SLA に集約されると考えられる．特に図 2.1 に示すように，複数のネットワークを経由するコネクションは，一方が同一のネットワークを経由し，他方は異なるネットワークを経由する場合があり，このとき，ネットワーク A で集約されたコネクションは RTT の大きく異なるコネクションで構成されることが予想される．

2.2.2 増大する遅延，拡大する帯域

現在のネットワークは，遅延が増大し，帯域の広帯域化が進み，帯域遅延積が大きくなってきている．

遅延が増大する背景として，ネットワークのグローバル化，衛星を経由する通信が挙げられる．ネットワーク間で利用されるアプリケーションの中で，帯域保証ネットワークに

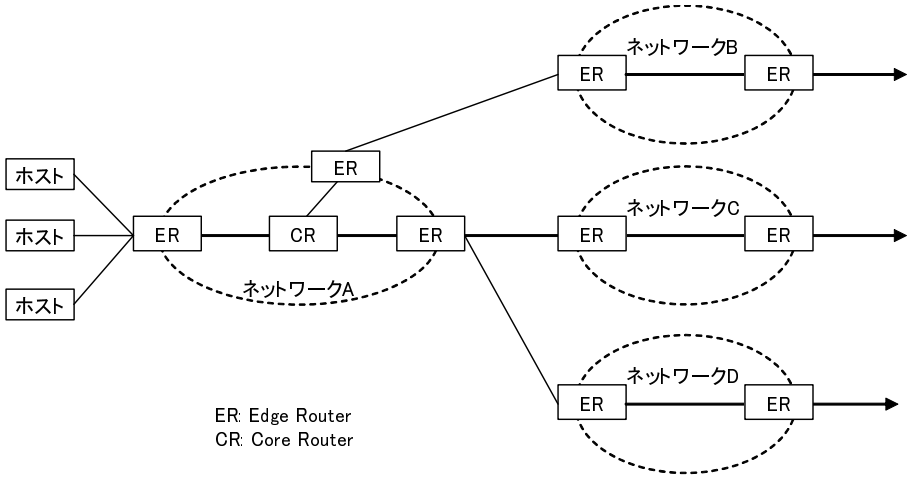

図 2.1: 1 対 n 型接続サービスの例

おいて用いられるアプリケーションとして，映像配信，会議システム，ファイル転送ソフト，遠隔操作システムなど多く挙げられる．このようなアプリケーションを 1 対 n 型接続サービスで使用し，複数の拠点で相互に通信する場合には，拠点が海外におかれるケース，拠点間を衛星通信を用いて転送するケースも想定される．このような場合，帯域保証ネットワークを使用するコネクションはその RTT が大きくなり，結果として集約されるコネクションの RTT が徐々に分散してきている．

帯域が広帯域化する背景として，ユーザ数の増加，ユーザあたりの転送帯域の増加が挙げられる．さらに，光ファイバ転送技術，転送装置の技術革新により，より安価に広帯域転送が実現できる技術的背景も広帯域化を促進している．これらの背景により，ネットワークの帯域も年々広帯域化が進んできている．そのため，1 ユーザが一時的に使用可能な帯域は，格段に増加している．ビジネスユーザにおいても，広帯域転送が必要なネットワークアプリケーションの出現により，よりビジネスユーザが帯域保証サービスを必要とし，帯域保証ネットワークを使用する帯域量も大きくなってきている．このように，帯域転送技術の発展により安価なサービスが実現され，より帯域が拡大してきている．

2.3 要求条件

上記で示した，帯域保証ネットワークの状況においても，高いサービス品質を提供することがネットワークに求められている．QoS 制御によって適用可能なサービスの品質の要素として，低遅延，低ジッター，公平性，帯域使用効率，低ロス等多く挙げられる．本研究では，遅延の増大，帯域の拡大に大きくかかわる要素として，以下の 2 つを取り上げる．

- 帯域使用効率

  ネットワークの帯域を効率よく使用し，保証する帯域，どのコネクションも使用し

ていない帯域をユーザに十分に提供可能なことが望ましい．また，帯域使用効率を上げることで，過剰なネットワーク設計を回避可能になり，設計コストの削減を行える．

- 公平性

  ネットワークを使用するコネクションは，コネクション間で公平にネットワークの帯域を享受することが望ましい．コネクションの RTT が異なる場合，TCP 等のプロトコルが異なる場合，転送開始時刻が異なる場合，公平性が損なわれることがあるが，公平に帯域を割当てることが望ましい．本研究では，特にコネクションの RTT が異なる場合，TCP のバージョンが異なる場合に着目し，公平性の改善を行う．

他のサービス品質の要素の，低遅延，少ないジッター，誤り率等に関しては，スイッチの性能，伝播する媒体の性能に依存する部分が大きく，本研究で対象としてる遅延の増大，帯域の拡大によって大きく影響する部分で無いため，本研究では考察の範囲外としている．

以下の本章では，拡大する帯域，分散する伝播遅延の帯域保証ネットワークにおいて，公平性，帯域使用効率の観点から指摘されている課題，(1) 異なる RTT のコネクション間の公平な帯域割当て，(2) 広帯域ネットワークにおける帯域使用効率の向上を説明する．

2.4 異なる RTT のコネクション間の公平な帯域割当て

Diffsev 帯域保証サービスの契約形態は，契約締結にかかる時間や手間などのコストを考慮した場合，個人，あるいは企業のような団体毎に 1 つの SLA を締結する形が一般的であると予想される．この場合，1 つの SLA には複数のコネクションが集約され，集約された各フローはネットワークが保証する帯域を分け合う形でサービスを享受することになる．ここでコネクション間の公平性を考慮した場合，集約内の各フローは同一のサービスを受けていることから，それぞれのコネクションは保証された帯域を公平に分け合うことが望ましい．すなわち，集約中の各コネクションはそのような公平な帯域量以上の速度で転送が行われることが望ましい．集約される各コネクションが全て同じ条件で転送される場合には，SLA によって保証される帯域は各コネクションに均等に割当てられることが可能となる．しかし，実際には集約されている全てのコネクションの転送条件が全く同じであることは稀であり，多くの場合条件の異なるコネクションが集約される形で転送が行われるため，その結果として各コネクションに割当てられる帯域量には差が生じる場合がある．そのように異なる転送条件としては様々なものが考えられるが，文献 [17] では，コネクション間の帯域割当ての公平性に大きな影響を与える条件として，コネクションの往復遅延時間である RTT が指摘されている．

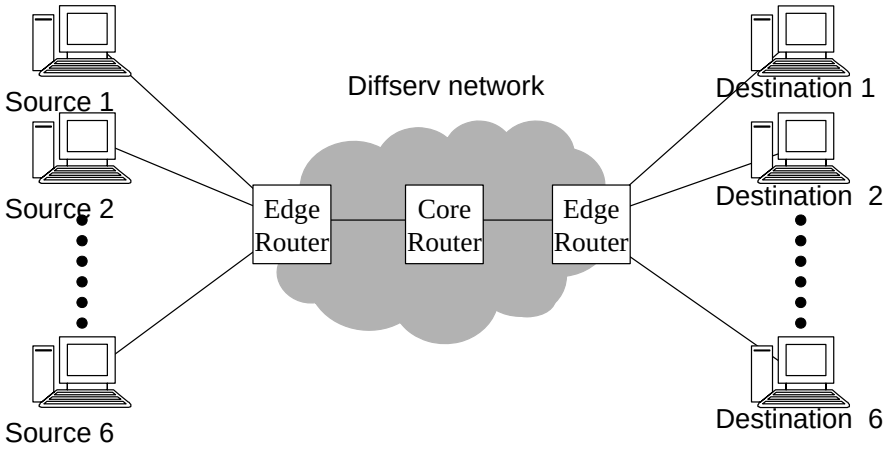

図 2.2: シミュレーションモデル

表 2.1: シミュレーション条件

契約帯域量	9 Mbit/s
集約内のコネクション数	6 (固定)
FR	1.5 Mbit/s
RTT	40ms, 80ms, 120ms, 160ms, 200ms, 240ms
TCP 輻輳制御機構	全て TCP Reno を使用

2.4.1 Fair Rate

本研究では集約コネクション内の各コネクションが達成すべき公平な転送帯域を Fair Rate (FR) と名付け，以降で用いるものとする．FR は契約帯域量と 1 つのサービスに集約されているコネクション数を用いて以下のように定義する．

$$FR = \frac{\text{契約帯域量}}{\text{集約コネクション数}} \tag{2.1}$$

2.4.2 不公平な帯域割当ての一例

RTT の異なるコネクションが集約された場合に発生する不公平な帯域割当てについて，コンピュータシミュレーションを利用した具体例を示すことで説明する．

シミュレーション環境を図 2.2 に，シミュレーション条件を表 2.1 に示す．

以上のような条件でシミュレーションを実行し，ネットワーク内における各コネクションのスループットを測定した結果を表 2.2 に示す．

表より，集約コネクション全体のスループットは契約帯域量を上回っているが，集約内の各コネクションに注目した場合，相対的に RTT の小さなコネクションほどそのスルー

表 2.2: 測定結果

Flow #	RTT (ms)	FR (Mbit/s)	スループット (Mbit/s)
1	40	1.500	3.067
2	80	1.500	2.085
3	120	1.500	1.596
4	160	1.500	1.273
5	200	1.500	1.111
6	240	1.500	0.851
Total		9.000	9.983

プットは大きくなっている．また各コネクションのスループットと FR とを比べた場合，RTT の大きなコネクションは達成されるべき FR を下回るスループットしか達成できていないことがわかる．これは，コネクション間の公平性の点から不公平な状態であると考えられる．

2.4.3 不公平な帯域割当てが生じる原因

不公平な帯域割当てには，データの送信速度をネットワークの状態に応じて調節する TCP の輻輳制御機構が大きな影響を与えている．この問題の原因として TCP の輻輳制御が関係する次の 2 点が挙げられる．

原因の 1 つとして，TCP 輻輳ウィンドウの増加速度が各コネクションによって異なることが挙げられる．TCP Reno の場合，正常転送時に輻輳ウィンドウを 1 RTT あたり 1 MSS の割合で増加させる．そのため，RTT はウィンドウ増加速度に多大な影響を与えており，RTT の長いコネクションほどウィンドウの増加速度が遅くなることが分かる．そのため集約コネクション中で相対的にウィンドウ増加速度の遅い TCP コネクションはネットワークの状態を速やかに反映した帯域調節を行うことができず，ウィンドウ増加速度の速いコネクションに利用可能な帯域を多く奪われてしまう．その結果として，RTT の異なるコネクションが同一ネットワークで帯域を共有する場合，割当てられる帯域量に不公平が生じる．

またもう 1 つの原因として，ルータにおける FR を考慮しないマーキングとパケット廃棄の影響が挙げられる．Diffserv 網は集約コネクションに対して AF PHB を適用することで契約帯域量分の速度での転送を保証することは可能だが，集約内の各コネクションに関しては何の保証も行わない．これは，より具体的にはエッジルータにおけるマーキングの動作と関係する．エッジルータではコネクション単位ではなく契約単位（集約コネクション単位）でパケットにマーキングを行うために，集約コネクション中のパケットは集約コネクション全体の帯域に応じてマーキングが行われる．そのため，集約全体の帯域が契約値を超過した場合には，FR に満たない速度で送信を行っているコネクションのパケット

に対しても OUT マーキングが行われる場合があり，そのようなパケットがネットワーク内で廃棄された際には TCP の輻輳制御機構によって送信帯域が低下してしまう．このようなパケット廃棄を契機とする送信帯域の調節は TCP 輻輳制御機構の重要な機能の 1 つではあるが，送信帯域を下げることによって空いた帯域は，RTT が短くウィンドウの増加速度が速いコネクションがより多く獲得し，帯域割当て量に不公平が生じることが予想される．そのため集約コネクション内の公平性の観点からは，FR を考慮しないマーキングやパケット廃棄に伴う送信帯域の低下は，不公平を増大させる原因の 1 つであると考えられる．

以上説明したように，集約内の不公平な帯域割当て問題には TCP 輻輳制御機構が大きな影響を与えており，既存の帯域制御技術では問題を解決することができない．

2.5 広帯域化ネットワークにおける帯域使用効率の向上

2.5.1 パケット廃棄率 p とリカバリ時間 τ

本節では，パケット廃棄率 p と平均輻輳ウィンドウ $\overline{W}$ の関係式およびその式から導かれるリカバリ時間 τ の式をもとに，TCP の各方式について議論する．ここで，輻輳ウィンドウとは，TCP 送信者が制御している，ACK を受信せずに送信可能なパケット数のことであり，単位はバイトである．本論文では輻輳ウィンドウをセグメント単位で考察を行うため，輻輳ウィンドウを W (セグメント) で表現する．TCP は輻輳がない場合は W を増加させ，輻輳を検出すると W を減少させて輻輳を回避する仕組みをもつが，ネットワークの使用可能帯域に変化がない場合，周期的に同じ挙動を繰り返す．本論文ではこれを定常状態と呼ぶことにし，この定常状態における輻輳ウィンドウの平均値を平均輻輳ウィンドウ $\overline{W}$（単位はセグメント) と呼ぶ．

パケット廃棄率 p は，送信パケット数に対する，輻輳によるパケット廃棄数の割合を示し，ネットワークの特性を表す指標である．例えば，$p = 10^{-6}$ のとき，パケットを 10^6 個送るごとに輻輳状態に達し，パケットが 1 つ廃棄されることを意味する．TCP の各輻輳制御方式において，この p における $\overline{W}$ が一意に決まる．すなわち，この p と $\overline{W}$ の関係が，TCP の性質を表すものとなる．これは，定期的に生じるパケット廃棄よって，適切な輻輳ウィンドウを決定する TCP の性質によるものである．

そのような TCP では，使用可能帯域量が変動した状態において，パケットが廃棄される間隔を短く保つことで，より最近の輻輳の情報を入手し，輻輳ウィンドウを適した値で推移させることが可能となる．このとき，p の値が高くなる．パケット廃棄の間隔が短すぎる場合，輻輳が起き過ぎてしまうため，輻輳ウィンドウを適した値に推移させ，効率の良い転送を行うためには，輻輳が起き過ぎない程度にパケット廃棄の間隔を短く保つことが必要となる．一方，広帯域転送を行うためには W を大きくし，一度に大量のパケットを転送する必要がある．以上より，$\overline{W}$ と p の関係式において，$\overline{W}$ が大きいときであっても，輻輳が起き過ぎない程度に p の値を高く維持できるとき，広帯域転送時において効率

のよい転送を行っているといえる．

また，輻輳制御時のウィンドウの増加量および減少量より，1 度減少させた W を再び同じ大きさまで回復するのにかかる時間 τ を求めることができる．本論文ではこの τ をリカバリ時間と呼ぶことにする．この τ の値が小さいほど，短時間で輻輳状態に達することとなり，前述のとおり，帯域を無駄なく利用できているといえる．

2.5.2 広帯域転送の転送効率

2.5.2.1 TCP Reno の問題点

TCP ではネットワークの輻輳に応じて，輻輳ウィンドウを変動させてネットワーク状況に最適な転送を行おうとしている．そこで，本節では具体的に TCP Reno がどのように振る舞い，どのような性質かあるかを検討する．

TCP には様々なバージョンがある [19, 20, 21] が，本研究では現在標準的に利用されている TCP Reno に焦点を絞って考察をする．TCP Reno の輻輳制御機構 [16] を式 (2.2), (2.3) に示す.

$$W = W + \frac{1}{W} \qquad \text{(ACK 受信時)} \tag{2.2}$$

$$W = W - \frac{1}{2}W \qquad \text{(パケット廃棄時)} \tag{2.3}$$

TCP Reno はネットワークの輻輳状態をパケットの到着の可否のみで推測し，輻輳が生じるまでは輻輳ウィンドウを 1 RTT あたり 1 MSS ずつ増加させ，輻輳が生じたら輻輳ウィンドウを半減させる動作を行う．TCP ではパケットを転送し，そのパケットに対する ACK が戻るまでの間に，1 RTT が経ち，ACK の受信数は輻輳ウィンドウのセグメント数となる．そのため，1 ACK 受信あたりの増加を W 回行うことで 1 RTT あたりの輻輳ウィンドウの増加が計算できる．本研究で扱う TCP においては ACK 受信における W の増加量の変化が十分に小さいため，[1 ACK 受信あたりの増加量]×[輻輳ウィンドウ] で 1 RTT あたりの輻輳ウィンドウの増加量が計算できる．TCP Reno の場合，1 ACK 受信あたり $1/W$ セグメントの増加を行うため，1 RTT あたり 1 セグメントの増加となる．このときの送信セグメント数に対する輻輳によるパケット廃棄の割合，パケットの廃棄率 p と平均輻輳ウィンドウ $\overline{W}_{reno}$ の関係を表した式を式 (2.4) に，パケット廃棄率 p とスループット T_{reno} の関係を表した式を式 (2.5) に示す [22]．このときのセグメント長を $Size_{seg}$ とする．

$$\overline{W}_{reno} = \frac{1.22}{p^{0.5}} \tag{2.4}$$

$$T_{reno} = \frac{Size_{seg}}{RTT}\frac{1.22}{p^{0.5}} \tag{2.5}$$

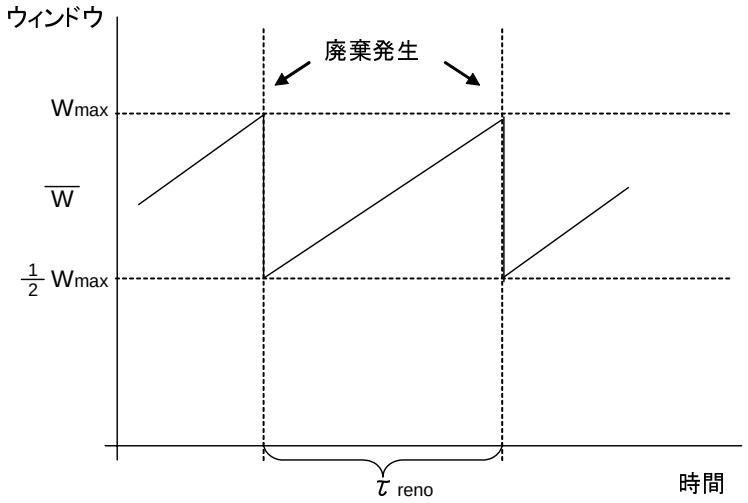

図 2.3: $\overline{W}$ と W_{max}

次にリカバリ時間を示す．廃棄によって生じるウィンドウの減少量を，τ 時間かけて減少直前の輻輳ウィンドウに回復するため，リカバリ時間 τ は，(ウィンドウ減少量) = (ウィンドウ増加速度) $\times \tau$ の式で表せる．

また，W が最大 (W_{max}) のとき，式 (2.3) より，輻輳制御による輻輳ウィンドウの減少量は $\frac{1}{2}W_{max}$ であり，式 (2.2) より，輻輳ウィンドウの増加量は 1 RTT あたり 1 セグメントであるから，

$$\begin{aligned} \tau_{reno} &= \frac{\text{ウィンドウ減少量}}{\text{ウィンドウ増加速度}} = \frac{\frac{1}{2}W_{max}}{\frac{1}{RTT}} \\ &= \frac{1}{2}W_{max}RTT \end{aligned} \tag{2.6}$$

となる．ここで，図 2.3 に示すように，$\overline{W} = \frac{3}{4}W_{max}$ より，

$$\tau_{reno} = \frac{1}{2}\Big(\frac{4}{3}\overline{W}\Big)RTT = \frac{2}{3}\overline{W}\ RTT \tag{2.7}$$

となる．

TCP Reno は，パケット廃棄を検出した際に輻輳ウィンドウを 1/2 と大きく減少させるのにもかかわらず，その増加量が RTT ごとに 1 セグメントと非常に小さいため，ネットワークの帯域を十分使う程度の輻輳ウィンドウまで回復するには時間を要する．ネットワークの帯域にあった輻輳ウィンドウに回復するまでの時間が長い場合，変動する帯域にも柔軟に対応できず，ネットワークの帯域を有効に利用することが困難である．例えば，$Size_{seg}$=1500 byte，RTT=100 ms で 10 Gbit/s の転送を TCP Reno で行おうとするとき，式 (2.5) より，パケット廃棄率が 2×10^{-10} に，リカバリ時間が 5500 秒になる必要がある．これは，2×10^{10} パケットを転送して初めて，ネットワークの帯域を十分利用できる輻輳

ウィンドウに達し，輻輳が生じることを示している．そのため，輻輳ウィンドウが回復するまでのリカバリ時間が非常に長くかかり，転送効率がよくない．

輻輳ウィンドウをネットワークの帯域に適した値で推移させ，効率のよい転送を行うためには，パケット廃棄に対する輻輳制御によって減少した輻輳ウィンドウを，リンク帯域にあった大きさに回復する時間を輻輳が起き過ぎない範囲で短くすることが必要である*．この時間を短くすると，輻輳によって生じるパケット廃棄の周期も短くなり，結果的に廃棄率が高くなる．その結果，広帯域データ転送下の大きな $\overline{W}$ においても，p を高く維持でき，効率のよい転送が可能となる．

2.5.2.2 広帯域転送のための TCP（HSTCP および STCP）

HSTCP では，p，輻輳ウィンドウ ($\overline{W}_{hs}$) とスループット (T_{hs}) の間の関係は式 (2.8)，式 (2.9) で示される．

$$\overline{W}_{hs} = \frac{0.12}{p^{0.835}} \tag{2.8}$$

$$T_{hs} = \frac{Size_{seg}}{RTT}\frac{0.12}{p^{0.835}} \tag{2.9}$$

前述のとおり，式 (2.9) より，$Size_{seg} = 1500$ byte，RTT = 100 ms で 10 Gbit/s の転送時，1×10^{-7} のパケット廃棄率である．この結果，文献 [23] より，10 Gbit/s 転送の際のリカバリ時間 τ は 123 RTT となる．TCP Reno の場合は，式 (2.7) より $\tau = 55333$ RTT であったことから，HSTCP は高帯域のリンクを利用する場合においても，より短時間で輻輳状態に達することが可能となる．したがって，HSTCP は高い廃棄率で大きい輻輳ウィンドウを実現でき，広帯域データ転送下でも効率のよい転送が可能となる．

また，STCP では，p，輻輳ウィンドウ ($\overline{W}_{sca}$) とスループット (T_{sca}) の間の関係は式 (2.10)，式 (2.11) で示される．

$$\overline{W}_{sca} = \frac{0.07}{p} \tag{2.10}$$

$$T_{sca} = \frac{Size_{seg}}{RTT}\frac{0.07}{p} \tag{2.11}$$

また，これより，リカバリ時間 τ は式 (2.12) のように示される [24]．

$$\tau_{sca} = 13.42RTT \tag{2.12}$$

*効率のよい転送を行う手段として，十分な輻輳ウィンドウに達するまでの輻輳ウィンドウの積分値を大きくすることも考えられる．この点に関しては別途考察が必要であるため，本研究では回復する時間のみに着目した．

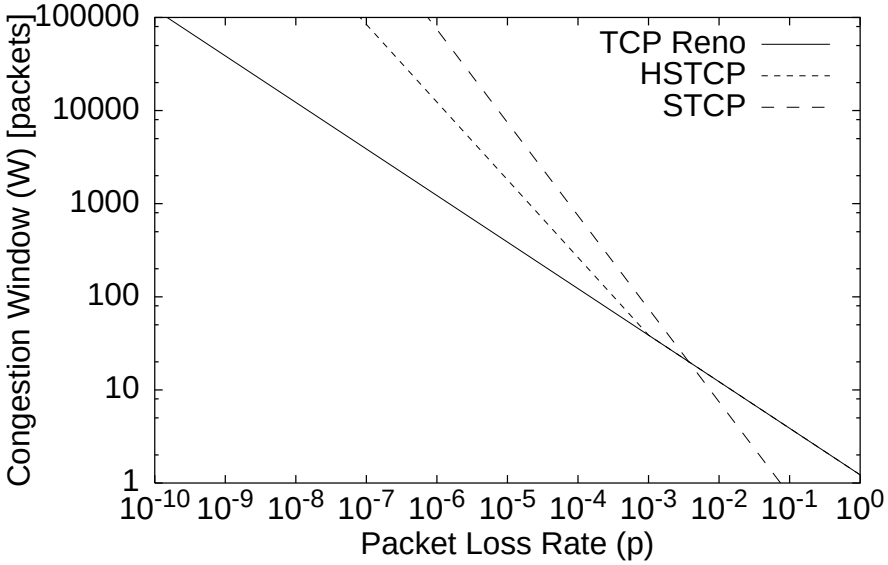

図 2.4: TCP Reno，HSTCP と STCP の輻輳ウィンドウと廃棄率の関係

式 (2.8)，式 (2.10)，式 (2.12) より HSTCP および STCP は TCP Reno と比較してネットワークの高廃棄率の場合においても，リカバリ時間が短く，広帯域転送可能な輻輳ウィンドウを維持できる輻輳制御方式である．

2.5.2.3 輻輳ウィンドウ，スループットとパケット廃棄率の関係

TCP Reno，HSTCP と STCP の輻輳ウィンドウと廃棄率の関係を図 2.4 に示す．図 2.4 はあるネットワークの廃棄率におけるその時の輻輳ウィンドウを表しており，同廃棄率の時，輻輳ウィンドウが大きいコネクションは帯域を多く獲得することができることを示している．図 2.4 より，廃棄率が大きい場合でも，平均輻輳ウィンドウが高く維持できる HSTCP と STCP は広帯域転送に向いている TCP 輻輳制御方式であることが確認できる．一方，TCP Reno では，廃棄率が小さくならないと平均輻輳ウィンドウが高くならず，広帯域転送には向いていない点も確認できる．

TCP Reno，HSTCP と STCP のスループットと廃棄率の関係を図 2.5 に示す．ここで，$RTT = 100msec$ とする．図 2.5 より，HSTCP と STCP はネットワークの廃棄率が高い場合においても，高いスループットを維持できることを示しており，広帯域ネットワークにおいて効率のよい転送を実現できている．しかしながら，HSTCP と STCP は TCP Reno と同じリンクを共有した際，HSTCP，STCP と TCP Reno の間に大きなスループットの差があるために，TCP の輻輳制御方式の違いによる不公平な帯域の分配が懸念される．

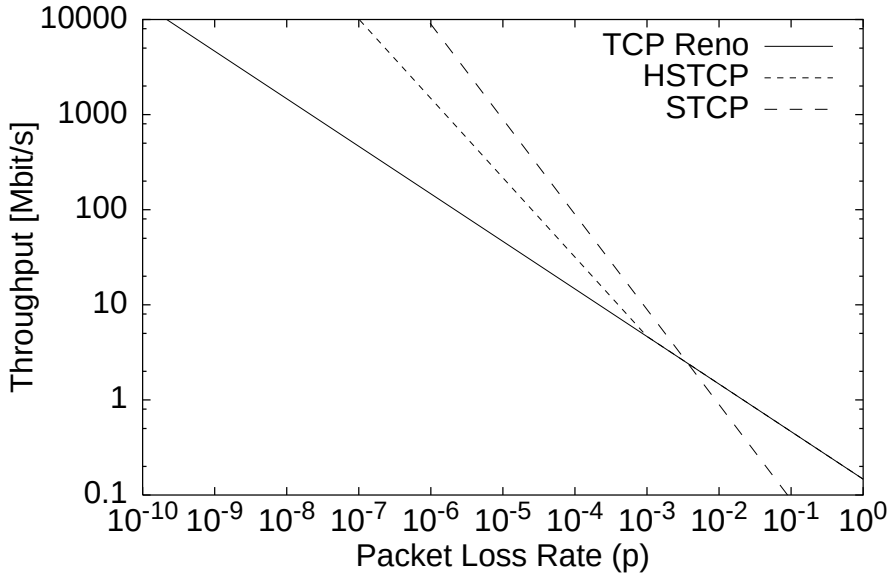

図 2.5: TCP Reno，HSTCP と STCP のスループットと廃棄率の関係（$RTT = 100msec$ のとき）

2.5.3 TCP Reno との親和性

2.5.3.1 HSTCP における TCP Reno との親和性

HSTCP は高帯域のリンクを利用する場合でも，輻輳状態に達しやすくするように廃棄率を高くしているために高い利用率で転送できる．しかし，HSTCP には TCP Reno との親和性についての問題点が指摘されている．式 (2.5)，式 (2.9) より，TCP Reno と HSTCP のスループットの比は式 (2.13) で示される．

$$\frac{T_{hs}}{T_{reno}} = \frac{1}{10p^{0.335}} \tag{2.13}$$

たとえばパケット廃棄率が 10^{-6} のとき，HSTCP と TCP Reno のスループットの比は 10.0 となり，HSTCP が TCP Reno の帯域を大きく食いつぶしてしまう．

2.5.3.2 STCP における TCP Reno との親和性

STCP についても，TCP Reno との親和性についての問題点が指摘されており，式 (2.5)，式 (2.11) より，TCP Reno と STCP のスループットの比が式 (2.14) として計算される．

$$\frac{T_{sca}}{T_{reno}} = \frac{0.0608}{p^{0.5}}. \tag{2.14}$$

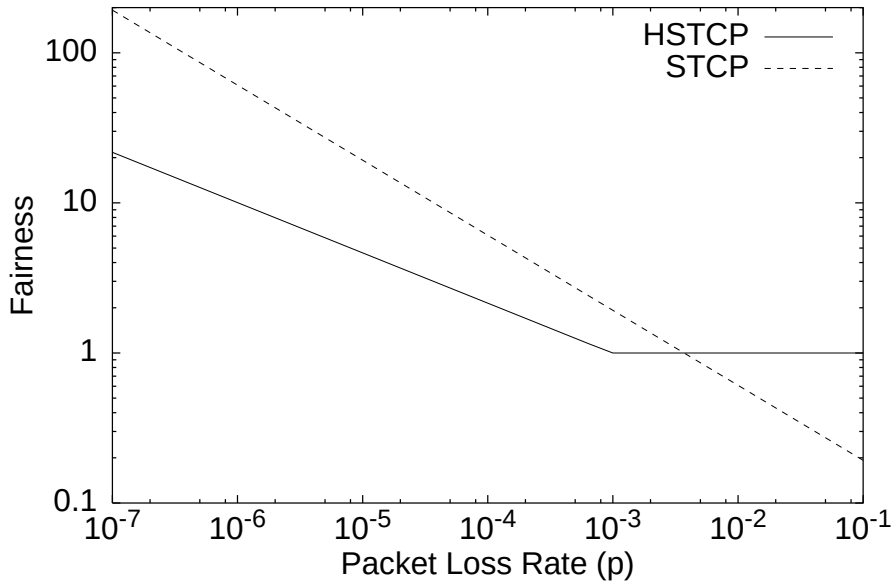

図 2.6: p と TCP Reno との親和性についての関係

たとえばパケット廃棄率が 10^{-6} のとき，STCP と TCP Reno のスループットの比は 60.9 となり，TCP のバージョンにより不公平が生じる．そのため，STCP の TCP Reno との親和性の問題は HSTCP よりも深刻であることが分かる．

2.5.3.3 廃棄率と TCP Reno との親和性についての関係

図 2.6 は式 (2.13) と式 (2.14) で導いた HSTCP と STCP の TCP Reno との親和性についての関係を示したグラフである．ここで，親和性を T/T_{reno} と定義し，T は TCP Reno と帯域を共有している TCP のスループットである．この際，親和性の値が 1 である場合，TCP Reno と他の TCP のコネクションのスループットが，同量になることから，公平な状況と考えられる．

HSTCP では $p \geq 10^{-3}$ の時，TCP Reno との親和性を意識して TCP Reno と同じ輻輳ウィンドウを設定している．そのため，この場合，HSTCP では親和性が 1 になっている．

一方，STCP では，$p \geq 2 \times 10^{-2}$ の時，TCP Reno よりも輻輳ウィンドウが小さく，ネットワークの廃棄率がこのような場合においては，TCP Reno よりも性能が落ちてしまう．そのため，この場合，STCP では親和性が 1 より少なくなってしまう．

$p \leq 10^{-3}$ の時，HSTCP の親和性は STCP よりも小さく，図 2.6 より，HSTCP のほうが TCP Reno との親和性があると分かる．しかしながら，p の値が小さくなるにつれて，HSTCP の親和性も悪化しており，改善の必要がある．

2.6 関連研究

本節では，(1) 公平性の課題に対するエンドホストの改善と，(2) 公平性の課題に対するネットワークの改善と，(3) 帯域使用効率の課題に対するエンドホストとネットワークの連携による改善に関する関連研究を示す．表 1.1 に示す課題と解決アプローチのマトリックスで示すように，公平性の課題に対するエンドホストの改善，帯域使用効率の課題に対するエンドホストとネットワークの連携による改善は他者により十分検討されており，本研究では考察の対象としていないため，関連研究の紹介にとどめる．

2.6.1 公平性の課題に対するエンドホストの改善

RTT の異なるコネクション間の帯域割当ての公平性については，現在までにその解決を目的とした様々な研究が行なわれている．下記に示すように，TCP 輻輳制御機構の改善におけるアプローチである関連研究では，ある程度の改善の効果が示されている．

2.6.1.1 Fang による提案方式

Fang らは文献 [25] にて，輻輳ウィンドウの増加速度を集約内の全てのコネクションが一定値に統一することで公平な帯域割当てを実現する提案を行っている．具体的には，輻輳回避フェーズにおける 1 秒あたりの輻輳ウィンドウ増加量 c (セグメント) を定め，ACK を受信する度に輻輳ウィンドウを下式のように増加させることで公平性の実現を図っている．

$$W_{new} = W_{old} + \frac{c \times RTT}{W_{old}} \tag{2.15}$$

この提案は問題の直接の原因である輻輳ウィンドウの増加速度を変更するものであるため，問題の改善に大きな効果を得ることができる．集約コネクション内にそのような動作を行わない通常の TCP を利用するコネクションが混在する場合には，そのコネクションと他のコネクションの間で割当てられる帯域量に差が生じることが予想される．特にウィンドウ速度が統一された値と混在する既存 TCP とで大きく異なる場合，両者の間には深刻な不公平性の問題が発生すると考えられる．

2.6.1.2 Hamann による提案方式

また文献 [26] では Hamann が増加速度を変更し，RTT による増加速度の影響を回避する提案を行っている．本提案では，下記のように増加速度を変更している．

$$W = W + \frac{1}{W} S_{wnd} \tag{2.16}$$

$$S_{wnd} = S_{wnd} + \beta RTT^2 \tag{2.17}$$

本提案では，ウィンドウの増加量を一定にせず，ウィンドウが大きくなるにつれて RTT の 2 乗に比例した値を加算し，その係数をウィンドウ増加量に利用している．この方式により，RTT が異なるコネクションが共存してもウィンドウの増加速度を RTT に依存しないという提案を行っている．

2.6.2 公平性の課題に対するネットワークの改善

集約コネクション内の帯域割当ての公平性の問題に対してネットワークにおける改善として，現在までにその解決を目的とした様々な研究が行なわれ，文献 [27, 28, 29, 25] などでその改善に関する提案が行われている．

文献 [27] において，Lin らは個別コネクションの情報を利用し，各コネクションの帯域を測定する TSW (Time Sliding Window) プロファイラと Random Early Detection (RED) with IN and OUT (RIO) [30] を組み合わせたキュー管理方式の改善を提案している．シミュレーションの結果，これらの改善アルゴリズムを組み合わせることにより，スループットや公平性が向上することを示している．Yeom らは文献 [28] にて，エッジノードがコネクション毎の情報を管理することで，異なる RTT を持つコネクションが集約された場合の公平性の向上を図る提案を行っている．またエッジノードにおける制御については，Nandy らが文献 [29] にて自律的な帯域調整機能を提案している．この提案では RTT の相違による余剰帯域の分配の不公平性を和らげるポリサーアルゴリズムの改善が行われており，更に TCP と UDP コネクションの混在環境に対しては TCP と UDP コネクションに対して異なる廃棄確率を適用することで不公平性の改善を図っている．このようなネットワーク内部の改善は，サービスを提供するネットワークが一括して機能を導入することが可能であるため，比較的実現が容易であると考えられる．しかし，上に挙げた提案は全てネットワーク内のノードに個別のコネクションの情報を持たせるものであり，多数のコネクションが集中する大規模なネットワークにおいてはスケーラビリティの問題が無視できなくなると予想される．

2.6.3 帯域使用効率の課題に対するエンドホストとネットワークの連携による改善

広帯域データ転送時の課題に対する TCP 輻輳制御機構とネットワークの連携による改善として，XCP を紹介する．XCP はエンドシステムとルータの体系的なシステムを提供し，ネットワークの帯域制御方式において理想形を追求している．XCP では，独自のパケットを使用し，エンドホストとネットワークで連携して各コネクションの輻輳ウィンドウの決定を行い，広帯域転送時の高い帯域使用率，異なる RTT のコネクション間での高い公平性を実現している．そのため，性能が他の方式と比較して著しくよい．しかし，実運用で適用することを考えると，現在使われている端末，装置を全て置き換える必要があり，実現は非常に困難になってしまう．

2.6.3.1 骨組み

送信者は輻輳ウィンドウ W と RTT を保持し，ルータにこれらの情報を輻輳ヘッダを介して知らせる．ルータは入力と出力の送信帯域を監視する．リンクの帯域と入力送信帯域の差をもとに，パケットの輻輳ヘッダを見ながら，リンクを共有しているコネクションに輻輳ウィンドウを増やすか減らすかの指示をする．フィードバックはコネクションごとの輻輳ウィンドウと RTT をもとに配分され，公平性を保てるようになっている．最終的に，そのパケットのフィードバックが，転送路のボトルネックの輻輳情報を保持させる．これが受信者に届き，それを ACK で送信者に知らせることにより，送信者は輻輳ウィンドウを必要に応じて変える．

2.6.3.2 輻輳ヘッダ

XCP パケットはそれぞれ輻輳ヘッダを持ち，コネクションの状態をルータが受信者に知らせるのに使われる．H_W フィールドは送信者の現在の輻輳ウィンドウ，H_rtt は送信者の現在の推定 RTT である．これらは送信者によって書き込まれ，転送の間は書き換えられることはない．

$H\_feedback$ には正か負の数が入り，送信者によって初期化される．転送経路上のルータがこのフィールドを更新し，これを介して送信者の輻輳ウィンドウを調整する．

2.6.3.3 XCP 送信者

TCP と同様，XCP 送信者は未送信パケットの輻輳ウィンドウ W と RTT の推定値を維持する．パケットを送信する際，送信者は輻輳ヘッダをパケットに付け，H_W フィールドに現在の W を代入し，H_rtt フィールドに現在の RTT の推定値を代入する．コネクションの最初のパケットでは，H_rtt には 0 を代入し，まだ RTT を推測するのに有効な情報がないことをルータに知らせる．

送信者は $H\_feedback$ に希望の輻輳ウィンドウの上げ幅を代入する．例えば，あるアプリケーションが送信速度 r を希望しているとき，送信者は $H\_feedback$ に，その希望の値になるような増加分 $(r \cdot rtt - W)$ を現在の輻輳ウィンドウの中にあるパケット数で割った値を代入する．もし帯域が十分に余っていれば，これによって送信者は 1 RTT で希望の送信速度にすることができるのである．

新しい ACK が返ってきたら，正のフィードバックであれば送信者は W を増やし，負のフィードバックであれば送信者は W を減らす．

$$W = max(W + H\_feedback, s)$$

s はパケットサイズである．

フィードバックに加え，XCP は，パケット廃棄にも対応しており，TCP と同様の方法で対処する．

2.6.3.4 XCP ルータ：制御法

XCP ルータの仕事は，フィードバックの計算をして理想的な効率と公平性にすることであり，パケット廃棄をしない．

フィードバックを計算するのに，XCP ルータは EC (efficiency controller) と FC (fairness controller) を使う．これらは両方とも，リンクを通っているコネクションの平均 RTT をもとに推測値を計算する．これによりウィンドウベース制御のプロトコルを安定させることができる．

具体的には，EC は乗法的増加乗法的減少 (MIMD) を用い，余った帯域に比例して増加を行うようにしている．こうすれば広帯域のリンクでもすぐに余った帯域を使うことができる．FC は AIMD を用い，公平性を保てるようにする [31]．つまり，それぞれのコントローラを独立に操作することにより，適切な制御法が実現するのである．

EC は，集合としてのトラヒックの動作はフィードバックに依存し，コネクションの数には依存しないという条件を満たしていることが分かる．それに加え，増加/減少法が輻輳の度合いに無関心な TCP に対し，EC のフィードバックは帯域がどの程度使えるのか，使いすぎているかに比例した度合いで送られる．さらに，フィードバックが平均 RTT で計算されていくことから，RTT が増加するとともに，帯域獲得の強度が弱くなる．

FC は AIMD を使うが，TCP よりも公平性を保てる．ここで注意してほしいのは，AIMD では送信速度にかかわらず全てのコネクションが同じ分だけ増えるということだ．したがって，乗法的減少が公平性を実現するのである．TCP では減少はパケットの廃棄と結びつくが，廃棄はまれであるべきだ．それに対し，XCP では減少は廃棄とは関係なく，平均 RTT ごとに毎回実行される．

2.6.4 関連研究と本研究の関連

公平性の課題に対するエンドホストの改善に関する Fang，Hamann による提案では，RTT の異なるコネクション間の公平性を改善しているが，公平性の改善の程度が高くないという点，既存方式の TCP Reno との親和性が高くないという点が指摘されている．エンドホストでの改善を図る方式は，スケーラビリティがあるが，他のコネクション間の性能向上を図ることを考えた場合，性能向上に限界があり，また，既存プロトコル間での性能悪化が懸念される．本研究では，コネクション間の性能を大きく向上するためには，ネットワークにおいて改善を図ることが望ましいと判断し，ネットワークの改善，エンドホストとネットワークの連携による改善により，公平性の課題の解決を行った．

公平性の課題に対するネットワークの改善の関連研究では，公平性の課題に対するエンドホストの改善と比較して，より公平に帯域の割当てが実現できている．しかし，紹介した方式は，ネットワークの全てのノードに個別のコネクションの情報を持たせるものであり，実現するには設備投資のコストが膨大になる．本研究の 3 章で提案する方式では，Diffserv 網の利点を生かし，エッジノードのみにコネクションの情報を持たせ，RTT の異なるコネクション間の公平性を実現する．

帯域使用効率の課題に対するエンドホストとネットワークの連携による改善で紹介した XCP の基本コンセプトは，理想的なネットワークを追求することである．そのため，帯域使用効率，RTT の異なるコネクション間の公平性，コネクション間の差別化が理想的に実現できる方式である反面，既存ネットワークを考慮していないため実運用で適用しようとする場合，装置を全て置き換える必要がある．本研究では，実運用で適用できるアプローチ，コネクション単体の性能向上に適したアプローチとして，エンドホストの改善によるアプローチが望ましいと判断し，5 章で提案を行った．XCP の理想的なネットワークの構想を元に，エンドホストの改善による最大限の課題の解決を図ることを目的とした．

第3章　ネットワークの改善による遅延の増大への対応

3.1 まえがき

本章では，RTT の異なるコネクション間で不公平な帯域割当て問題に注目し，ネットワークルータによる解決を図ることを目的としている．本研究では，この課題に対して，AF PHB を利用した Diffserv 帯域保証サービスにおける集約コネクション中での適用に焦点を絞り，エッジルータに対してパケットマーキングポリシーを提案する [32, 33, 34].

3.2 エッジルータにおけるマーキング方法の提案

3.2.1 エッジルータの機能

AF PHB を用いた Diffsev 網の帯域保証サービスでは，エッジルータでパケットに複数の優先度のマークを付ける．エッジルータおよびコアルータにおいて，ネットワークが輻輳した場合，優先度が低いパケットから順に廃棄して，優先度が高いパケットはできるだけそのまま転送する．エッジルータがマーキングを行う仕組みとして，図 3.1 で示すように，クラシファイア，メーターとマーカーの 3 つの機能をもつ．まずエッジルータに入ってきたパケットに対して，クラシファイアで集約ごとに分類し，メーターで集約コネクションごとに流入帯域を測定する．その後，計測した流入帯域をもとにマーカーで優先度に基づいたマーキングをパケットに対して行う．このとき，パケットの流入帯域を測定する方法として TSW (Time Sliding Window) を用いる．TSW は，過去に管理していた流入帯域と新しく到着することによって測定された流入帯域を重み付けをして，平均化された新たな流入帯域とするものである．その後，TSW の測定に基づいて，低廃棄率で転送するパケットには IN と，高廃棄率で転送するパケットには OUT とマーキングする．このマーキング方式は，TSW の計測に基づいて，2 種類にマーキングをするため，TSW2CM (Time Sliding Window Two Color Marker) と呼ばれる．一般的に，エッジルータにおけるマーキングは契約単位である集約コネクションに対して確率的に行っている．そこで，本章ではこのような集約単位でマーキングを行うマーカーを集約マーカーと呼び，集約マーカーのみを備えた従来のマーキング方式を既存マーキングと呼ぶ．マーキング処理の後，エッジルータおよびコアルータにおいて，RIO により優先度情報に基づいた廃棄処理を行う．ここで集約内の個々のコネクションに注目すると，既存マーキングはコネクション間の公平性を考慮せずにマーキングしているため，パケットの廃棄率はコネクションごとの転送帯域に関係なくどのコネクションでも同じである．したがって，コネクション i の OUT パケットの帯域，R_{i_out} は式 (3.1) で示される．ここでコネクション i の転送帯域を R_i，コネクション数を N，契約帯域を B とする．

$$R_{i\_out} = \frac{\sum_{i=1}^{N} R_i - B}{\sum_{i=1}^{N} R_i} R_i. \tag{3.1}$$

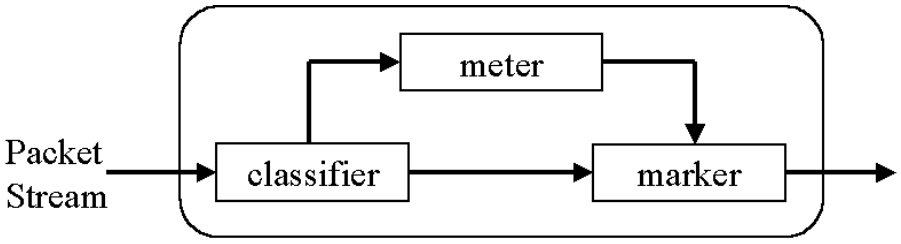

図 3.1: エッジルータの機能

3.2.2 不公平な帯域割当てが生じる原因

2.4.3 節に示したように，不公平な帯域割当てが生じる原因として，エッジルータにおける FR を考慮しないマーキングとパケット廃棄の影響が挙げられる．エッジルータではコネクション単位ではなく契約単位（集約コネクション単位）でパケットにマーキングを行うために，集約コネクション中のパケットは集約コネクション全体の帯域に応じてマーキングが行われる．そのため，集約全体の帯域が契約値を超過した場合には，FR に満たない速度で送信を行っているコネクションのパケットに対しても OUT にマーキングされる場合がある．OUT にマーキングされたパケットがネットワーク内で廃棄された際には，TCP の輻輳制御によって送信帯域が低下してしまう．送信帯域を下げることによって空いた帯域は，RTT が短くウィンドウの増加速度が速いコネクションがより多く獲得することが予想される．したがって，契約単位でパケットにマーキングをするだけでは，パケット廃棄に伴う送信帯域の低下により，RTT の違いに折より帯域獲得に不公平が生じてしまうと考えられる．

3.2.3 FR の算出，通知

本研究では，エッジルータでは，FR を用いてマーキングを行う．FR は契約帯域量と集約されているコネクション数を用いて計算され，$FR = \frac{\text{契約帯域量}}{\text{集約コネクション数}}$ で示される．しかし，集約されているコネクションの数は一定値ではなく，動的に変動している場合が一般的であると考えられる．そのため，エッジルータは，常に正確な FR を管理している必要がある．

ネットワークにおいて，FR を管理する方法として，送信ホストと Bandwidth Broker (BB)[35] やポリシーサーバが連携して制御することを提案する．図 3.2 にその仕組みを図示する．

帯域保証契約を結んでいる利用者が新たに帯域保証を希望するデータ転送を開始（あるいは転送を終了）する場合には，利用者はコネクションごとに送信開始（終了）メッセージを契約ホストに送る．このことにより，契約ホストは常に各コネクションの転送状況を管理可能になり，集約されたコネクションのコネクション数を把握可能になる．そこで，契約ホストは契約帯域と集約コネクション数から FR を算出し，ポリシーサーバに通知す

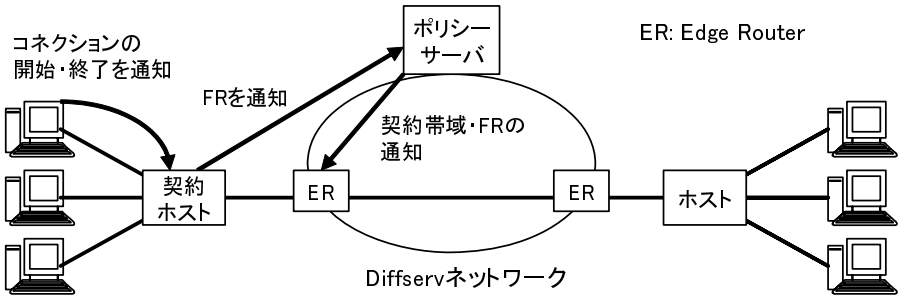

図 3.2: ポリシーサーバを介した FR の管理

ることで，ネットワーク側が集約コネクション数の変化を把握できるようにする．

ポリシーサーバは契約情報やネットワーク内のリソースを集中的に管理し，ネットワークの動的な変化に合わせた柔軟なポリシー制御を実現する．またポリシーサーバとエッジルータの間では，ポリシー制御のための通信プロトコルである COPS (common open policyservice) プロトコル [36] を利用した通信が可能であるため， ポリシーサーバが集約コネクション数の変化を検知した場合には，COPS プロトコルを利用してエッジルータに設定されている集約されたコネクション数に関する情報を更新する．

集約されたコネクション数が更新された場合は，契約ホストは FR を再計算し，ポリシーサーバを介して，エッジルータに通知する．

3.2.4 AI マーキング

不公平な帯域割当てを解決するためには，既存マーキングの機能を拡張し，集約に対するマーキングを行う集約マーカーだけでなく，個々のコネクションに対してマーキングを行うことが必要である．そこで，集約マーカーだけでなく，個別コネクションに対してマーキングを行う個別マーカーを備えたマーキングを提案する．集約マーカーと個別マーカーを組み合わせて制御する方式について考察する．

まず始めに提案するマーキング方法を AI マーキングと呼ぶ．図 3.3 に AI マーキング機構を示す．AI マーキングは集約マーカー，個別マーカーの順にマーキングを行う．既存マーキングと同様に集約マーカーにおいて集約コネクションの契約帯域に等しい量だけ，パケットを IN にマーキングし，それを上回るパケットを OUT にマーキングを行う．その後，個別マーカーにおいて転送帯域が FR に満たないコネクションの OUT パケットを IN に再マーキングを行う．以上のマーキングにより AI マーキングは，転送帯域が FR に満たないコネクションを優先転送する．

しかしこの方法は，個別マーカーにおける再マーキングの結果，IN のパケットが契約帯域を超過し，他の契約にも影響を及ぼす恐れがある．この解決策として，再マーキングによる IN パケットの超過量を集約マーカーにフィードバックし，次に入ってくるパケットに対して再マーキングしたパケット数だけ IN にマーキングするパケット数を減らすこ

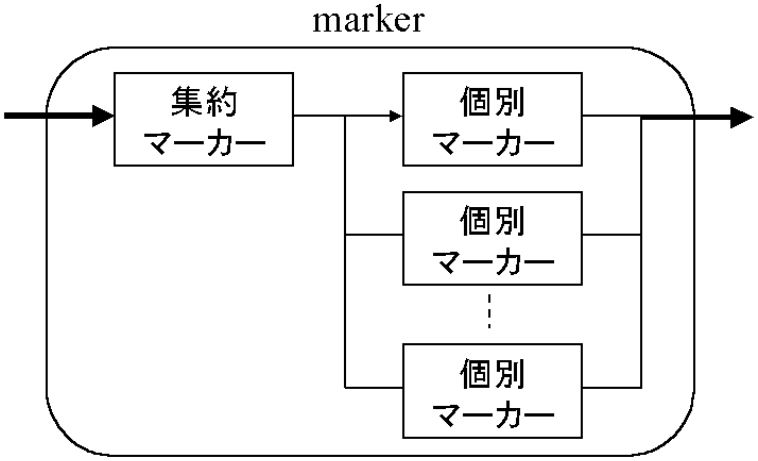

図 3.3: AI マーキング

とにする．

AI マーキングでは，転送帯域が FR に満たないコネクションへの OUT にマーキングを行わず，転送帯域が FR を超えるコネクションに対してはそのコネクションの転送帯域に比例した量のパケットを OUT にマーキングする．以上より R_{i_out} は式 (3.2) で示される．ここで R_i が FR を超えたコネクション i の集合を X とする．

$$R_{i\_out} = \begin{cases} \dfrac{\sum_{i=1}^{N} R_i - B}{\sum_{i \in X} R_i} R_i & \text{if } i \in X \\ 0 & \text{if } i \notin X \end{cases} . \tag{3.2}$$

3.2.5 IA マーキング

AI マーキングは，FR に満たないコネクションに対してのみ優先的にマーキングを行う．つまり，AI マーキングは FR に達していないコネクションの転送帯域が FR まで達することを可能にするが，コネクションの平均転送帯域を FR にするようにマーキングを行っているわけではない．

AI マーキングの FR を超えたコネクションについて考察する．上で述べたように OUT パケットの量はそのコネクションの転送帯域に比例する．また TCP コネクションのウィンドウは式 (2.2) で示すように徐々に増加するため，コネクションの送信帯域が大きく FR を上回ることはほとんどない．以上の 2 点から，FR を超えたコネクション間では OUT パケットがほとんど同じ比で存在していると考えられる．そのため，平均転送帯域が FR を下回っているコネクションが転送帯域が FR を超えてすぐに，パケット廃棄される可能性があり，平均転送帯域が FR に近づかない現象が考えられる．

この問題を解決しコネクションの平均転送帯域を FR にするには，FR を超えた後でも転送帯域の少ないコネクションに対しては優先的なマーキングを行う必要がある．具体的

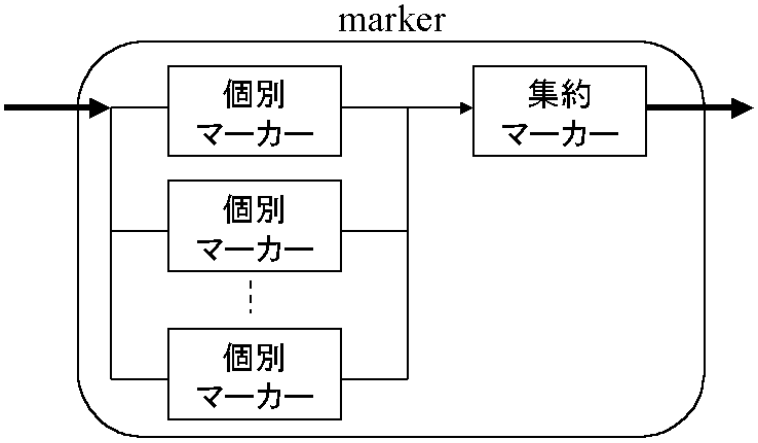

図 3.4: IA マーキング

には転送帯域の多いコネクションの廃棄率を高く，転送帯域の少ないコネクションの廃棄率を低くすることである．そこで，AI マーキングより公平なマーキングを実現するために，転送帯域が FR を超えている場合でも転送帯域の少ないコネクションに対する優先的なマーキングを行う方法の，IA マーキングを提案する．

図 3.4 に IA マーキング機構を示す．IA マーキングはまず個別マーカーで，個々のコネクションの FR を実現するために，FR に等しい量だけ IN にマーキングし，それを上回るパケットに対しては OUT にマーキングする．FR を下回る帯域のコネクションはすべてのパケットに対して IN にマーキングする．次に集約マーカーで，IN および OUT の帯域の最適化を行う．具体的には OUT にマーキングされたパケットの一部を IN に再マーキングし，集約コネクションの IN にマーキングされたパケットの帯域を契約帯域に等しくする．これは，個別マーカー終了時点では，FR を下回る帯域のコネクションがある場合，その分だけ IN パケットが契約帯域より少なくなるからである．IN パケットが少ない状態は，契約帯域に達しないときにもパケットが廃棄される点や，非契約帯域が空いている場合にその帯域を効率よく利用できないという点で問題である．

IA マーキングでは，FR を上回るコネクションの OUT にマーキングされたパケットに対して再マーキングされるため，OUT パケットはコネクションの転送帯域のうち FR を上回る帯域の比で存在する．したがって R_{i_out} は式 (3.3) で示される．

$$R_{i\_out} = \begin{cases} \dfrac{\sum_{i=1}^{N} R_i - B}{\sum_{i \in X}(R_i - FR)}(R_i - FR) & \text{if } i \in X \\ 0 & \text{if } i \notin X \end{cases} . \tag{3.3}$$

以上より IA マーキングは，FR の超過量に比例するようにパケットに対して OUT にマーキングをするために，転送帯域に比例して OUT へのマーキングを行う AI マーキングよりも相対的に送信帯域の高いコネクションを優先的に廃棄でき，より公平な帯域割当てを実現すると期待できる．

AI マーキングおよび IA マーキングは，既存のマーキングと異なり，集約されたコネクションだけでなく，各コネクションの帯域を計測し，計測結果および FR に基づきマーキングを行う．また，AI マーキング deha, 各コネクションのマーキング状況に応じて集約に対するマーキングにフィードバックも行う．そのため，エッジルータに高い性能が要求され，装置コストが上がることが避けられない．特定の集約コネクションに特化して，コネクション間の公平性を実現するサービスとして運用することで，装置のコスト高を抑えることが可能になる．

3.2.6 IND マーキング

また AI マーキングおよび IA マーキングの有効性を確認するために，個別マーカーのみを持ち，個々にしかマーキングを行わない場合を考え，このマーキングを IND マーキングと呼ぶ．

IND マーキングとは IA マーキングと異なり，個別マーカーの後に，集約マーカーがないもので，そのマーキング機構を図 3.5 に示す．そのためルータの負荷を考えた場合は，IND マーキングのほうが提案マーキングよりも処理が速くなると考えられる．しかし，IA マーキングでは集約マーカーを設置することで，高廃棄率による過剰なパケット廃棄を回避でき，集約マーカーの有無による影響は全体の帯域利用率に反映されると考えられる．また，IND マーキングは IA マーキングと同じ個別マーカーでパケットを OUT にマーキングしているため，同等の公平性を実現すると考えられる．IND マーキングの OUT パケットの帯域を式 (3.4) に示す．

$$R_{i\_OUT} = \begin{cases} R_i - FR & \text{if } i \in X \\ 0 & \text{if } i \notin X \end{cases} . \tag{3.4}$$

3.2.7 提案機能の実装インパクト

本節では，AI マーキングおよび IA マーキングにおける実装のインパクトについて考察を行う．提案する両マーキング方式は，個別マーカーを備えているため，クラシファイヤの機能が複雑になり，メータ，マーカーの機能が個別のコネクションごとに設けて，処理を行う必要がある．

Diffserv 網のエッジルータにおいて，クラシファイアでは，コネクションを契約単位で識別を行い，DS コードポイント付与することでクラス分けを行う．提案するマーキングでは，クラシファイした後に，コネクション毎に識別する必要があり，識別子として VLAN タグや送信元 IP アドレスと宛先 IP アドレスの組み合わせが考えられる．本機能を提供するためには，数千単位の VLAN を VLAN 毎に識別し，処理を行う機能が必須となる．また，識別子を用いて識別された各コネクション毎に，帯域の計測を行う必要がある．帯域計測の機能は，コネクション毎に設ける必要があるため，数千単位のメータが必要になる．集約されたコネクションの帯域は，各コネクションの帯域を元に，算出することで実

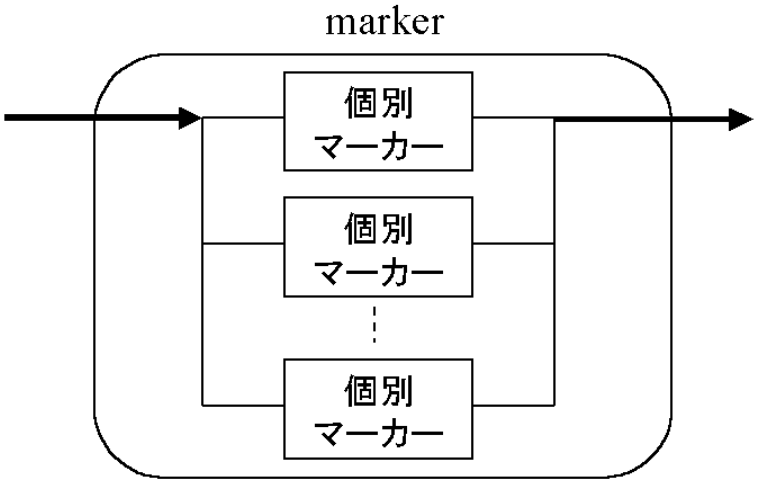

図 3.5: IND マーキング

現できる．次に，マーカーでは，集約マーカーと個別マーカーを直列に設ける必要がある．AI マーキング，IA マーキングともに，後段のマーカーにおいては，メータで計測した帯域および前段でマーキングされた情報を元にマーキングを行う．

実装する際には，コネクション毎に識別する機能，コネクション毎の帯域計測機能，直列にマーカーを配備し高速で処理する機能，後段のマーカーにおける，計測帯域とマーキング情報を元に行うマーキング機能に実装の困難性が考えられる．Diffserv 網の帯域保証の契約は，企業の 1 支店等が単位となっているため，提案するマーキングで 1 契約の中に集約されるコネクションは，多くとも数百コネクションであると考えられる．そのため，複数の契約で提案するサービスを行う場合であっても，数千単位のコネクションの識別機能，数千単位のコネクションの帯域計測機能，数千単位のコネクションのマーキング機能があれば十分であると考えられる．

現在の装置実装状況は，数千単位のコネクションの識別機能は，VLAN サービスが汎用的になっていることからもわかるとおり，現在の装置で実装されており [37]，容量も数百 Gbit/s の大規模なルータ，スイッチが開発されている．また，数千単位の帯域計測機能，マーキング機能も，機能として実現の難易度が高いが，複数のメーカーで実装されており [38]，容量も数百 Gbit/s の大規模なルータ，スイッチが開発されている．以上より，提案するマーキング機能を実装するには十分インパクトがあるが，数千単位のコネクションに対してならば，現技術で実現可能であると考えられる．

3.3 提案マーキング方法の性能評価

Diffserv 網の帯域保証サービスで，エッジルータにおいて AI マーキングと IA マーキングを適用した場合の集約コネクション内の公平な帯域割当てを確認するために，コンピュータシミュレーションによる評価を行った．

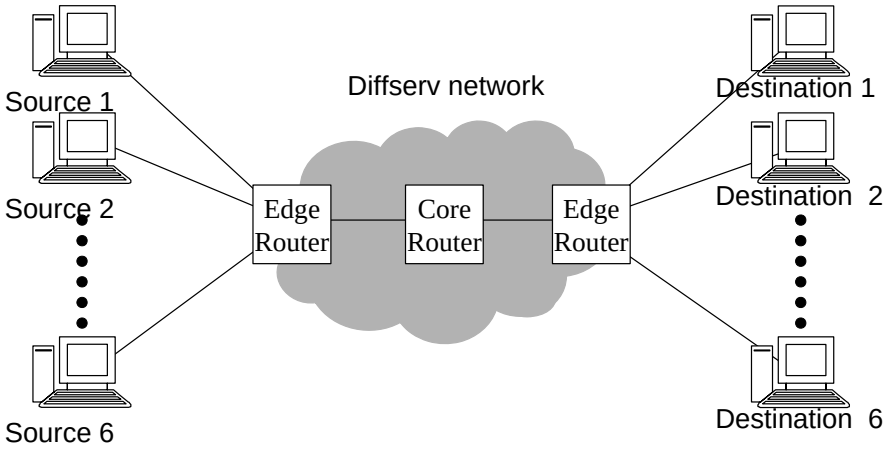

図 3.6: シミュレーションモデル

3.3.1 シミュレーション環境

AF PHB を利用した Diffserv 帯域保証サービスでの集約コネクション内の帯域の公平な帯域割当てを確認するために，提案マーキング方式の AI マーキングと IA マーキングを既存マーキング方式と個々のマーキングを行う方式の IND マーキング方式と比較した．シミュレーションは network simulator ver.2(ns-2) [39] を用いて評価を行った．

本シミュレーションは図 3.6 に示すシミュレーションモデルを用いた．6 つの TCP バルクデータをそれぞれ Source 1〜6 から Destination1〜6 に送信し，Diffserv 網内で契約帯域量 9 Mbit/s の 1 つのコネクションに集約する．各々のコネクションの RTT は評価内容により変化させる．エッジルータでは既存マーキング, AI マーキング, IA マーキングと IND マーキングの 4 種のマーキング方式の比較を行う．

シミュレーション中に用いたパラメータとして，以下の項目は共通に用いた．また，本章で行うシミュレーションでも，以下のパラメータを用いることとする．

- TCP タイムスタンプオプション [40] を利用する
- TCP 最大輻輳ウィンドウはウィンドウスケールオプション [40] を用いることで最大 1 Gbytes まで変動可とする
- TCP ECN 機能を利用 [12] する
- 受信側でのパケットの処理時間は 0 秒とする
- TCP セグメントサイズは 1000 bytes とする

シミュレーション条件は表 3.1 の通りである．

シミュレーションの評価値として個々のコネクションに割当てられる帯域量を用いる．契約は集約に対して行われるため，その契約帯域量を集約されたコネクションで平等に割

表 3.1: シミュレーション条件

DS ドメイン内の帯域	10 Mbit/s
契約帯域量	9 Mbit/s
集約内のコネクション数	6
FR	1.5 Mbit/s
IN マーキング廃棄確率	0.02
OUT マーキング廃棄確率	0.50
シミュレーション時間	100 秒 (最初の 20 秒を除く)

表 3.2: シミュレーション条件 (基本動作)

RTT	20 ms, 40 ms, 60 ms, 80 ms, 100 ms, 120 ms
転送プロトコル	全て TCP Reno を使用

当てられた場合の FR を，各々のコネクションの公平な帯域量して考えられる．そこで本章のシミュレーションではコネクションに割当てられた帯域量と FR の比較に着眼する．

なお, 帯域割当て量の計算式は

$$Rate = \frac{\text{転送データ量}}{\text{転送時間}} \tag{3.5}$$

で与えられる．

本章で行った評価の内容について述べる．

(1) 帯域割当ての公平性 ： 本章の目的である集約コネクション内の帯域割当ての公平性が提案方式によって改善されているかを確認する．

(2) 余剰帯域使用時の使用率，及びそのときの公平性 ： 契約されていない Diffserv 内の帯域を提案方式が効率よく使用できるか，そのとき公平性を維持しているかを評価する．

3.3.2 帯域割当ての公平性

提案方式である AI マーキングと IA マーキングの基本的な動作を確認するため，以下の環境でシミュレーションを行った．シミュレーション条件を表 3.2 にまとめる．

図 3.7 に各マーキングにおける個々のコネクションの転送帯域を示し，表 3.3 に集約コネクションの転送帯域と Fairness index（F）[41] を示す．F は式 (3.6) で示される公平性を示す指標の一つであり，$F = 1$ に近いほど公平性が高い．

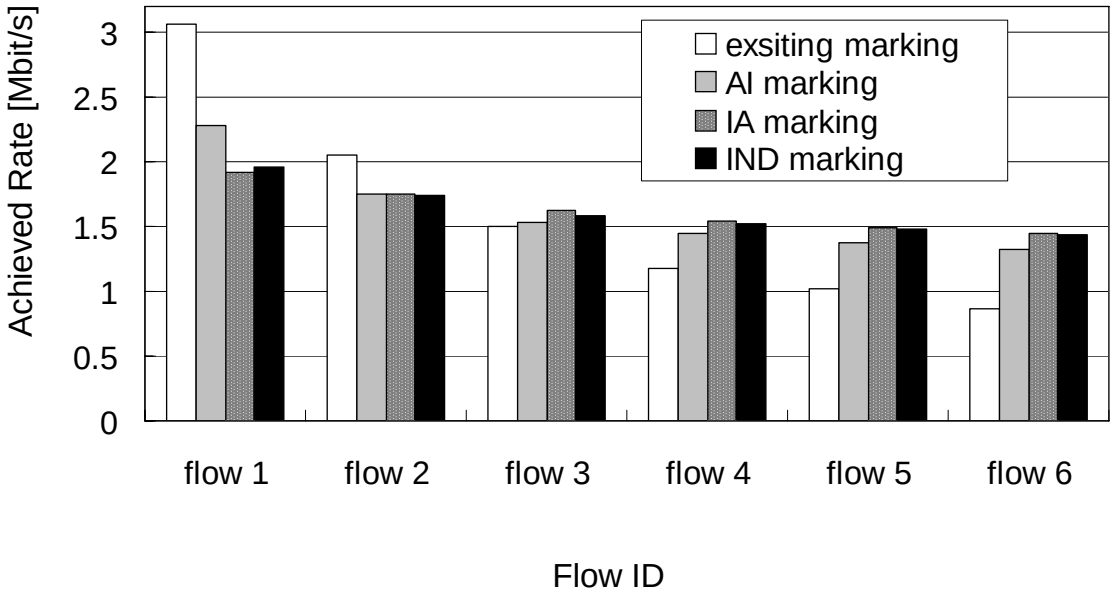

図 3.7: コネクションと転送帯域の関係

表 3.3: 基本動作での結果

marker	集約コネクションの帯域	F
既存マーキング	9.688 Mbit/s	0.8205
AI マーキング	9.716 Mbit/s	0.9608
IA マーキング	9.784 Mbit/s	0.9903
IND マーキング	9.739 Mbit/s	0.9878

$$F = \frac{(\sum_{i=1}^{N} R_i)^2}{N \times \sum_{i=1}^{N} (R_i)^2}. \tag{3.6}$$

表 3.3 より，4 つのマーキングにおける集約コネクションの転送帯域がほぼ同量であることがわかる．さらに，図 3.7 より個々のコネクションの転送帯域は既存マーキングでは RTT の大小に大きく依存しているのに対して，提案マーキングではその点を改善している．特に IA マーキングと IND マーキングは，RTT の長いコネクションと短いコネクションの転送帯域の差が少なく，すべてのコネクションが FR に近づいているため，より公平な帯域割当てを実現しているといえる．またこのことは表 3.3 の F からも分かる．本シミュレーションにおいて，IA マーキングと IND マーキングは，契約帯域における帯域割当ての公平性を実現していることを示した．

次に集約コネクション内に RTT の短い 5 本の TCP コネクションと RTT の長い 1 本の TCP コネクションが混在する場合の各コネクションの帯域割当ての公平性を評価する．

シミュレーション条件を表 3.4 にまとめる．

表 3.4: シミュレーション条件 (公平性の評価 1)

RTT	40ms, 40ms, 40ms, 40ms, 40ms, 130ms
転送プロトコル	全て TCP Reno を使用

表 3.5: 公平性の評価 1 の測定結果

Flow #	RTT (ms)	FR (Mbit/s)	既存マーキング (Mbit/s)	AI マーキング (Mbit/s)	IA マーキング (Mbit/s)	IND マーキング (Mbit/s)
1	40	1.500	1.862	1.700	1.695	1.670
2	40	1.500	1.863	1.699	1.692	1.685
3	40	1.500	1.778	1.730	1.676	1.678
4	40	1.500	1.786	1.698	1.687	1.666
5	40	1.500	1.795	1.723	1.683	1.683
6	130	1.500	0.705	1.269	1.410	1.410
total		9.000	9.790	9.820	9.844	9.791
F			0.9390	0.9900	0.9961	0.9963

表 3.5 に提案マーキング方式と既存マーキング方式の 4 つのマーキング方式についてネットワーク内における各コネクション割当てられた帯域の測定結果をまとめる．ただし，帯域割当て量の単位は Mbit/s である．

本評価においても，提案方式の AI マーキング，特に IA マーキングは既存マーキング方式と比べ，公平性が改善されていることが分かる．しかしながら，RTT が 130 ms のコネクションの帯域は目標とすべき FR に達していない．これは，提案方式では，RTT の長いコネクションの帯域を一時には FR を越えるまでは優先的に転送はするものの，FR を越えた時点でパケットに OUT にマーキングされ，転送帯域が半減するる恐れがある方式だからである．結果として，輻輳ウィンドウの増加量が低い RTT の長いコネクションは，転送帯域の平均値が FR を越えないことになる．また，IND マーキングも個々のコネクションを意識したマーキング方式であり，公平性があるといえる．

次に，提案するマーキング方式がどの程度公平性を改善するかを評価するため，1 コネクションだけの RTT が異なるようにコネクションを集約し，そのコネクションの RTT を変化させた場合の各コネクションの帯域割当ての公平性を評価する．

シミュレーション条件を表 3.6 にまとめる．

図 3.8，図 3.9 に，各マーキング方式における変化させた RTT (T) と当該コネクションのネットワーク内で測定したスループットおよび公平性 F の関係を示す．

図 3.8 より，コネクションの RTT が増加するにしたがって，既存マーキングは著しくそのコネクションの帯域割当て量が低下しているのに対して，他の 3 種のマーキング方式，

表 3.6: シミュレーション条件 (公平性の評価 2)

RTT	40 ms, 40 ms, 40 ms, 40 ms, 40 ms, T ms (可変)
転送プロトコル	全て TCP Reno を使用

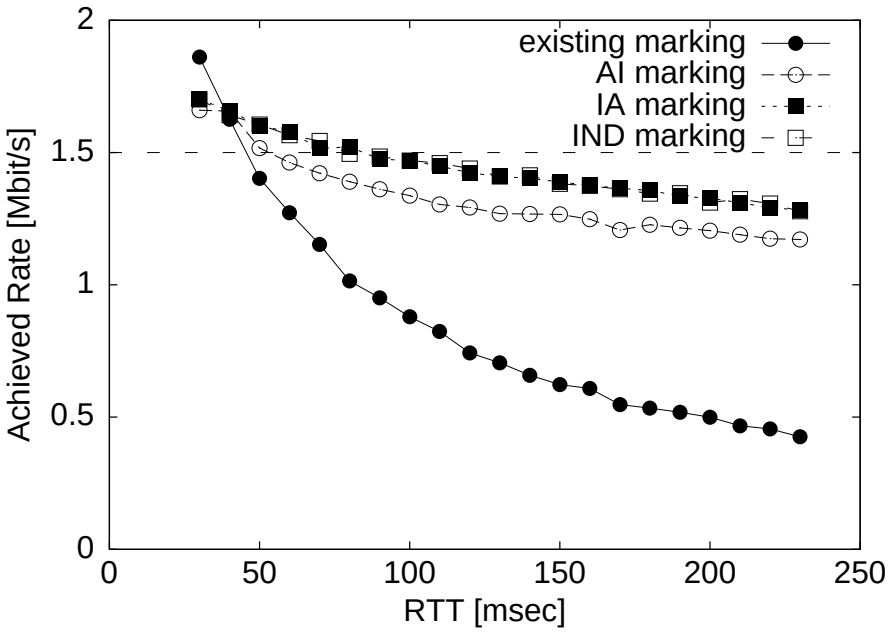

図 3.8: コネクションの RTT の変化に対する帯域割当て量の関係

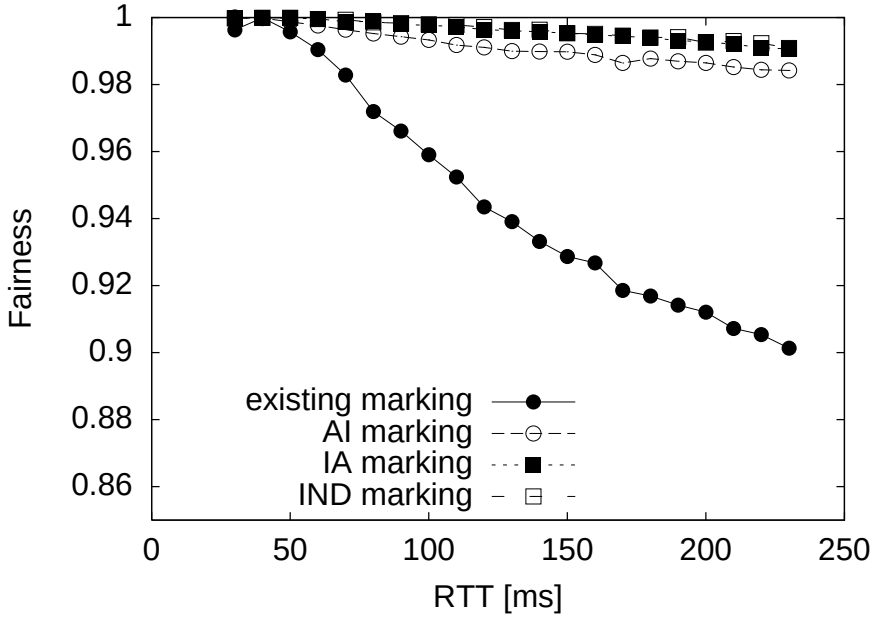

図 3.9: コネクションの RTT の変化に対する公平性の関係

AI マーキング, 特に IA マーキングと IND マーキングは FR に近い値を維持して推移している．また，スループット同様，公平性も，図 3.9 より，コネクションの RTT が増加するにしたがい，既存マーキングの公平性が低下しているのに対して，他の 3 種のマーキング方式，AI マーキング, 特に IA マーキングと IND マーキングは高い公平性を維持している．以上から，提案方式および IND マーキングは既存マーキングと比べ，RTT の長いコネクションと短いコネクションの比が大きくなり，送信ウィンドウの増加率が大きく異なるという厳しい状況でも公平性を実現できるマーキング方式であることが分かる．さらに，IA マーキングと IND マーキングは RTT の長さに関わらず，ほとんど同じ帯域割当て量を示しているため，IND マーキングと同様に IA マーキングは公平性を実現できていることが分かる．また，RTT の短いコネクションの RTT が 40 ms に対して，長いコネクションの RTT は 230 ms という 2.5 倍以内の比の場合 FR に近い値を実現できることも分かる．以上から，提案方式 AI マーキング, IA マーキングは RTT が長くなるという厳しい状況でも，公平な帯域割当てを実現できることが分かる．

3.3.3 余剰帯域使用時の使用率およびそのときの公平性

集約コネクション内のコネクションの一部が FR に満たない転送を行っている場合の, 余剰帯域の帯域使用率およびそのときの公平性についての 4 種のマーキング方式の測定結果をまとめる．

シミュレーション条件を表 3.7 にまとめる．

表 3.7: シミュレーション条件 (余剰帯域使用率)

RTT	40 ms, 40 ms, 40 ms, 40 ms, 40 ms, T ms (可変)
転送プロトコル	コネクション 1 のみ UDP (0.1 Mbit/s の固定帯域) ほかのコネクションは TCP Reno を使用

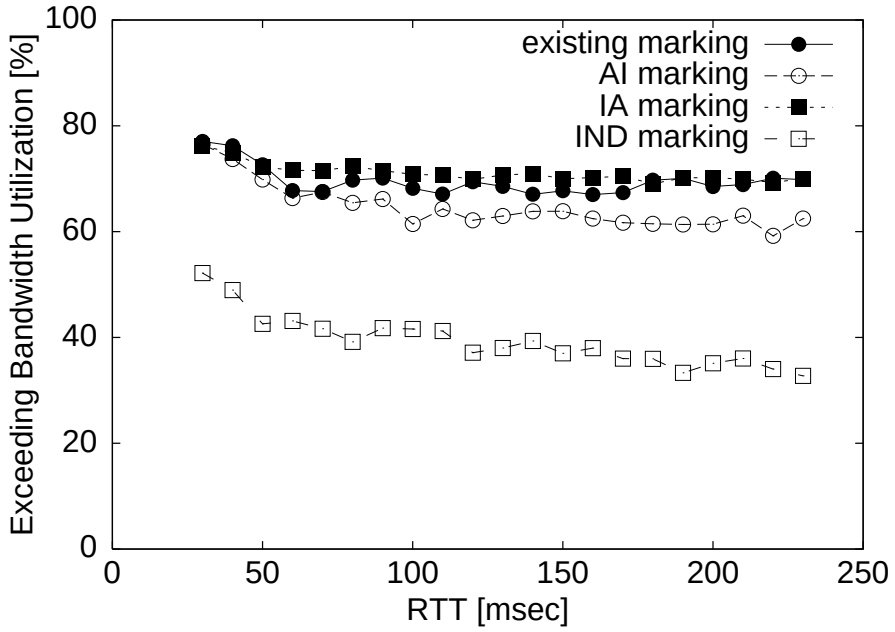

図 3.10: コネクションの RTT の変化に対する余剰帯域使用率の関係

図 3.8 に，各マーキング方式における変化させた RTT (T) と余剰帯域使用率の関係を示す．グラフの横軸はコネクションの RTT で単位は ms，縦軸は余剰使用率で単位は%である．

また，表 3.8 に，$T = 230$ ms のときの帯域割当て量，公平性および 3.3.2 節の公平性の評価 2 との帯域割当て量，公平性の差分を示す．

図 3.10 から，余剰帯域使用率について 4 種のマーキングを比較する．

既存マーキング, AI マーキングおよび IA マーキングは RTT の長さに関わらず，余剰帯域を 70%前後で使用できているのに対して，IND マーキングは，余剰帯域を 40%前後しか使用できていないことがわかる．これは，既存マーキング, AI マーキングおよび IA マーキングはマーキングにおいて，IN のマーキング量を契約帯域と等しい量にしているのに対して，IND マーキングは IN へのマーキングの量を調節していないため，OUT へのマーキングの量が過剰になっており，IND マーキングのパケット廃棄率が高くなっているからである．以上から，提案方式は余剰帯域使用率の点で，IND マーキングより優れていることが分かる．

次に，余剰帯域使用時の公平性について，表 3.8 より考察をする．3.3.2 節の公平性の評

表 3.8: $T = 230$ ms のときの帯域割当量と公平性

		既存 マーキング	AI マーキング	IA マーキング	IND マーキング
本評価における結果	Flow #1	0.099	0.100	0.100	0.100
	Flow #2〜5	2.286	2.078	2.057	1.973
	Flow #6	0.456	1.214	1.371	1.334
	Fairness	0.8730	0.9680	0.9799	0.9811
3.3.2 節公平性の評価 2 との差分	Flow #1	-1.753	-1.633	-1.597	-1.596
	Flow #2〜5	0.405	0.354	0.344	0.278
	Flow #6	0.030	0.043	0.087	0.056
	Fairness	-0.0087	-0.0135	-0.0090	-0.0082

価 2 と本評価で，$T = 230$ ms の時に着目し，Flow #1 の使用率が変化したときの他のコネクションの帯域割当量の変化，公平性の変化を評価する．表からわかるとおり，本評価の前提条件により，Flow #1 の使用帯域が著しく減少している．このとき，全てのマーキング方式も RTT の短いコネクション (Flow #2〜5) に多くの帯域 (0.27〜0.45 Mbit/s) が割当てられている．一方で，RTT の長いコネクション (Flow #6) に割当てられる帯域は，0.03〜0.09 Mbit/s と著しく少ない．このことから，FR として割当てられている帯域以外の余剰帯域に関しては，RTT の異なるコネクション間で帯域を公平に割当てられていないことがわかる．これは，表の公平性を示す値 F が低下していることからもわかる．しかし，本評価では目標とすべき値として FR を設定しており，FR 以外の余剰帯域に関する公平性については，課題は残るものの，本研究での目的は達成していると考えられる．

3.3.4 シミュレーション評価のまとめ

提案マーキング方式の AI マーキング, IA マーキングはともに公平な帯域割当てを行っており，これは RTT の長さが大きくなっても実現できることがわかる．しかしながら，RTT の差が大きくなるにつれて，RTT の長いコネクションが目標としていた FR の値を達成できていないこともわかった．IA マーキングに関しては，個々にマーキングを行う IND マーキングと同程度の公平な帯域割当てを実現できている．また，IND マーキングに比べ IA マーキングは余剰帯域を効率よく使用できた．一方で，集約されたコネクションが FR に満たない帯域を使用している場合，その空き帯域の使用に関しては，依然公平性の課題が残ることもわかった．

以上のことから，AI マーキング, IA マーキングは状況に応じた公平な帯域割当てを行うマーキングを行っており，特に IA マーキングは公平性を実現する最適なマーキングであることが分かる．

3.4 まとめ

本章では AF PHB を適用した Diffserv 帯域保証サービスにおける集約コネクション中の不公平な帯域割当て問題に注目した．これは帯域保証契約が複数のコネクションで締結され，集約コネクション内の各 TCP コネクションの RTT が異なる場合に，TCP の輻輳制御機構が原因で生じる，それぞれのコネクションの帯域割当て量が不公平になる問題である．

その解決策として，以下の 2 つの提案を行った．1 つ目は，エッジルータへの FR の通知である．まず，集約コネクション内のコネクションが公平とされる帯域として，FR を定義した．その後，契約ホストにおいて契約帯域と集約コネクション数から FR を計算し，ポリシーサーバを介してエッジルータに FR を通知方法を提案した．2 つ目は，FR に基づく，エッジルータによる公平な帯域割当て量を実現するマーキング方法，AI マーキングと IA マーキングを提案した．これらの提案マーキング方式は，集約コネクションに対するマーキングを行うだけでなく，個々のコネクションに対するマーキングを行っている．

次に，既存マーキング方式と個別にマーキングを行う方式の IND マーキングの 2 種の方式と比較するために，コンピュータシミュレーションによる性能評価を行った．コンピュータシミュレーションを用いたその有効性の検証の結果，既存マーキング方式よりも AI マーキングを用いた場合において，RTT の長いコネクションに対しての帯域割当て量が増加した．AI マーキングの問題点を解決する IA マーキングも併せて提案し，評価を行った結果，それぞれのコネクションが FR に近い帯域割当て量を獲得でき，より公平な帯域割当てが実現できた．そして，これらの提案方式は RTT の長さが大きくなっても公平性は改善できた．しかしながら，提案方式の AI マーキング，IA マーキングは，RTT の長いコネクションの割当て帯域を FR に達することができない場合があることも確認した．OUT マーキングの量の関係から，IND マーキングに比べ AI マーキングは余剰帯域を効率よく利用できることもわかった．さらに，集約されたコネクションが FR に満たない帯域を使用している場合，その空き帯域の使用に関しては，依然公平性の課題が残ることもわかった．

以上のことから，AI マーキング, IA マーキングは状況に応じた公平な帯域割当てを行うマーキングを行っており，特に IA マーキングは，RTT の異なるコネクション間の公平性の課題に対してネットワークにおける解決アプローチとしては，最適なマーキングであることが分かる．したがって，提案方式（IA マーキング）はネットワーク上での公平な帯域保証サービス実現を実現するマーキング方式である．

第4章　ネットワークとエンドホストの連携による遅延の増大への対応

4.1 まえがき

本章は，3 章と同様に，RTT の異なるコネクション間で不公平な帯域割当て問題に注目し，この課題に対して，AF PHB を利用した Diffserv 帯域保証サービスにおける集約コネクション中での適用に焦点を絞っている．この課題に対して，3 章と異なるアプローチのネットワークとエンドホストの連携による解決を図り，FR-TCP を提案する [42, 43, 44].

4.2 FR-TCP の提案

4.2.1 必要な機能

提案機構では，エッジルータで計算される FR を送信側ホストへデータパケットを用いて通知し，送信側ホストが通知された FR に合わせた輻輳制御を行うことで，集約コネクション内の不公平性の改善を図る．

集約コネクション内の不公平な帯域割当ては，TCP の輻輳制御が大きく関わっている．既存の TCP は他のコネクションを意識した制御を行うことができないため，たとえ集約内のコネクション間で不公平な状態でデータ転送が行われていたとしても，それぞれの TCP は公平な状態へ送信帯域を調節することができない．そこで TCP の輻輳制御を改善し，公平性を意識した送信帯域の調節を行う必要があると考えられる．

公平性を意識した制御では，ネットワーク内においてその時点で使用可能な帯域を集約されている各コネクションが均等に分け合うべきとも考えられる．しかし，それを実現するためにはリンクを共有している集約コネクション以外のコネクションへの影響も考慮する必要がある．また現行の TCP とは大幅に動作の異なる制御が必要となることが予想され，既存のプロトコルとの異種混在環境における性能の不安定性が懸念されることから，そこまでの変更は現実的ではないと考えられる．

そこで本章ではより現実的な方法として，集約内の各コネクションに最低限割当てられるべき公平な帯域量として定義した FR を意識した輻輳制御を TCP が行い，TCP が平均して FR 以上の帯域での送信を可能にする FR-TCP を提案する．FR-TCP を用いることにより，集約コネクション内で実現されるべき最低限の公平性が達成されると考えられる．そのため，送信側ホストにて FR を意識した制御を行う場合には，契約帯域量と集約コネクション数より計算可能な FR を送信側ホストへ通知する仕組みが必要になると考えられる．本章ではそのような FR の通知を TCP のフィードバック型輻輳制御機構の中で実現する方法についても併せて提案する．

以上より，FR-TCP に必要な機能としては

1. ネットワークにおいて送信側ホストへの FR の通知機構
2. エンドホストにおいて FR を意識した TCP 輻輳制御

が挙げられる．

以下，これら 2 つの機能の実現方法について詳しく説明する．

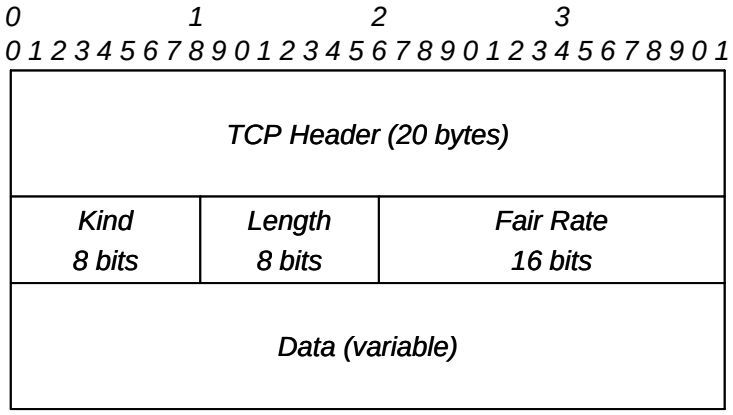

図 4.1: TCP FR オプションヘッダ形式

4.2.2 FR の通知機構

送信側ホストにおいて FR を意識した制御を行うためには，契約帯域量と集約コネクション数より計算可能な FR を送信側ホストへ通知する必要がある．FR-TCP ではこれを実現するため，エッジルータにて各 TCP コネクションのパケットヘッダに FR を記述することによって通知する方法を提案する．

ここで，FR の算出およびエッジルータへの通知方法は，3 章で提案した方法と同じ方法を使用する (3.2.3 節).

4.2.2.1 FR フィールド

パケットヘッダに FR を記述するために，TCP ヘッダ，IP ヘッダ (IPv4) のオプションフィールドにそれぞれ TCP FR オプションフィールドと IP FR オプションフィールドを定義する．図 4.1，図 4.2 にそれぞれのヘッダ形式を示す.

TCP FR オプションの構成は以下の通りである.

Kind : 8 bits : オプションの種類を記述

Length = 4 : 8 bits : オプションの全体長 (4 オクテット) を記述

FR (Fair Rate) : 16 bits : FR の値を記述 (単位は bit/s)

IP (IPv4) FR オプションの構成は以下の通りである.

Type : 8 bits : オプションの種類を記述

Length = 4 : 8 bits : オプションの全体長 (4 オクテット) を記述

FR (Fair Rate) : 16 bits : FR の値を記述 (単位は bit/s)

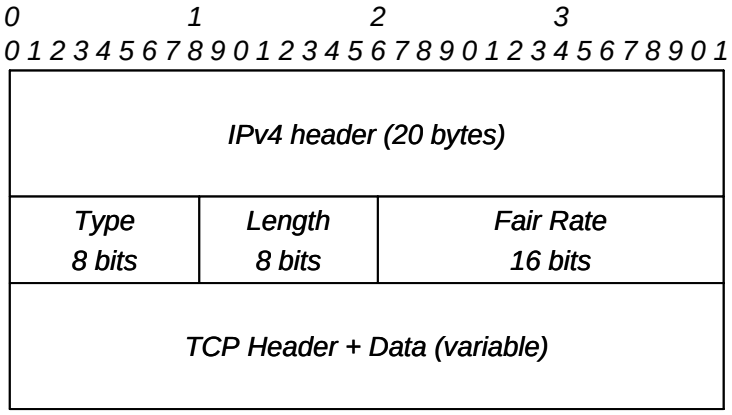

図 4.2: IP (IPv4) FR オプションヘッダ形式

各 FR オプションの内，FR を記述するフィールドは 2 オクテット長とし，FR は二進浮動少数形式で表される．これは，ATM ABR 転送方式 [45, 46] を参考にしており，RM セル中の ACR を記述するフィールドと同様の記述方法を採用した．

また IPv6 を利用する場合，IP FR オプションはホップバイホップオプションヘッダを利用する．図 4.3 にヘッダ形式を示す．

IP (IPv6) FR オプションの構成は以下の通りである*．

Next Header：8 bits：次のヘッダの種類を記述

Extention Length：8 bits：ホップバイホップオプションの全体長を記述

Type：8 bits：オプションタイプの識別子を記述

Length = 4：8 bits：オプションデータフィールド長 (4 octets) を記述

FR (Fair Rate)：16 bits：FR の値を記述 (単位は bit/s)

Padding：16 bits：パディングオプション

このようにパケットのヘッダに FR オプションを定義して FR を記述することで，データとは独立して FR を伝達する場合と比べて伝達の際のオーバヘッドが抑えることができると考えられる．

4.2.2.2 FR の通知手順

ポリシーサーバは，契約帯域を管理しており，契約ホストは TCP コネクションの開始，終了をポリシーサーバに通知する．ポリシーサーバは，契約ホストから通知される FR に

*TLV 符号化オプションを利用

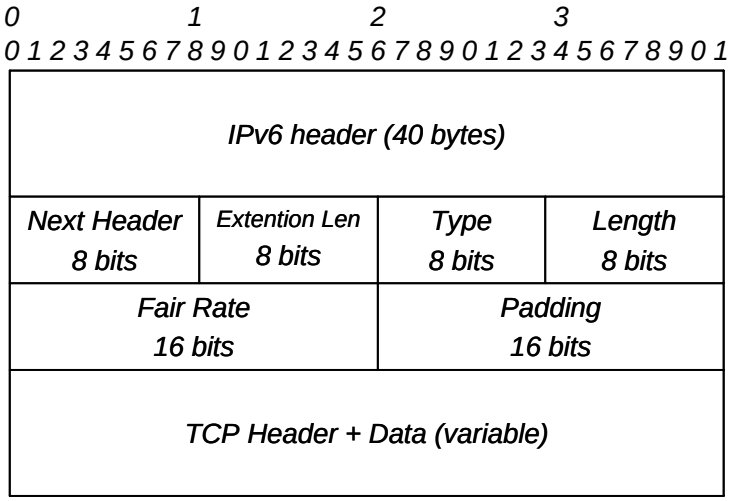

図 4.3: IP (IPv6) FR オプションヘッダ形式

変更があった場合は，その FR をエッジルータに通知する．FR-TCP では，全てのパケットに前節で定義した FR オプションフィールドを設け，以下の手順で送信側ホストへ FR を通知する．

(手順 1)：各 TCP コネクションは，パケットの TCP ヘッダー，IP ヘッダーに FR フィールドオプションを適用し，データ転送を継続する．

(手順 2)：各 Diffserv ドメインの入口のエッジルータにてルータは IP FR フィールドを参照し，記入値が 0 の場合は計算した FR を記述する．また 0 以外の値が記入されていた場合，計算値と記入値を比較して最小値を FR として記述する．

(手順 3)：IP FR フィールドに値が記述されたパケットが受信側へ到着した場合，受信側ホストは通知された FR を ACK パケットの TCP FR フィールドにコピーして送信側ホストへ送信する．

(手順 4)：ACK パケットの TCP FR フィールドを参照することで，FR を送信側に伝えることが可能になる．

図 4.4 に FR 通知機構を示す．図では集約内の 1 つの TCP コネクションに注目し，コネクションが集約内コネクション数の異なる 2 つのドメインを通過する場合について描かれている．この場合，FR は最初に通過するドメインの方が小さくなるため，送信側へは最初のドメインの入口エッジルータにて計算，記述された FR が通知される．

TCP, IP 両方のヘッダに FR オプションを定義したのは，FR 通知機構内でレイヤ・バイオレーション†を避けるためである．上で説明してきたように FR は送信側ホストが利用

†階層型プロトコル内で階層構造を破り，他層の情報を識別，判断材料とすること

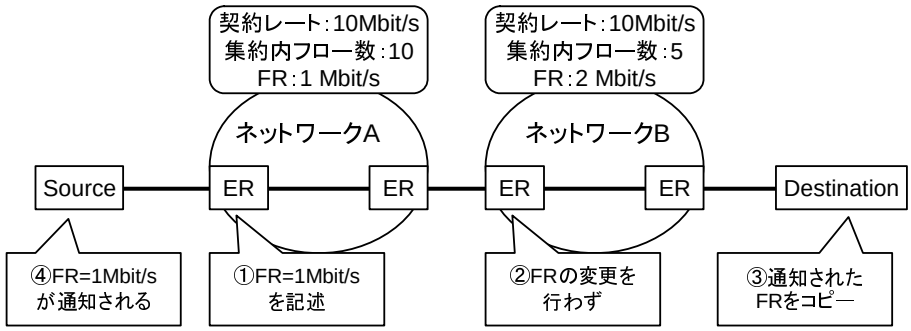

図 4.4: FR の通知機構

するので，トランスポート層の情報として送信側ホストへ通知される必要がある．しかしネットワークノードであるルータが参照できるのは基本的にネットワーク層までであり，トランスポート層の情報は参照しない．そのため，TCP, IP どちらか一方のみにしか FR オプションを定義しない場合，FR を伝達するためにはルータ，もしくは送信側ホストのどちらかにおいてレイヤ・バイオレーションを引き起こしてしまう．

そこで提案方式では，TCP の ECN 機構 [12] において送信側ホストへ輻輳発生信号 (ECN) を通知する際の手順を参考に，TCP, IP 両方に FR オプションを定義する．ネットワーク中のルータでは FR を IP FR フィールドに記述し，受信側ホストにおいて到着パケットから FR の値を取り出して ACK パケットの TCP FR フィールドへコピーすることで，レイヤ・バイオレーションを起こすことなく FR をトランスポート層の情報として送信側ホストへ伝達する．

ただし，ECN と FR の通知機構を比較した場合，前者はビットをセットするのに対し後者はフィールドに値を記入するものであるため，FR 通知機構の方がその処理により大きなコストがかかると考えられる．

4.2.3 FR を意識した TCP 輻輳制御機構

本節では，通知された FR を利用した送信側ホストにおける輻輳制御について述べる．

提案する輻輳制御機構は，TCP Reno の輻輳制御を基本としている．これは，既存の TCP とできる限り同じ制御を行うことで FR-TCP の導入を容易にし，また既存 TCP と FR-TCP が混在するような環境にも問題なく対応できるようにすることを意図している．ただし FR-TCP は FR を考慮した制御を行うため，TCP Reno とは異なる以下の 2 つの機能，(1)FR に合わせた送信帯域の調節機能，(2)ECN 機能により制御する．

以下，それぞれの機能について説明する．

4.2.3.1 FR に合わせた送信帯域の調節

提案輻輳制御固有の機能の 1 つとして，通知される FR に合わせて送信帯域を調節する輻輳ウィンドウ制御を提案する．これは，輻輳ウィンドウ制御のパラメータの 1 つである $ssthresh$ を常に FR に対応する値に設定することによって実現される．

$ssthresh$ はスロースタートフェーズと輻輳回避フェーズの境界値として用いられている．また TCP Reno では，$ssthresh$ は再送処理後の輻輳ウィンドウの値と同じ値に設定される．このことから，$ssthresh$ は安定状態において TCP コネクションが見積もるネットワーク中の利用可能帯域量の最低値といえる．そこで各コネクションが $ssthresh$ を FR に合わせた値にすることで，平均して常に FR 以上の速度でデータ送信を行うことが可能になると考えられる．

ACK パケットにより FR が通知される度に，送信側ホストは FR (bit/s) と TCP が見積る RTT (sec)，送信セグメントサイズ $Size_{seg}$ を用いて $ssthresh_{FR}$ を次のように計算し，$ssthresh$ に設定する．

$$ssthresh_{FR} = \frac{RTT \times FR}{Size_{seg}}(Segments) \tag{4.1}$$

式 (4.1) の導出については後述する．

更に FR-TCP では，再送処理後の輻輳ウィンドウとして $ssthresh_{FR}$ の値を設定する．FR-TCP のウィンドウ制御は，各コネクションが FR に対応する大きさにウィンドウを固定した上で，残された使用可能な帯域量を見積りつつ送信を行うような制御となっている．これは FR まではより積極的にデータを送信することを意味しており，積極的なデータ送信によってネットワークへ過度の負荷を掛けることも懸念される．しかし，提案ウィンドウ制御では相対的に RTT の大きなコネクションがより積極的にデータを送信すると共に，相対的に RTT の小さなコネクションがより控え目にデータを送信するような状態へ導くため，集約コネクション全体のデータ送信を考えた場合，ネットワークへの影響は既存の TCP と変わらないと考えられる．

なお，再送信タイマのタイムアウトによってパケットの廃棄を検出した場合には，転送路の故障や資源供給の失敗による重度の輻輳の発生によってネットワークが帯域保証サービスを実行できない状態に陥っていると考えられる．そのため，通常どおり輻輳ウィンドウを 1 (または 2) にセットしてスロースタートアルゴリズムを実行する．

式 (4.1) は以下のように導いた．

輻輳ウィンドウ W は，そのコネクションが送信したデータの内，ACK を受け取っていないデータ量，すなわちネットワーク中に存在しているデータ量を表している．通常，TCP は輻輳ウィンドウ分のデータをバースト的に送信し，RTT 時間分待った後にそれらのデータに対する ACK を受け取ることから，TCP の平均送信帯域は次のように近似される[‡]．

[‡]RTT の見積り誤差およびネットワーク中のパケット廃棄による再送処理により値には誤差が生じる．

$$SendRate = \frac{W}{RTT} \tag{4.2}$$

ここで上で説明した理由から，安定状態における平均送信帯域の最低値は $ssthresh$ を用いて次のように表される．

$$SendRate_{min} = \frac{ssthresh}{RTT} \tag{4.3}$$

これより式 (4.3) 中の $SendRate$ を目標値である FR に置き換えることで，FR に合わせた $ssthresh_{FR}$ を計算する式 (4.1) が導かれる．

4.2.3.2 ECN 機構 [12] の利用

前節で説明したウィンドウ制御に加え，FR-TCP では TCP ECN 機構を利用した制御を行うことを提案する．具体的には ECN によって輻輳が通知された際に，輻輳ウィンドウを $ssthresh_{FR}$ に設定する．

ECN を採用する理由として，ECN によって再送処理に伴う送信帯域の低下を極力抑えることが可能になる点が挙げられる．TCP 輻輳制御では，重複 ACK の受信や再送信タイマのタイムアウトによってパケットの廃棄を検出した場合，通常のデータ転送に優先して廃棄されたパケットの再送処理を行う．この場合，再送処理中は新しいパケットを送信できないため，その期間中の送信帯域は目標値である FR を下回ってしまう．また，再送処理にかかる時間は RTT に比例すると考えられるため，再送処理を行うことによって RTT の異なるコネクション間のデータ送信の不公平性の増大が懸念される．FR-TCP の再送制御は TCP Reno と同じ制御を行う．そのため前節で説明した FR に合わせたウィンドウ制御を用いただけでは集約内の全てのコネクションが FR 以上での送信を実現できないと考えられる．

図 4.5 に，提案ウィンドウ制御のみを用いた場合の送信側ホスト (FR を 1.5 Mbit/s とする) におけるスループットの時間変化を示す．2.4.2 で行ったシミュレーションと同じ条件で行い，フロー #1 のスループットの時間変化を計測した．

図のように，送信中には廃棄されたパケットの再送処理に伴い，送信帯域が著しく低下する期間があることが分かる．この事実から，集約内の全てのコネクションが平均して FR 以上の速度でデータを送信するためには，セグメント廃棄によって発生する送信帯域の低下を生じさせないような仕組みが必要である．

以上の理由から，FR-TCP では安定して FR 以上の速度でデータ送信を行うために，ECN を利用した制御を行う．ECN を用いることで，FR-TCP は ECN が正常に動作する限りはパケット廃棄による重複 ACK の受信よりも速やかに，かつネットワーク内でパケットが廃棄されること無く輻輳発生を検知することが可能となる．そこで FR-TCP では ECN によって輻輳発生を検知した際に，セグメント再送処理後の場合と同様に輻輳ウィンドウを $ssthresh_{FR}$ に設定することで，送信帯域が安定して FR 以上となるようなデータ送信を実現する．

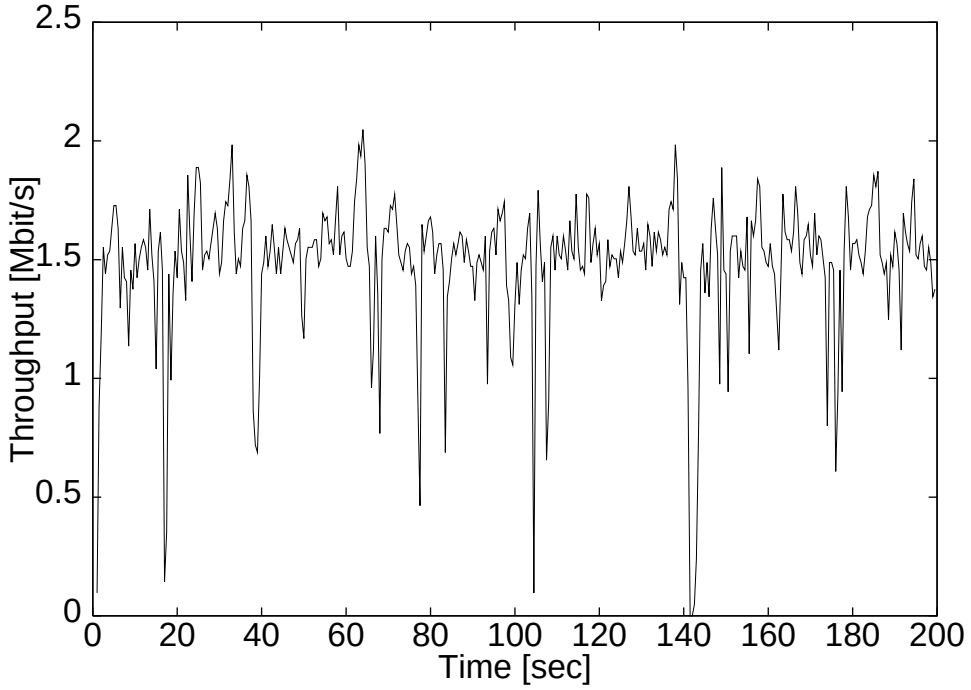

図 4.5: 送信側スループットの時間変化（提案ウィンドウ制御のみを利用）

4.3 FR-TCP の性能評価

4.3.1 シミュレーション環境

前章で提案した FR-TCP の有効性を確かめるために，提案方式を TCP Reno と比較するためのコンピュータシミュレーションを行った．

今回のシミュレーションでは提案方式の評価方法として，現在当該分野のシミュレーションに広く利用されている Network Simulator version 2 (ns-2)[39] に対して提案方式を実装し，既存の TCP Reno と比較して提案方式が Diffserv 網内における帯域割当ての公平性を改善しているか否かを評価する．

シミュレーションは図 4.6 に示すようなトポロジーを用いて行った．

図のようなトポロジにおいて, 契約者とネットワークの間には B Mbit/s を保証する契約が締結され, 複数のコネクションがこの契約を利用してネットワーク内を通過する．この場合, FR はその時点でのコネクション数を n として $FR = \frac{B}{n}$ (Mbit/s) と計算される．

各リンクの帯域は 10 Mbit/s とした．また, Diffserv 網内の各リンク, および出口エッジルータと受信側ホストの間のリンクの遅延は 10 ms とし, 送信側ホストと入口エッジルータの間のリンクの遅延を様々な値に設定することで, 多様な RTT を持つ TCP コネクションが集約されるようなシミュレーションを行った．

FR-TCP のパラメータとして以下の機能を用いた．

- 送信パケットについて IP FR オプションを利用する．

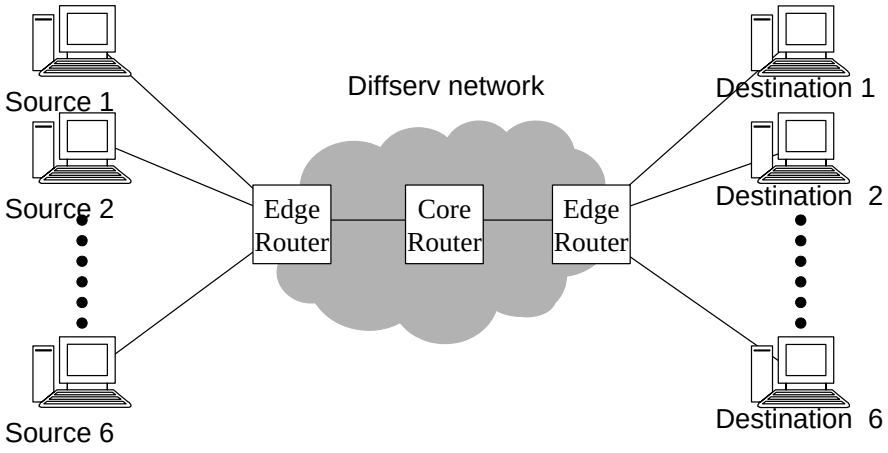

図 4.6: シミュレーショントポロジ

- ACK パケットについて TCP FR オプションを利用する．

エッジルータに用いたパラメータは以下の通りである．

トラヒック調整機構 : TSWTCM (Time Sliding Window Three Colour Marker)[47]

CIR (Committed Information Rate) : 9 Mbit/s

PIR (Peak Information Rate) : 9.5 Mbit/s

Diffserv 網中の全て（コア，エッジ含む）のルータで用いるパラメータは下表のように定める．

キュー管理アルゴリズム : RIO を AF PHB 用に拡張する．

バッファサイズ : 200 packets

定常状態における性能を測定するために, シミュレーション開始から 20 秒が経過した時点で測定を開始した．この時点では, TCP 輻輳制御機構が安定してから十分な時間が経過していると考えられる．また各 TCP コネクションの動作が同期することを防ぐため，シミュレーション開始後 4.5 秒から 5.5 秒が経過するまでの間に各 TCP コネクションをランダムに発生させた．シミュレーション開始後 200 秒が経過した時点でシミュレーションを終了させ，その際に以下のような値を測定した．

- Diffserv 網内における各コネクションのスループット *Rate* (単位：Mbit/s)
- 送信側ホストの輻輳ウィンドウ (単位：セグメント)
- 送信側ホストにおけるスループット (単位：Mbit/s)

- コネクション間の帯域割当ての公平性を示す指標 [41] F

なお, スループットの計算式は

$$Rate = \frac{\text{転送データ量}}{\text{転送時間}} \tag{4.4}$$

であり，F は i 番目のコネクションのスループットを $Rate_i$, コネクション数を n として

$$F = \frac{(\sum_{i=1}^{n} Rate_i)^2}{n(\sum_{i=1}^{n} Rate_i^2)} \tag{4.5}$$

と計算される．

シミュレーションは同じ条件について 10 回ずつ行い，測定値の平均値を結果として採用した．

評価項目は以下の通りである．

(1) 帯域割当ての公平性 : 本章の主たる目的である，集約コネクション内の帯域割当ての公平性が FR-TCP によって改善されているかどうかを確認する．

(2) 複数契約環境への対応 : 帯域保証契約が複数締結されている環境における FR-TCP の性能を評価する．

(3) コネクション数の動的変化への対応 : コネクション数が変化し，FR が動的に変化するような環境に FR-TCP が対応できるかどうかを評価する．

(4) 異種混在環境への対応 : 既存の TCP と FR-TCP が混在する環境において，FR-TCP が安定した性能を発揮できるかどうかを評価する．

4.3.2 帯域割当ての公平性

FR-TCP の基本的な動作を確認するため，2.2.1 節で説明したシミュレーションとほぼ同じ環境で，トランスポート層プロトコルとして FR-TCP を利用した場合のシミュレーションを行った．

シミュレーション条件を表 4.1 にまとめる．

ネットワーク内における各コネクションのスループットの測定結果 $Rate_{FR}$ を TCP Reno と FR-TCP に関して，図 4.7 に示す．

また，表 4.2 に TCP Reno と FR-TCP の場合の帯域使用量と公平性を示す Fairness Index (F)[41] を示す．

表中の F より，提案する FR-TCP は TCP Reno と比較して帯域割当ての公平性が改善されていることがわかる．また $Rate_{FR}$ より，FR-TCP を利用した各コネクションは全て FR 以上の帯域で転送が行われていることがわかる．

表 4.1: シミュレーション条件 (基本動作)

集約内のコネクション数	6 本 (固定)
契約帯域量	9 Mbit/s
FR	1.5 Mbit/s
RTT	40 ms, 80 ms, 120 ms, 160 ms, 200 ms, 240 ms
TCP 輻輳制御機構	全て FR-TCP を使用

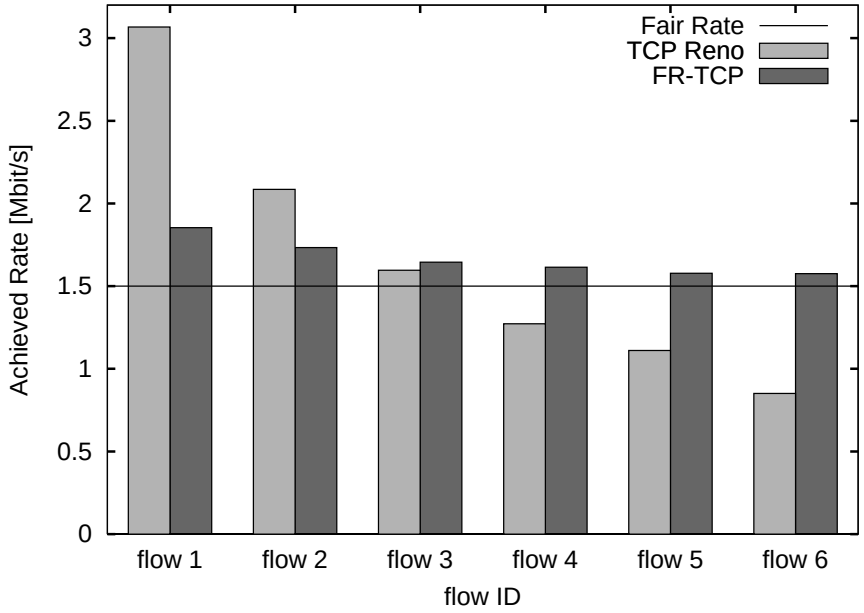

図 4.7: 測定結果 (基本動作)

表 4.2: 測定結果 (基本動作)

	$Rate_{Reno}$	$Rate_{FR}$
Total	9.983 (Mbit/s)	9.999 (Mbit/s)
F	0.8355	0.9965

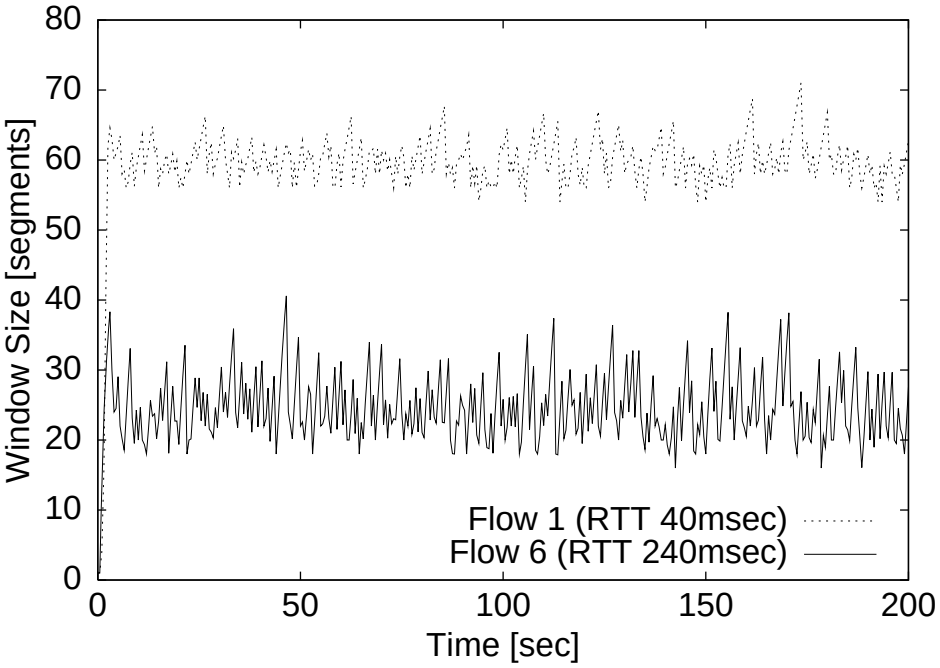

図 4.8: 輻輳ウィンドウの時間変化

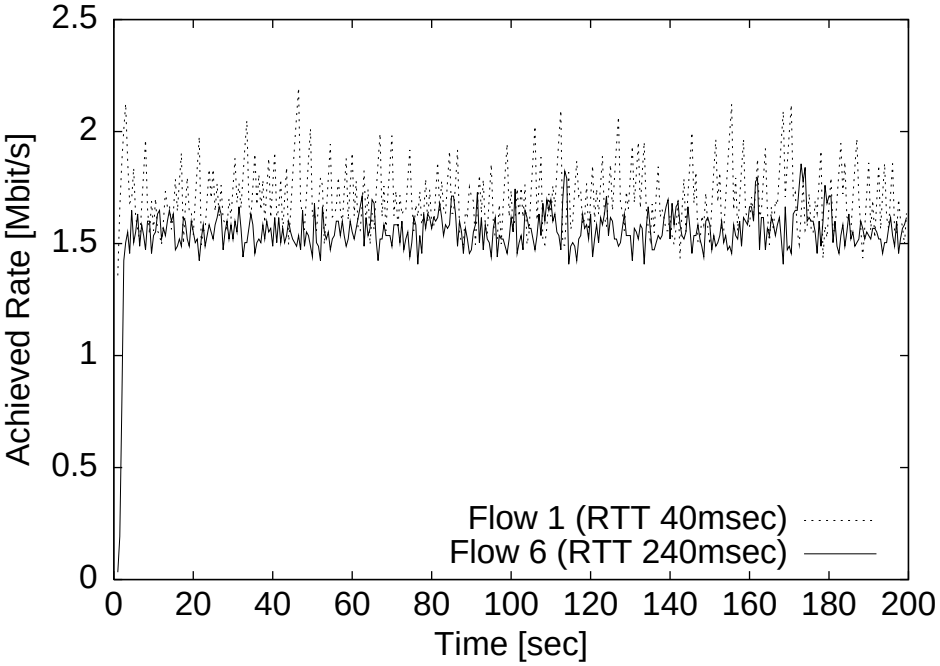

図 4.9: 送信側スループットの時間変化

表 4.3: シミュレーション条件 (公平性 1)

集約内のコネクション数	6
契約帯域量	9 Mbit/s
FR	1.5 Mbit/s
RTT	40 ms, 40 ms, 40 ms, 40 ms, 40 ms, 30〜1210 ms (T ms)
TCP 輻輳制御機構	全て FR-TCP を使用

図 4.8 に Flow 1 (RTT = 40 ms) と Flow 6 (RTT = 240ms) が含む TCP コネクションの輻輳ウィンドウの時間変化を示す．また図 4.9 に各コネクションの送信側におけるスループットの時間変化を示す．

シミュレーション終了時の TCP が見積る RTT の平均値は，RTT が 40 ms のコネクションで約 96.8 ms, RTT が 240 ms のコネクションで約 308.8 ms であった[§]．これより，各コネクションが FR に合わせた送信を行うために必要なウィンドウサイズは，RTT が 40 ms の場合は，

$$\frac{1.5 \times 10^6(bit/s) \times 0.0968(sec)}{1000 \times 8.0(bit/MSS)} = 18.2(\text{セグメント}) \tag{4.6}$$

と計算され，また，RTT が 240 ms の場合は，

$$\frac{1.5 \times 10^6(bit/s) \times 0.3088(sec)}{1000 \times 8.0(bit/MSS)} = 57.9(\text{セグメント}) \tag{4.7}$$

と計算される．図 4.8 において，各コネクションの輻輳ウィンドウはほぼ常に計算値を上回っていることから，FR-TCP では通知される FR に対応する大きさで輻輳ウィンドウが変化していることがわかる．そしてそのような輻輳ウィンドウの調節を行い，ECN 機能を利用することで，図 4.9 にあるように各コネクションが FR に近い帯域で安定したデータ転送を行っていることがわかる．

次に FR-TCP がどの程度公平性を改善するかを評価するため，1 コネクションだけ RTT が異なるようにコネクションを集約し，そのコネクションの RTT を変化させた場合の各コネクションの帯域割当ての公平性を評価する．

シミュレーション条件を表 4.3 にまとめる．

図 4.10 に変化させた RTT (T) と当該コネクションのネットワーク内で測定したスループットの関係を示す．なお図には比較のため，同様の条件で輻輳制御に TCP Reno を用いた場合についての結果も併せて示す．

図に示されるように，TCP Reno を利用した場合は RTT が増加するにつれそのスループットが低下し，RTT が他のコネクションと比べて約 10 倍の大きさ (410ms) になる場合には，スループットは FR の約 30%しか達成できていないことがわかる．それに対し FR-TCP を利用した場合，RTT が他のコネクションよりも突出して大きくなった場合で

[§]実際の RTT にはキューイングによる遅延が含まれる

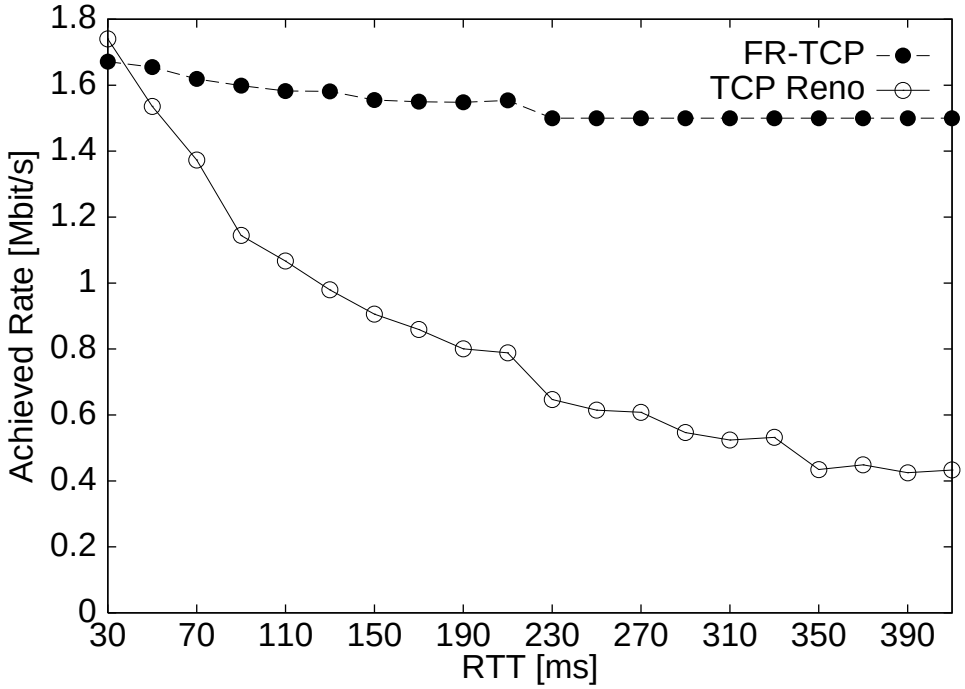

図 4.10: RTT の変化に対するスループット特性

も常に FR 以上の帯域で転送が行われていることがわかる．これより，FR-TCP は集約コネクション内でより RTT の大きなコネクションに対して適用した場合に，その改善効果が大きくなると考えられる．

更に図 4.11 に，変化させた RTT (T) と集約コネクションの公平性を示す指標 F の関係を示す．

図より明らかなように，FR-TCP は帯域割当ての公平性を大きく改善しており，変化させた RTT が他のコネクションよりも大きくなるにつれてより大きな改善効果が得られていることがわかる．

また，図 4.10 の RTT が 30 ms の場合に注目すると，TCP Reno を用いた場合のスループットの方が FR-TCP のそれを上回っており，この環境では TCP Reno の方が優れた性能を示しているように見える．

しかし，これは当該コネクションが他のコネクションよりも RTT が短くなったために割当てられる帯域量が増した結果であり，公平性の点からは 図 4.11 にあるように，FR-TCP の方が優れた結果を示していることがわかる．

このように FR-TCP は，既存の TCP を用いた場合に FR 以上の帯域で転送を行うようなコネクションが利用している帯域を，FR 以下の帯域でしか転送を行うことができないコネクションに割当てることによって公平性を向上させている．これより，FR-TCP は集約コネクション全体のスループットを落とすことなく公平性を向上させることができていると考えられる．また，全体のスループットが既存 TCP と同様なことから，FR-TCP は

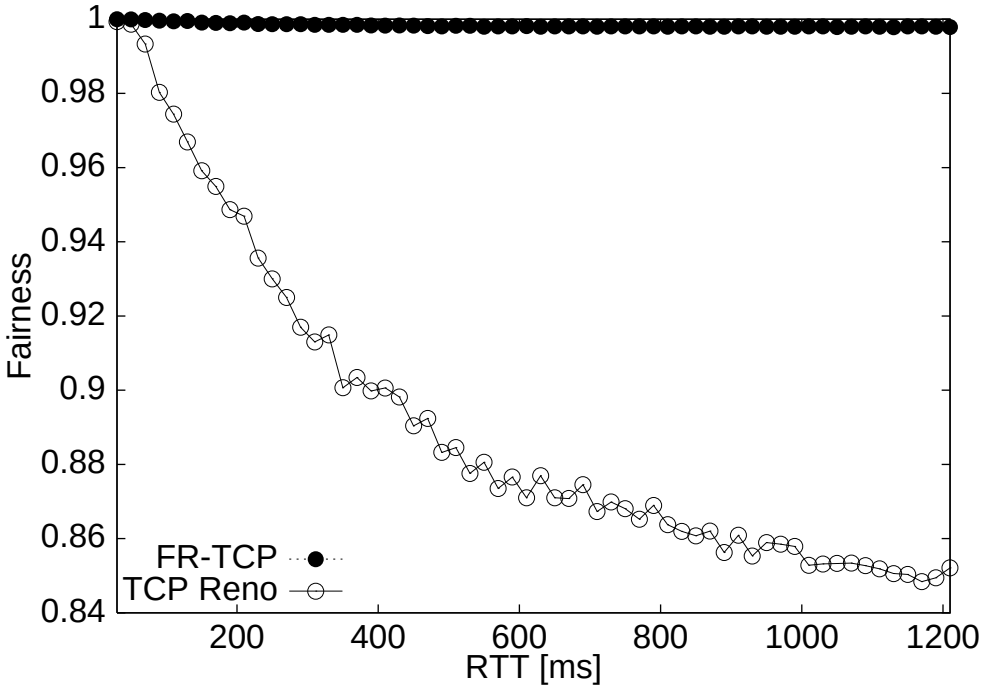

図 4.11: RTT の変化に対する帯域割当ての公平性

集約以外のコネクションに影響を与えることなく集約内の公平性を向上させることができると考えられる.

更に FR-TCP の効果を検証する評価として，FR を固定したままでリンクの最大帯域量を変化させた場合の帯域割当ての公平性を評価する.

シミュレーション条件を表 4.4 にまとめる.

図 4.12 にリンクの最大帯域量と帯域割当ての公平性の関係を示す.

図のように，リンクの最大帯域量が大きくなるにつれて F の値は減少しており，リンクの帯域量が 50 Mbit/s 以上の場合には FR-TCP と TCP Reno の間の性能にほとんど差がないことがわかる.

これは，FR-TCP は FR 以上の帯域利用に関してはそのアルゴリズムに何の修正も行っ

表 4.4: シミュレーション条件 (公平性 3)

集約内のコネクション数	6
契約帯域量	9 Mbit/s
FR	1.5 Mbit/s
RTT	40 ms, 80 ms, 120 ms, 160 ms, 200 ms, 240 ms
TCP 輻輳制御機構	全て FR-TCP を使用
リンクの最大帯域量	10 〜 100 Mbit/s

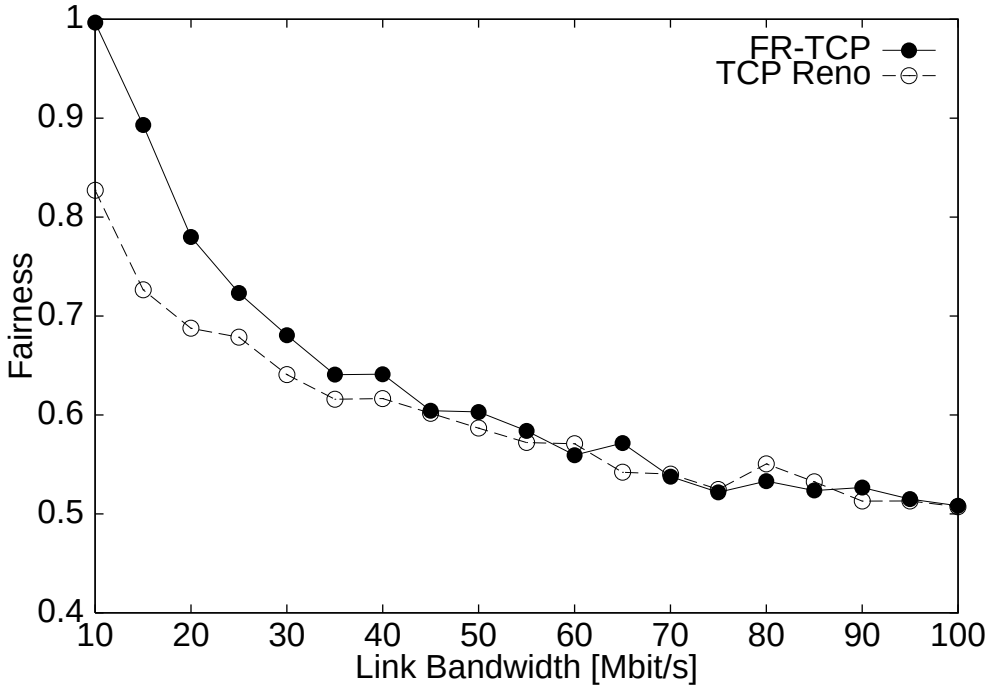

図 4.12: リンクの最大帯域量と帯域割当ての公平性の関係

ておらず，従来の TCP と同様の動作をすることが原因として挙げられる．FR-TCP は FR までは積極的にデータを送信するが，FR 以上のデータ送信に関してはネットワークへ与える影響を考慮して従来の TCP と全く同じ制御を行う．そのため，今回の評価のように契約帯域量以外に余剰な帯域が大量の存在する場合，FR-TCP は余剰帯域の割当ての公平性を改善することができないので，RTT の短いコネクションほどより多くの余剰帯域を獲得してしまう．結果として，余剰帯域に関しては，不公平に帯域が割当てられてしまっている．今回の評価結果は，そのような余剰帯域利用に関する不公平性が，FR-TCP による改善効果上回ったことが反映されていると考えられる．

4.3.3 複数契約環境への対応

次に，帯域保証契約が複数 (契約 A，契約 B の 2 契約) 締結されている環境における FR-TCP の性能を評価する．

シミュレーション条件を表 4.5 にまとめる．

表 4.6 に各契約全体のスループットの測定結果 ($Rate$) と，それぞれの契約についての帯域割当ての公平性 (F) の計算結果をまとめる．なお，表には比較のために，全てのフローが TCP Reno を利用した場合の測定結果，計算結果についても併せて示す．

表のように，FR-TCP を利用した場合，各契約は TCP Reno を用いた場合よりも帯域割当ての公平性が改善されていることがわかる．

表 4.5: シミュレーション条件 (複数契約環境)

集約内のコネクション数	契約 A, B ともに 6
契約帯域量	6 Mbit/s, 3 Mbit/s
FR	1 Mbit/s, 0.5 Mbit/s
RTT	契約 A : (40 ms, 80 ms, 120 ms, 160 ms, 200 ms, 240 ms) 契約 B : (40 ms, 80 ms, 120 ms, 160 ms, 200 ms, 240 ms)
TCP 輻輳制御機構	全て FR-TCP を使用

表 4.6: 測定結果 (複数契約環境)

SLA #	CIR (Mbit/s)	$Rate_{FR-TCP}$ (Mbit/s)	F_{FR-TCP}	$Rate_{Reno}$ (Mbit/s)	F_{Reno}
A	6.000	6.541	0.9984	6.343	0.8906
B	3.000	3.459	0.9962	3.657	0.8949

4.3.4 コネクション数の動的変化への対応

次に，シミュレーション中にコネクション数が動的に変化する環境における FR-TCP の性能を評価する．

シミュレーション条件を表 4.7 にまとめる．

図 4.13 に Flow #1 の送信側におけるスループットの時間変化を示す．

図より，FR-TCP はコネクション数が変化することによって FR が動的に変化する場合でも，その変化に速やかに対応して送信帯域を変化させていることがわかる．またコネクション数が増加した場合，Flow #1 はコネクション数が増加した直後は FR を大きく上回る帯域で送信を行っている．これは，送信を開始した TCP コネクションの送信帯域が FR 付近に安定するまで，Flow #1 がそのコネクションに割当てられるべき帯域を利用して送信を行っていることが原因であると考えられる．その後，送信を開始した TCP コネクションが FR に合わせて安定したデータ送信を行うため，Flow #1 は FR に近い速度で

表 4.7: シミュレーション条件 (コネクション数変化)

集約内のコネクション数	1 から 20 の間で 20 秒毎にランダムに変化 (Flow #1 は常に送信を行う)
契約帯域量	9 Mbit/s
FR	20 秒毎に変化
RTT	全て 40 ms
TCP 輻輳制御機構	全て FR-TCP を使用

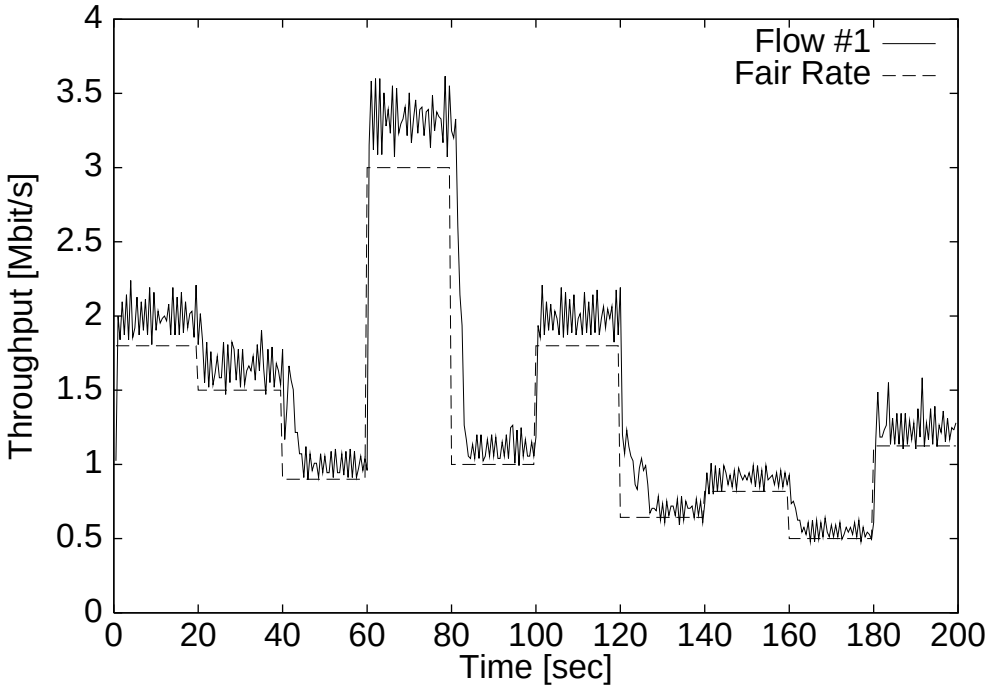

図 4.13: 送信側スループットの時間変化

表 4.8: シミュレーション条件 1 (異種混在環境)

集約内のコネクション数	6
契約帯域量	9 Mbit/s
FR	1.5 Mbit/s

データ送信を行っていることがわかる.

4.3.5 異種混在環境への対応

本節では，TCP Reno と FR-TCP が集約コネクション内に混在する環境での FR-TCP の性能を評価する.

シミュレーション条件を表 4.8，表 4.9 にまとめる.

表 4.10 および図 4.14 に各コネクションのスループット $Rate_{hetero}$ の測定結果をまとめる. なお，比較のために，全てのフローが TCP Reno を利用した場合のスループット $Rate_{Reno}$ の測定結果についても併せて示す.

表より，FR-TCP は混在環境においても FR 以上のスループットを実現できていることがわかる．更に，F についても TCP Reno のみを用いた場合よりも高い値を示していることから，FR-TCP は既存の TCP との混在環境においても帯域割当ての公平性を改善す

表 4.9: シミュレーション条件 2 (異種混在環境)

flow #	1	2	3	4	5	6
RTT (ms)	40	120	240	40	120	240
condition A	TCP Reno			TCP Reno		
condition B	FR-TCP			TCP Reno		

表 4.10: 測定結果 (異種混在環境)

	$Rate_{hetero}$	$Rate_{Reno}$
Total 9.0	9.9999 (Mbit/s)	9.99955 (Mbit/s)
F	0.883734	0.828779

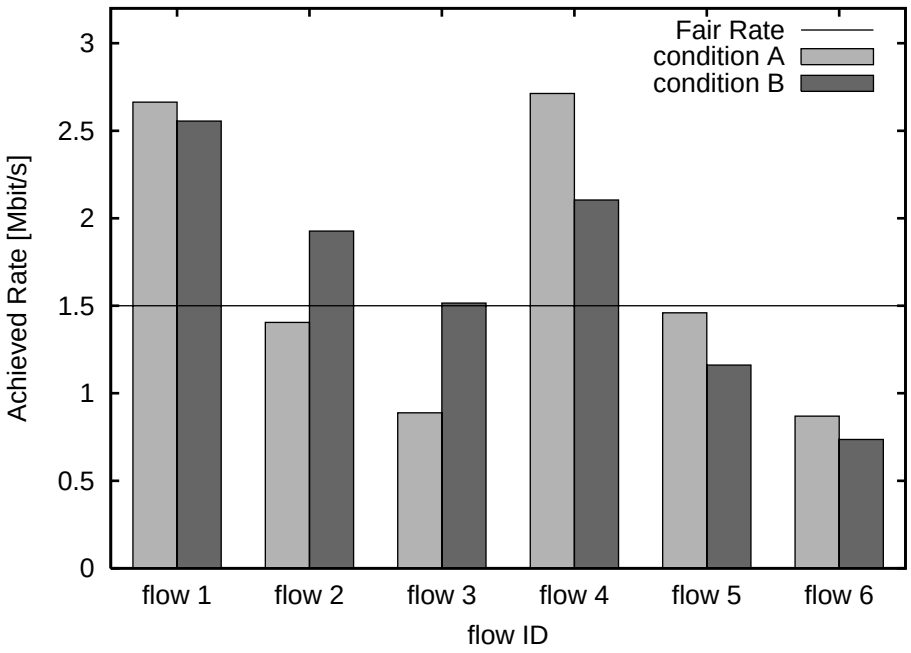

図 4.14: 測定結果 (異種混在環境)

ることができると考えられる．

4.3.6 シミュレーション評価のまとめ

以上の結果より，FR-TCP は集約コネクション内の帯域割当ての公平性を改善し，様々な環境において FR 以上の帯域で常にデータ転送を行うことができると考えられる．しかしながら，図 4.12 が示すように，FR-TCP を用いてもコネクション間の帯域割当ての公平性を改善できない場合が存在する．

このように，FR-TCP の制御では集約コネクションに完全な公平性を提供することができない．しかし，FR-TCP の機能は契約コネクション以外のコネクションへの影響や，既存 TCP と混在する環境での動作を考慮した上で提案されている．そのため，余剰帯域の獲得に関しては変更を加えず，各コネクションが最低 FR 以上でデータ送信を行うことによって最低限確保されるべき公平性の実現を目指している．そのような観点から評価した場合，本章で行った全ての測定において FR-TCP は常に FR 以上の帯域で送信を行っていることから，本章の結果は FR-TCP が公平性の点で目的を達成していることを示している．

4.4 まとめ

本章では AF PHB を利用する Diffserv 帯域保証サービスにおける集約コネクション内の帯域割当ての不公平性の問題を改善するために，エッジルータで管理される公平な帯域分配値を利用する TCP 輻輳制御機構である FR-TCP を提案した．

FR-TCP 輻輳制御機構では 2 点の提案を行った．1 点目として，FR を送信側ホストへ通知する機構についての提案を行った．この通知機構では，契約ホストにおいて FR が計算され，ポリシーサーバを介してエッジルータに FR が通知される．エッジルータにおいて通知された FR をパケットヘッダに記入し，その値を ACK パケットにコピーすることによって送信側へ通知する仕組みとなっている．2 点目として，送信側ホストにおいて通知された FR を考慮した送信帯域の制御を行う輻輳ウィンドウ制御アルゴリズムを提案した．このアルゴリズムでは，通知された FR に対応する値に *ssthresh* を設定し，更に ECN 機構を利用することで，FR 以上の帯域での安定したデータ送信を実現している．以上のように，本提案はネットワークノードとエンドホストが連携することにより，帯域割当ての公平性を向上させる提案である．

提案機構について，コンピュータシミュレーションを用いてその有効性を検証した．まず帯域割当ての公平性についての評価を行った結果，FR-TCP は集約されているコネクションの RTT，またリンク全体の帯域幅に関わらず，常に各コネクションに含まれる TCP コネクションが FR 以上の帯域でデータ送信を行い，既存の TCP Reno と比較して帯域割当ての公平性を大きく改善することがわかった．次に帯域保証契約が複数締結された環境についての評価を行った結果，FR-TCP は各契約の帯域割当ての公平性を従来方式よ

りも改善することができることが確認できた．また，コネクション数が動的に変化する環境について FR-TCP の性能を調べた結果，FR-TCP はコネクション数の変化に伴って変動する FR に速やかに対応し，柔軟に送信帯域を変化させることができることが確認できた．更に FR-TCP と既存の TCP が混在する環境について評価を行った結果，FR-TCP はそのような異種混在環境においても十分にその性能を発揮し，帯域割当ての公平性を改善することが示された．

以上のように， FR-TCP は従来の TCP の場合と比較して，想定したあらゆる場合について集約全体の帯域使用量を抑える事なく，かつ帯域割当ての公平性について大きな改善が得られることがわかった．よって，提案機構は AF PHB を利用する Diffserv 帯域保証サービスを集約コネクションに適用する場合に，集約コネクション内の RTT に関わらずより公平なデータ転送を行うことを可能とし，同一のサービスを利用するコネクションにより均等にサービスを享受させることに関して非常に有効な方法であるといえる．

第5章　エンドホストの改善による遅延の増大，広帯域化への対応

5.1 まえがき

本章では，ネットワークにおいて標準的に用いられている TCP Reno を用いて広帯域データ転送を行った場合の，リンク帯域を効率よく利用できないという問題に着目し，エンドホストのみの改善を行い解決することを目的とし，TCP Sapipa を提案する [48, 49, 50, 51, 52].

公平性，効率性の問題点に対し，TCP の輻輳制御方式において様々提案され改善が図られている．それらの提案は，大きく分けてパケット廃棄の有無のみでネットワークの輻輳状態を把握するものと，パケット廃棄の有無のほかに RTT の変動からキューイング遅延を計算し，ネットワークの輻輳状態を把握するものも提案されている．後者の代表例として TCP Vagas[53, 54, 55]，FAST TCP[56] などが提案されている．しかしながら，RTT の変動からキューイング遅延を計算する際，リンクのホップ数によって値が大きく異なり，輻輳状態を正確に把握することが困難であると考えられる．さらに，広帯域転送を行う際に，送信者は各々のパケットに対して ACK の受信，輻輳ウィンドウの変動，パケットの送信を行わなければならず，多くの負荷がかかる．そのため，それらの作業に加えてキューイング遅延を計算することで，エンドホストにおける処理量が膨大になり，ネットワークの転送容量を十分利用できない可能性が考えられる．

広帯域転送のための TCP としては HSTCP, STCP のほかに FAST TCP が提案されているが，HSTCP, STCP および FAST TCP は TCP Reno との親和性に問題がある．

TCP Reno との親和性を持つ輻輳制御方式の提案として，TFRC[57], Binomial congestion control[58] やその他多くの AIMD mechanisms[59, 60] が，提案されているが，広帯域ネットワークで効率よく転送することができないという問題点も抱えている．

そこで本章では以上の問題点をエンドホストでかつパケット廃棄の有無のみで解決すべく TCP 輻輳制御機構を提案し，広帯域ネットワークで効率よく転送可能な方式について考察を行う．本方式により，広帯域データ転送に適すること，TCP Reno との親和性との問題を改善すること，コネクション間に存在する RTT の差異による不公平な帯域割当ての問題を改善する．

5.2 関連研究

本章では，本研究に関連する研究，HSTCP[23, 61, 62]，STCP[24] について述べ，その問題点について述べる．

5.2.1 既存の広帯域転送のための TCP 輻輳制御方式

5.2.1.1 HSTCP

TCP Reno が高いスループットを得ることができないのは，輻輳ウィンドウの増加の幅が RTT ごとに 1 セグメントと非常に小さいにもかかわらず，パケット廃棄を検出した際に輻輳ウィンドウを 1/2 以下へと大きく減少させるために，輻輳ウィンドウがなかなか

大きくならないことが原因である．そこで，HSTCP が提案された．HSTCP が輻輳ウィンドウを増減させるアルゴリズムは，基本的には TCP Reno と同じであるが，輻輳回避フェーズにおいて，1 RTT ごとの輻輳ウィンドウの増加幅と，重複 ACK の受信によってパケット廃棄を検出した際の輻輳ウィンドウの減少幅が異なる．

HSTCP では，W_{Low}，W_{High} と P_{High} の 3 つのパラメータを用いる．TCP の互換性を考えて，輻輳ウィンドウが W_{Low} 以下のときは，HSTCP では TCP Reno と同じ挙動をとるようにし，W_{Low} を超えた場合には，HSTCP の対応を行うようにする．ここでは，廃棄率が 10^{-3} に対応する TCP の値としての $W_{Low} = 38[segments]$ とする．これは 45.6 Kbit/s 程度のスループットに値する輻輳ウィンドウである．

HSTCP の対象とする範囲の上限として，平均輻輳ウィンドウ $W_{High} = 83000$ とそれを実現するのに必要とされる廃棄率 $P_{High} = 10^{-7}$ を設定する．これは，RTT=100 ms としたときの 10 Gbit/s に耐えうる輻輳ウィンドウで，廃棄率が 10^{-7} の時に HSTCP が 83000 の平均輻輳ウィンドウになる．

簡単のため，HSTCP の W と p の関係を示す関数として，対数グラフにおいて線形になるような性質にした．これにより，W が W_{Low} よりも大きな値では，以下のような関数になる．

$$W = (\frac{p}{P_{Low}})^S W_{Low} \tag{5.1}$$

$$S = \frac{logW_{High} - logW_{Low}}{logP_{High} - logP_{Low}} \tag{5.2}$$

$$logW = \frac{logW_{High} - logW_{Low}}{logP_{High} - logP_{Low}}(logp - logP_{Low}) + logW_{Low} \tag{5.3}$$

W と p の関係式は，P_{Low} に対しての W_{Low} に対応する廃棄率と S から成り立ち，対数をとることで線形の式になることが観察される．

文献では，$W_L = 38$ としている．また 10 Gbit/s に耐えうる輻輳ウィンドウとして，$W_H = 83000$ を設定し，その時の廃棄率 $P_H = 10^{-7}$，減少値 $b_H = 0.1$ としている．以上の値の場合には p，輻輳ウィンドウ ($\overline{W}_{hs}$) とスループット (T_{hs}) の間の関係は式 (5.4)，式 (5.5) で示される．

$$\overline{W}_{hs} = \frac{0.12}{p^{0.835}} \tag{5.4}$$

$$T_{hs} = \frac{Size_{seg}}{RTT}\frac{0.12}{p^{0.835}} \tag{5.5}$$

TFRC[57] の方程式に基づいた輻輳制御機構の考え方を用いて，HSTCP の W と p の関係関数から，TFRC 輻輳制御機構に置き換えることができる．次に，TCP として用いるためには，HSTCP の W と p の関係関数を加算的な増加パラメータと乗法的減少のパラメータに置き換える必要がある．HSTCP の W と p の関係関数からは 1 RTT あたり 1 セ

グメントの加算的増加と輻輳ウィンドウを半減させる乗法的減少では実現できないため，増加か減少のパラメータを変える必要がある．HSTCP は増加と減少のパラメータを変え，増加と減少を繰り返し制御する．

したがって，HSTCP は輻輳がないときに 1 RTT あたり $a(W)$ セグメントの増加と輻輳時に $(1 - b(W))$ セグメントに減少させるとする．

$$W = W + \frac{a(W)}{W} \tag{5.6}$$

$$W = (1 - b(W))W \tag{5.7}$$

TCP Reno は W の値に関係なく，$a(W) = 1$，$b(W) = 1/2$ である．

$W = W_{High}$ のときに P_{High} の廃棄確率になる必要があるため，$a(W)$ と $b(W)$ の間に以下の関係式が必要である．

$$a(W) = 2W_{High}^2 P_{High} \frac{b(W)}{2 - b(W)} \tag{5.8}$$

$W = W_{High}$ の時の $b(W)$ の減少パラメータを設定する際，$W = W_{High}$ における減少値 $Decr_{High}$ を用いる．その後式 (5.8) を用いて $W = W_{High}$ の時の $a(W)$ の増加パラメータを導き出す．たとえば，$P_{High} = 10^{-7}$，$W_{High} = 83000$ のとき，$Decr_{High} = 0.1$，つまり $b(83000) = 0.1$ とする．このとき，$a(83000) = 72$ となり，72 セグメントの増加は 0.1% 以下になる．

この 10% の減少はより深刻な輻輳に応じて送信帯域を制限する際 TCP の ACK クロッキングと用いた場合にうまく働く．より急激な減少を行う場合，輻輳でないときの RTT あたりの輻輳ウィンドウでの帯域獲得の強度が強い増加を必要とし，深刻な輻輳を起こす．

W_{Low} のとき $b(W) = \frac{1}{2}$ で，W_{High} のとき $b(W) = Decr_{High}$ とすると，$b(W)$ のほかの値は以下のように線形に変化し，それに伴い $a(W)$ も決まる．ここで，P_W とは輻輳ウィンドウ W の時の廃棄率で，式 (5.2) で求まる．

$$b(W) = \frac{(Decr_{High} - 0.5)(log(W) - log(W_{Low}))}{log(W_{High}) - log(W_{Low})} + 0.5 \tag{5.9}$$

$$a(W) = 2W^2 P_W \frac{b(W)}{2 - b(W)} \tag{5.10}$$

以上より，HSTCP は高帯域のリンクを利用する場合でも，輻輳状態に達しやすくするように廃棄率を高くしているために高い利用率で転送できる．

5.2.1.2 STCP (Scalable TCP)

STCP は TCP Reno の輻輳制御アルゴリズムに変更を加えたものであり，HSTCP と同様，広帯域転送ネットワークにおいて TCP の性能を向上させる．STCP の輻輳ウィンドウの増加関数，減少関数は式 (5.11)，式 (5.12) と提案されている．

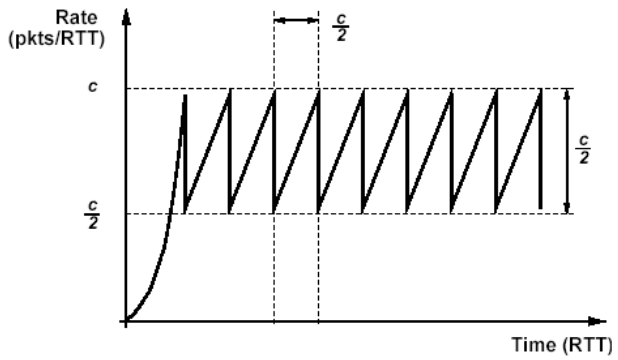

図 5.1: TCP Reno のウィンドウ変化

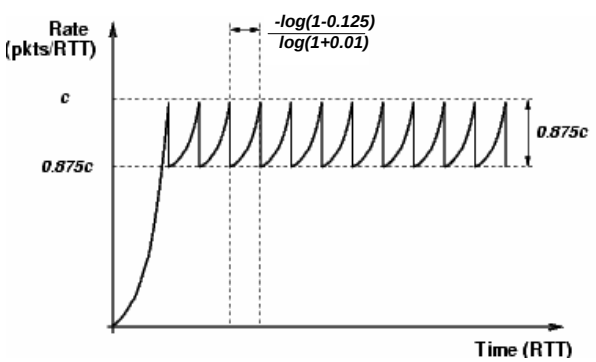

図 5.2: STCP のウィンドウ変化

表 5.1: TCP Reno と STCP のリカバリ時間の関係

Rate	TCP Reno リカバリ時間	STCP リカバリ時間
1 Mbit/s	0.8 s	1.3 s
10 Mbit/s	8.5 s	1.3 s
100 Mbit/s	1 mins	1.3 s
1 Gbit/s	14 mins	1.3 s
10 Gbit/s	1 hrs 21 mins	1.3 s

$$W = W + 0.01 \qquad \text{(ACK 受信時)} \tag{5.11}$$

$$W = W - 0.125W \qquad \text{(パケット廃棄時)} \tag{5.12}$$

図 5.1，図 5.2 は TCP Reno と STCP のスケーラブルな性質についての重要な違いを示している．TCP Reno のリカバリ時間は送信帯域と RTT に比例しているが，STCP のリカバリ時間は RTT だけに比例しており，W に比例しないため，広帯域 IP ネットワークに対してスケーラブルな方法になっている．

この STCP の実際の環境での動作を調べるために，TCP コネクションがパケットサイズ 1500 Byte，RTT = 100 ms である場合を考える．以下の表は様々な送信帯域での TCP Reno の輻輳制御と STCP 輻輳制御のおおよそのリカバリ時間である．STCP のリカバリ時間は帯域に依存せず，一定であることがわかる．

ここで，リカバリ時間について考察する．輻輳ウィンドウの増加関数，減少関数を式 (5.13)，式 (5.14) とすると，

$$W = W + a \qquad \text{(ACK 受信時)} \tag{5.13}$$

$$W = W - bW \qquad \text{(パケット廃棄時)} \tag{5.14}$$

1 RTT でウィンドウの増加量は Wa であり，$1+a$ 倍になる．X RTT 経過した時は，$(1+a)^X$ 倍になる．そのため，輻輳による廃棄によって減少した輻輳ウィンドウ $(1-b)W$ が，X RTT 時間後，廃棄前の輻輳ウィンドウに戻るとすると，下記の式に示すような計算式になる．

$$(1+a)^X(1-b)W = W \tag{5.15}$$

$$X = -\frac{log(1-b)}{log(1+a)} \tag{5.16}$$

$a = 0.01$, $b = 0.125$ の時は，$X = 13.42$ になる．したがって，$\tau = X \times RTT$ より，$\tau = 13.42RTT$ である．また，輻輳ウィンドウ $(\overline{W}_{sca})$ とスループット (T_{sca}) の間の関係は式 (5.18)，式 (5.19) で示される．

$$\overline{W}_{sca}X = \frac{1}{p} \tag{5.17}$$

$$\overline{W}_{sca} = \frac{0.0745}{p} \tag{5.18}$$

$$T_{sca} = \frac{Size_{seg}}{RTT}\frac{0.07}{p} \tag{5.19}$$

となる．

5.2.2 関連研究における問題点

広帯域ネットワークで効率よく帯域制御行うにあたり，新たな TCP 輻輳制御機構は以下の公平性を考慮する必要がある．

- 既存の TCP コネクションとの異種混在状況における公平性（TCP Reno との親和性）
- RTT の異なるコネクション間の公平性

現在普及している TCP 輻輳制御機構は，TCP Reno である．広帯域のために新たに考案された TCP は，各エンドホストが使用する輻輳制御機構を変更して初めて，使用可能になる．そのため，TCP Reno と新たな TCP 輻輳制御機構が混在する環境は避けられない．この状況において，新たな TCP 輻輳制御機構は，TCP Reno のコネクションとの異種混在状況において，既存の TCP の帯域に大きな影響を与えることは望ましくない．つまり，新たな TCP 輻輳制御機構の特徴として，TCP Reno との親和性は重要な課題となってくる．

2 章で指摘したように，遅延が増大したネットワークにおいては，各コネクションの RTT の差が広くなるため，RTT の異なるコネクション間の公平性を維持する必要がある．

この問題は式 (2.5)，式 (2.9)，式 (2.11) より TCP Reno，HSTCP と STCP のスループットは RTT に反比例していることからも RTT の異なるコネクション間で生じることがわかる．TCP Reno の場合，半減した輻輳ウィンドウが回復するまでに，$W/2$ RTT 必要で，そのリカバリ時間は輻輳ウィンドウと RTT に比例している．そのため，RTT の異なるコネクション間の公平性を実現できない．また，リカバリ時間が RTT に比例している場合，特に広帯域ネットワークでは，リカバリ時間が長くなるために帯域を効率よく利用できないという問題も生じてくる．そのために，RTT に依存しないウィンドウ増減が必要である．

5.3 提案

現行の TCP および関連研究の TCP では，以下の 3 つの課題が同時に解決されていない．

- 広帯域ネットワークで帯域利用効率が悪い．
- TCP Reno との親和性がよくない．
- RTT の異なるコネクションでは帯域が分散する．

そこで，本章では，TCP の送信側で制御可能な方法で，広帯域ネットワークで帯域利用効率が高く，TCP Reno との親和性を改善し，RTT の異なるコネクションでは公平に帯域を獲得できる方式，TCP Sapipa を提案する．ここで広帯域ネットワークの具体的指標として，1 Gbit/s のネットワークを想定する．現在のネットワークの帯域が数十〜数百 Gbit/s になってきているため，1 コネクションが使用できる帯域として考えたときの最大でも 1 Gbit/s であると想定し，設定した．

5.3.1 提案方針

輻輳ウィンドウをネットワークの帯域に適した値で推移させ，効率のよい転送を行うためには，パケット廃棄に対する輻輳制御によって減少した輻輳ウィンドウを，リンク帯域にあった大きさに回復する時間を輻輳が起き過ぎない範囲で短くすることが必要である*．この時間を短くすると，輻輳によって生じるパケット廃棄の周期も短くなり，結果的に廃棄率が高くなる．その結果，広帯域データ転送下の大きな $\overline{W}$ においても，p を高く維持でき，効率のよい転送が可能となる．また，本提案では，帯域割当ての不公平，帯域の使用効率の問題は，TCP の氏魚の中で一番長いフェーズの TCP の輻輳回避フェーズにあると考えているため，TCP の輻輳回避フェーズの制御において提案を行っている．スロー

*効率のよい転送を行う手段として，十分な輻輳ウィンドウに達するまでの輻輳ウィンドウの積分値を大きくすることも考えられる．この点に関しては別途考察が必要であるため，本章では回復する時間のみに着目した．

スタートフェーズの制御，タイムアウトの制御は，既存の TCP Reno と同じ方式を採用する．

TCP Reno との親和性を重視するために，TCP Reno と同様の AIMD の法則を一般化した式 (5.20)，(5.21) から TCP Sapipa のウィンドウの増減を検討する．

$$W = W + aW^{\alpha} \quad \text{(ACK 受信時)} \tag{5.20}$$

$$W = W - bW^{\beta} \quad \text{(パケット廃棄時)} \tag{5.21}$$

ここで，aW^{α}，bW^{β} はそれぞれ輻輳制御を行う際の増加量と減少量である．α，β は輻輳ウィンドウの指数であり，a，b は増加量，減少量の係数である．以下では，これらの一般式の各パラメータ α，β，a，b を検討し，TCP Sapipa の輻輳制御方式を決定する．

5.3.2 α，β の検討

広帯域データ転送効率を行うためには，α と β の値をまず検討する必要がある．α は，増加量の指数であり，β は減少量の指数である．そのため，TCP の輻輳制御方式の帯域獲得の強度を示す基準として，$\beta - \alpha$ が使用可能である．$\beta - \alpha$ が大きい場合，帯域獲得の強度が弱く，$\beta - \alpha$ が小さい場合，帯域獲得の強度が強い．

TCP Reno では，AIMD 輻輳制御方式を採用し，$\beta = 1$，$\alpha = -1$ であり，$\beta - \alpha = 2$ となっている．TFRC[57] や Binomial 輻輳制御方式 [58] は $\beta - \alpha = 2$ を行うことで，TCP Reno との親和性を実現している．しかし，$\beta - \alpha = 2$ である輻輳制御方式で広帯域データ転送を行おうとすると，ウィンドウ減少量が大きいのにもかかわらず，ウィンドウ増加量が小さく，効率のよい転送が行えない．

$\beta - \alpha = 1$ では，式 (5.22) で示すように，τ が W の関数ではないため，リカバリ時間がそのコネクションの輻輳ウィンドウに依存しない．

$$\begin{aligned} \tau &= \frac{bW^{\beta}}{aW^{\alpha+1}} RTT = \frac{bW^{\alpha+1}}{aW^{\alpha+1}} RTT \\ &= \frac{b}{a} RTT \end{aligned} \tag{5.22}$$

式 (5.22) は以下の方法で導かれる．τ = (ウィンドウ減少量 / ウィンドウ増加速度) である．式 (5.20) より，ウィンドウ増加量は 1 ACK 受信あたり aW^{α}，つまり 1 RTT あたり $aW^{\alpha+1}$ である．したがってウィンドウ増加速度は $aW^{\alpha+1}/RTT$ である．式 (5.21) より，ウィンドウ減少量は bW^{β} であり，$\beta = \alpha + 1$ であるため，$bW^{\alpha+1}$ となる．以上より，$\beta - \alpha = 1$ では，リカバリ時間が輻輳ウィンドウに依存しないため，広帯域データ転送時にも短時間でウィンドウの回復が行え，広帯域データ転送に適している．よって，本提案では $\beta - \alpha = 1$ を採用する．

また，α，β の値それぞれにも，特性がある．$\alpha = -1$ のとき，RTT あたりの増加量が定数になり，TCP Reno と同様，加算的増加になる．そのため，TCP Reno よりも帯域獲

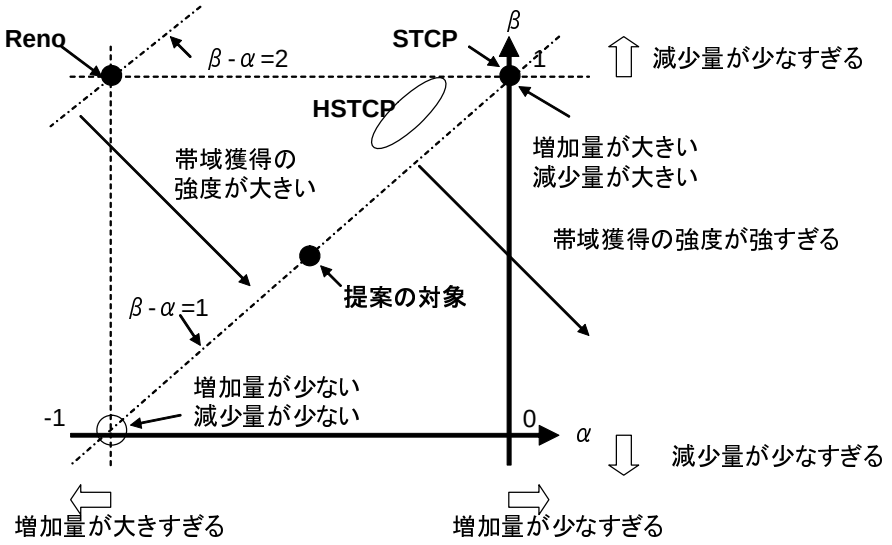

図 5.3: α，β と TCP の特性の関係

得の強度が弱い $\alpha < -1$ は，広帯域転送に適していない．$\alpha = 0$ のとき，RTT あたりの増加量が W に比例する．輻輳ウィンドウが大きい場合，RTT あたりの増加量が大きくネットワークの輻輳の原因となり，a が小さく，輻輳ウィンドウが小さい場合には，増加速度が小さくなり，帯域使用効率が落ちる．その結果，広帯域および狭帯域の両方の転送には適していない．$\beta = 0$ のとき，減少量が定数になり，輻輳ウィンドウが大きい場合には減少率が小さくなり，小さい場合には減少率が大きくなり，広帯域および狭帯域の両方の転送には適していない．$\beta = 1$ のとき減少量が W に比例するため，TCP Reno と同様，定率減少になる．したがって $\beta > 1$ の W が大きくなるにつれて減少率が大きくなる場合は，広帯域転送に適していない．

$\beta - \alpha = 1$ の場合，まずは，$(\alpha, \beta) = (-1, 0), (0, 1)$ が候補に挙がる．$(\alpha, \beta) = (-1, 0)$ は，減少量が定数であるため，輻輳ウィンドウが大きい場合には減少量が小さく，輻輳ウィンドウが小さい場合には減少量が大きくなる場合が生じ，ネットワークの輻輳に対して迅速に対応できなくなる．そのため，広帯域および狭帯域の両方のデータ転送には向かない．

STCP では $(\alpha, \beta) = (0, 1)$ を採用している．この場合，輻輳ウィンドウが大きくなると 1 RTT あたりの増加量が大きく，過度なネットワークの輻輳の原因となる．また増加係数を小さくしてしまうと，輻輳ウィンドウが小さくなった場合に増加速度が小さくなり，問題が生じる．すなわち，広帯域および狭帯域の両方のデータ転送には向かない．

$\alpha < -1$ かつ $\beta < 0$ の場合は，増加量，減少量が共に極端に小さくなってしまい，$0 < \alpha$ かつ $1 < \beta$ の場合は，増加量，減少量が極端に大きくなってしまうため，広帯域および狭帯域の両方のデータ転送には向かない．

図 5.3 に α，β と TCP の特性の関係を示す．図に示すように，$-1 < \alpha < 0$ かつ $0 < \beta < 1$ が必要である．この範囲における α，β の候補は複数ある．ここで，輻輳ウィンドウの増加量の計算は ACK ごとに行う必要があり，輻輳ウィンドウが大きくなればなるほどその

演算回数は増えてくる．また，α，β は W の指数である．したがって，α，β の値は指数として演算が容易なものが適している．さらに，$(\alpha,\beta)=(-1,0),(0,1)$ が望ましくない原因が，前者は $\beta=0$ にあり，後者は $\alpha=0$ にあるため，その問題を共に解決することが必要となる．以上より，α，β の値の候補の 1 つとして，$(\alpha,\beta)=(-1/2,1/2)$ を選択し，本提案ではこの値についての考察を行う．図 5.3 に，提案の対象となる α，β の値を示す．

$(\alpha,\beta)=(-1/2,1/2)$ では，輻輳ウィンドウの増減が平方根の計算になり，他の指数演算と比較して，計算が容易である．また，輻輳ウィンドウを，その値に呼応して，大きすぎない程度に増加または減少させることができるため，広帯域データ転送，低速転送の両方に向いている．このとき，p，$\overline{W}$ と T の間の関係は式 (5.23)，式 (5.24) で示される．

$$\overline{W} = \frac{RTT}{p\tau} = \frac{RTT}{p(\frac{b}{a}RTT)} = \frac{a}{bp} \tag{5.23}$$

$$T = \frac{Size_{seg}}{RTT}\frac{a}{bp} \tag{5.24}$$

式 (5.23) は，以下の方法で導かれる．廃棄によって減少した W が再び同じ大きさまで回復すると再度廃棄が生じるため，τ 時間に 1 度パケット廃棄が発生する．一方で，1 RTT あたり転送している平均パケット数は $\overline{W}$ であるため，τ 時間に転送したパケット数は $\tau\times\overline{W}/RTT$ である．つまり，$\tau\times\overline{W}/RTT$ 回パケットを送信したら，1 回パケットが廃棄するため，$p=1/(\tau\overline{W}/RTT)$，$\overline{W}=RTT/p\tau$ となる．

以上より，TCP Sapipa では，輻輳がないときには輻輳ウィンドウの増加量をその輻輳ウィンドウの平方根に反比例させ，輻輳時にはその輻輳ウィンドウの平方根に比例して減少させる．

5.3.3 a，b の決定

$(\alpha,\beta)=(-1/2,1/2)$ の輻輳制御方式は輻輳ウィンドウを 1 ACK 受信あたり $aW^{-\frac{1}{2}}$，つまり1 RTT あたり $aW^{\frac{1}{2}}$ だけ増加させ，パケット廃棄が生じた場合には $bW^{\frac{1}{2}}$ だけ減少させる．式 (5.24) に基づいて，$(\alpha,\beta)=(-1/2,1/2)$ の時の a，b と TCP の特性の関係を，図 5.4 に示す．要求条件は，1 Gbit/s の転送で効率よく帯域を使用できること，TCP Reno との親和性を実現すること，RTT の異なるコネクション間で公正性を実現することである．

そのため，以下のことが必要になる．

- 図 5.4 において，1 Gbit/s 転送時の平均輻輳ウィンドウである $W=8333$ の時に，低い廃棄率 (p) を実現できること．
- TCP Reno の関数の線と提案する関数が図上で乖離しないこと．
- RTT の異なるコネクション間で公正性を実現するためには，スループットが RTT に依存しないこと．

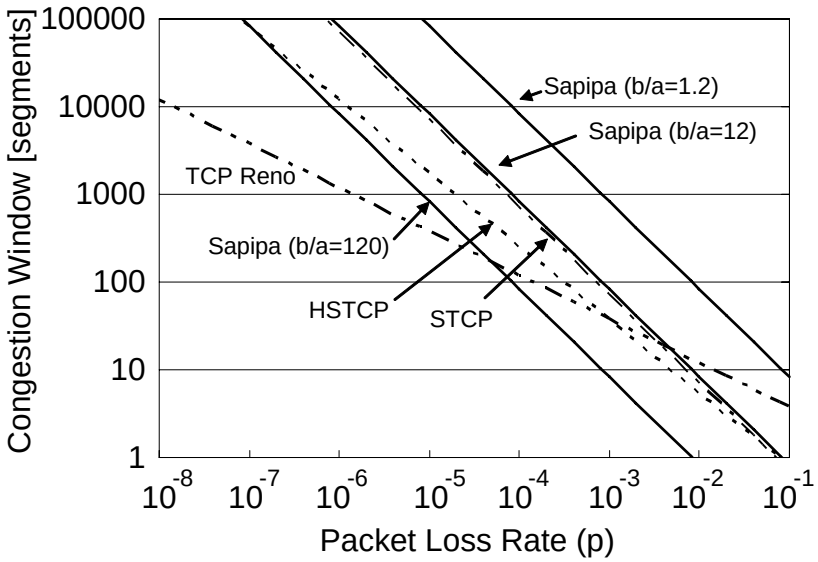

図 5.4: a，b と TCP の特性の関係

式 (5.22)，(5.24) より，平均スループットが輻輳制御フェーズの周期に反比例しているので，スループットが RTT に依存しない値にするためには，$a/b \propto RTT$ が条件となる．つまり，ウィンドウ減少量が RTT に反比例すること ($b \propto 1/RTT$)，またはウィンドウ増加量が RTT に比例すること ($a \propto RTT$) が考えられる．

ウィンドウの減少量を RTT に反比例させた場合，1 RTT あたりのウィンドウ増加量は RTT に依存しない値になり，単位時間あたりのウィンドウ増加量は RTT に反比例する．RTT の異なるコネクション間において輻輳ウィンドウの増加量，減少量が異なり，RTT 間で依然，不公平が生じることが予想される．ウィンドウの増加量を RTT に比例させた場合，1 RTT あたりのウィンドウ増加量が RTT に比例し，単位時間あたりのウィンドウ増加量は RTT に依存しない値になる．その結果，RTT の異なるコネクションの輻輳ウィンドウの増加量，減少量も一定になり，RTT の影響をなくすことができると考えられる．そのため，本提案ではウィンドウの増加量を RTT に比例させる方法を採用し，$a = a'RTT$ とおく．

また，a' と b の値を以下の条件を用いて選択した．$W = 83000$ (RTT=100 ms で 10 Gbit/s 転送) の時に $p = 10^{-7}$ となることが理想と考え，式 (5.23) より $b/a' = 12$ とした．これは，HSTCP の思想を取り入れ，10Gbit/s 転送時に廃棄率が 10^{-7} であれば広帯域転送と行った場合にも，十分短いリカバリ時間を実現できる値として選択した．

また，輻輳ウィンドウの減少量を HSTCP と同じ $0.1W$ として a の値を算出した場合，輻輳ウィンドウの小さな値から TCP Sapipa 独自の広帯域転送に適した輻輳ウィンドウの増減を行うことになり，TCP Reno との親和性の改善の程度が小さくなる．そのため，ウィンドウ減少量が $0.1W$ 以下であることが望ましいと考え，$0.05W$ を選択した．減少量を $0.05W$ と選択することで，広帯域での効率の良い帯域転送と狭帯域での TCP Reno と

の親和性の両方の点を重視した．以上より，$bW^{\frac{1}{2}} = 0.05W, b = 14.4$ となる．ここで，減少量が少なくなり過ぎないように $0.05W$ 以上とし，b を $\lceil 14.4 \rceil$ の 15 にした．したがって，$b/a' = 12$ より，$a' = 1.25$ となる．$a' = 1.25$，$b = 15$ の時，コネクションの RTT やその時の輻輳ウィンドウの値に関係なく，パケット廃棄が生じ輻輳ウィンドウの減少後，減少分を回復するまでに，12 秒という一定な値となる．最後に，TCP Sapipa のウィンドウの増減を式 (5.25)，(5.26) に示す．

$$W = W + 1.25RTT\ W^{-\frac{1}{2}} \text{ (ACK 受信時)} \tag{5.25}$$

$$W = W - 15W^{\frac{1}{2}} \text{ (パケット廃棄時)} \tag{5.26}$$

本提案のパケット廃棄率 p，輻輳ウィンドウ $\overline{W}_{sapipa}$ とスループット T_{sapipa} の関係を式 (5.27)，(5.28) に示す．

$$\overline{W}_{sapipa} = \frac{0.0833RTT}{p} \tag{5.27}$$

$$T_{sapipa} = \frac{0.0833Size_{seg}}{p} \tag{5.28}$$

式 (5.28) より，T_{sapipa} は RTT の値に依存しない値で計算され，異なる RTT 間での公平な帯域を実現できることがわかる．

5.3.4 TCP Reno との親和性

TCP Sapipa は輻輳ウィンドウが小さいとき，TCP Reno との親和性を重視し，TCP Reno と同等の性能にする．

式 (2.2)，(5.25) より，ウィンドウ増加量を比較した場合，TCP Reno が $\frac{1}{W}$ であり，TCP Sapipa が $1.25RTT\ W^{-\frac{1}{2}}$ である．$W \leq \frac{0.64}{RTT^2}$ の場合に，TCP Reno が TCP Sapipa のウィンドウ増加量を上回るため，TCP Reno が有利な領域では，TCP Sapipa のウィンドウ増加量を TCP Reno と同じにする．

また，式 (2.3)，(5.26) より，ウィンドウ減少量は TCP Reno が $\frac{1}{2}W$ であり，TCP Sapipa が $15W^{\frac{1}{2}}$ である．$W \leq 900$ では，TCP Sapipa が TCP Reno のウィンドウ減少量を上回るため，この場合には TCP Sapipa のウィンドウ減少量を TCP Reno と同じにする．

図 5.5 に W の違いにおける TCP Sapipa の α，β の遷移を示す．W が小さい値の時は，$\beta - \alpha = 2$ を示し，TCP Reno と同じであるため，TCP Reno との親和性を維持できる．また，W が大きくなるにつれて，$\beta - \alpha$ が 2 から，1.5，1 へ減少し，帯域の有効使用を実現するように遷移していることがわかる．

以上より，TCP Sapipa は $(\alpha, \beta) = (-1/2, 1/2)$，$(a, b) = (1.25RTT, 15)$ を採用し，式 (5.29)，(5.29) で示す輻輳制御機構になる．

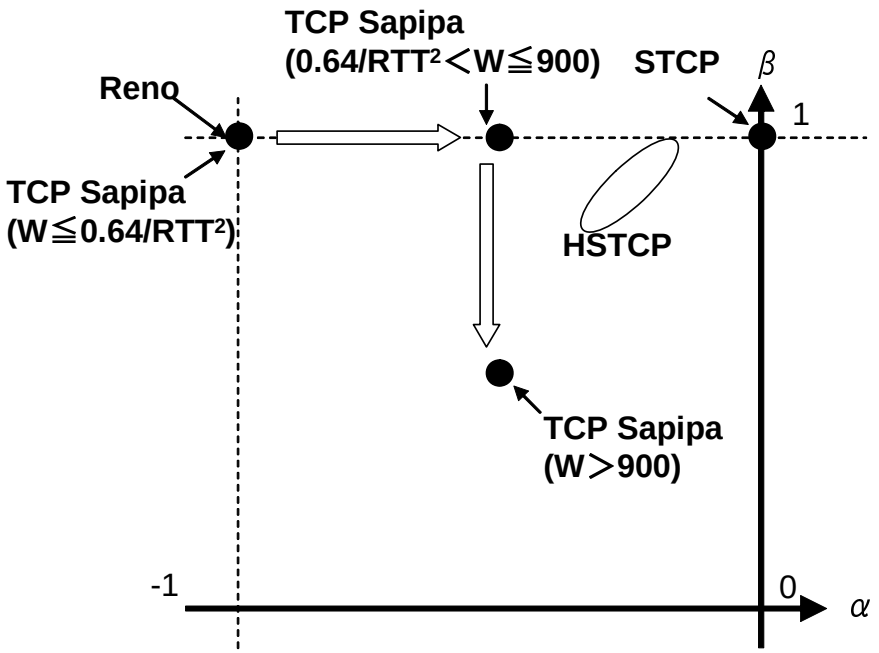

図 5.5: TCP Sapipa の α，β の遷移

$$
\begin{aligned}
W &= W + max(1.25RTT\ W^{-\frac{1}{2}}, \frac{1}{W})(\text{ACK 受信時}) \\
W &= W - min(15W^{\frac{1}{2}}, \frac{1}{2}W)\ (\text{パケット廃棄時})
\end{aligned}
$$

RTT = 100 ms の場合，本提案のパケット廃棄率 p，輻輳ウィンドウ $\overline{W}_{sapipa}$ とスループット T_{sapipa} の関係は，$W \geq 900$ のときは，式 (5.27)，(5.28) で示した関係になり，$W \leq 64$ のときは，式 (2.4)，(2.5) で示した TCP Reno の関係になる．$64 < W < 900$ のときは，$\overline{W}$ と p，T と p が対数目盛において線形に近似できると考え，式 (5.29)，(5.30) とした．

$$\overline{W}_{sapipa} = \frac{0.214}{p^{0.72}} \tag{5.29}$$

$$T_{sapipa} = \frac{2.14Size_{seg}}{p^{0.72}} \tag{5.30}$$

5.4 数値解析

本章では，帯域利用効率，TCP Reno との親和性，異なる RTT のコネクションにおける公平性の数値解析を行い，提案した TCP Sapipa の有効性を示す．

5.4.1 帯域利用効率

式 (2.4)，(2.8)，(2.10)，(5.27)，(5.29) を用いて，$Size_{seg} = 1500$ byte のときのパケット廃棄率 p と TCP Sapipa の輻輳ウィンドウの関係を図 5.6 に示す．このとき，TCP Sapipa の RTT として，50 ms，100 ms，200 ms の 3 種で示した．また，式 (2.5)，(2.9)，(2.11)，(5.28)，(5.30) を用いて，$Size_{seg} = 1500$ byte, RTT = 100 ms のときのパケット廃棄率 p と TCP Sapipa のスループットの関係を図 5.7 に示す．

図 5.6 において，ある輻輳ウィンドウにおけるパケット廃棄率を比較したとき，パケット廃棄率の高い TCP は輻輳状態に達しやすく，その輻輳ウィンドウにおいて帯域を効率よく利用できる．また，TCP Sapipa は RTT の違いに応じて，輻輳ウィンドウが変化している．これは，同一の廃棄率で比較したときに，RTT が長いコネクションほど高い輻輳ウィンドウを保つことで結果としてスループットが RTT に依存しない帯域を獲得できるからである．TCP Sapipa は HSTCP，STCP と比較して，TCP Reno との輻輳ウィンドウの差が相対的に少ないため，親和性を改善している．また，輻輳ウィンドウが大きくなるにつれて廃棄率の値が HSTCP の値に近づいていることから，広帯域データ転送の時には帯域利用効率も向上する．また，図 5.7 において示したスループットに関しても，輻輳ウィンドウと同じように，TCP Reno との親和性の改善，広帯域データ転送時の帯域利用効率の向上が実現していることが確認できる．一方で，式 (2.5)，(2.9)，(2.11) からわかるとおり，TCP Reno，HSTCP，STCP のスループットは RTT に反比例する．TCP Reno，HSTCP，STCP では RTT が短くなると，図 5.7 において，グラフが全体的に右に平行移動し，ある廃棄率におけるスループットは向上する．一方で，RTT が長くなると，図 5.7 において，グラフが全体的に左に平行移動し，ある廃棄率におけるスループットは低下する．

5.4.2 TCP Reno との親和性

TCP Reno との親和性を T/T_{reno} で示すことにする．T は TCP Reno とリンクを共有している TCP のスループットである．TCP Reno との親和性の値が 1 の場合，2 つのコネクションが帯域を同量ずつ分けあっていることから，親和性が高い状況であると考えられる．

TCP Sapipa と TCP Reno との親和性は，RTT = 100 ms とすると，$W \geq 900$ のときは，式 (2.5)，(5.28) より式 (5.31) で計算され，$64 < W < 900$ のときは，式 (2.5)，(5.30) より式（5.32）で計算され，$W \leq 64$ のときは TCP Reno と同じウィンドウの増減をするため，$\frac{T_{sapipa}}{T_{reno}} = 1$ となる．

$$\frac{T_{sapipa}}{T_{reno}} = \frac{0.00694}{p^{0.5}} \tag{5.31}$$

$$\frac{T_{sapipa}}{T_{reno}} = \frac{0.175}{p^{0.22}} \tag{5.32}$$

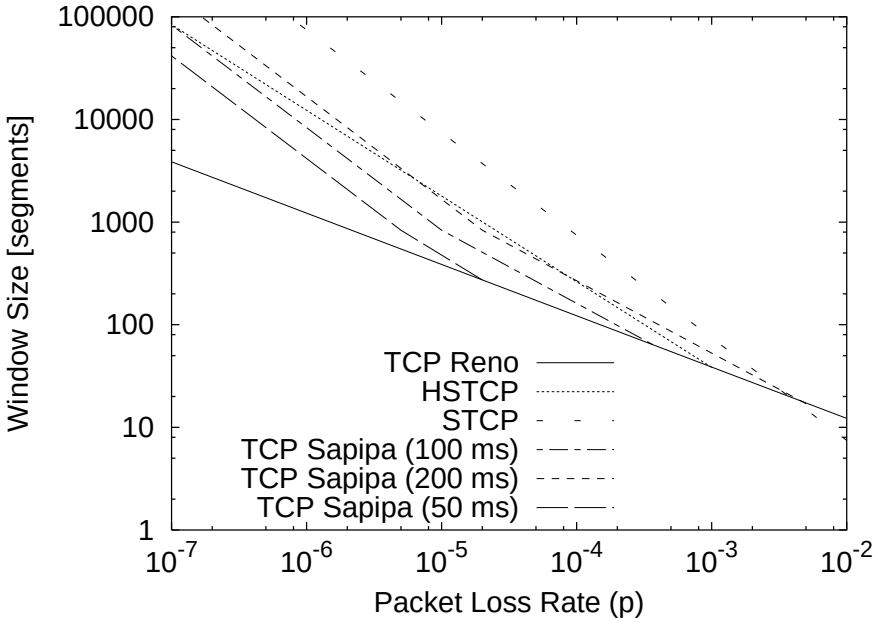

図 5.6: パケット廃棄率と輻輳ウィンドウの関係

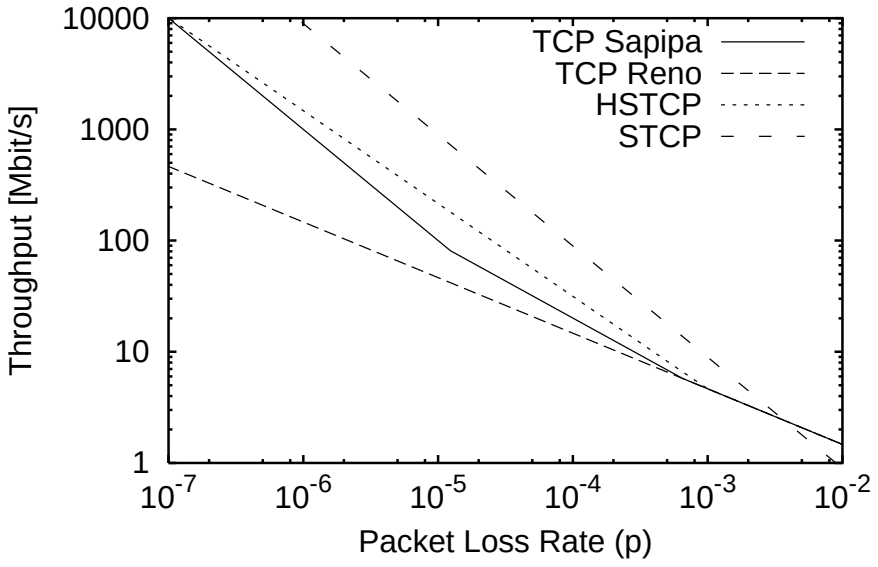

図 5.7: パケット廃棄率とスループットの関係

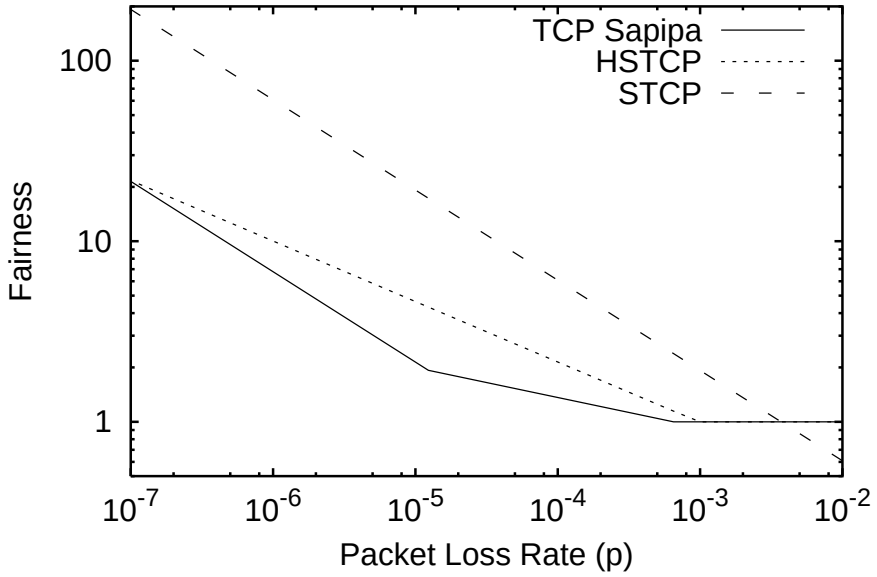

図 5.8: パケット廃棄率と TCP Reno との親和性の関係

図 5.8 にパケット廃棄率 p と，TCP Sapipa の TCP Reno との親和性の関係を示す．TCP Sapipa は HSTCP と同様，親和性が 1 以上になっており，TCP Reno と比較して，スループットが悪化する場合はなく，p が大きい場合は，親和性は 1 を示している．これは，TCP Sapipa のウィンドウ増減が，p が大きい場合は TCP Reno のウィンドウ増減を採用しており，p が小さい場合は効率のよい広帯域転送のためのウィンドウ増減を採用しているからである．また，p が小さい場合は，HSTCP，STCP よりも公平であることが，図からわかる．以上より，TCP Sapipa は，HSTCP，STCP よりも TCP Reno との親和性が高いといえる．

5.4.3 異なる RTT のコネクションにおける公平性

式 (2.5)，(2.9)，(2.11) より，TCP Reno，HSTCP，STCP のスループットは RTT に依存する値になっているのに対して，式 (5.28) より，TCP Sapipa のスループットは RTT に依存しない値になっている．そのため，TCP Reno，HSTCP，STCP はコネクションの RTT の値に応じてスループットが異なるのに対して，TCP Sapipa はスループットの変化は理論上はない．

以上で述べた HSTCP，STCP および TCP Sapipa の性能を表 5.2 に示す．

表 5.2: HSTCP，STCP および TCP Sapipa の性能

	HSTCP	STCP	TCP Sapipa
帯域利用効率 T	$\frac{Size_{seg}}{RTT}\frac{0.12}{p^{0.835}}$	$\frac{Size_{seg}}{RTT}\frac{0.0745}{p}$	$\frac{0.0833Size_{seg}}{p}$
TCP Reno との親和性	$\frac{1}{10p^{0.335}}$	$\frac{0.0608}{p^{0.5}}$	$\frac{0.00694}{p^{0.5}}$
RTT の異なるコネクション間の公平性	T が RTT に反比例するため不公平	T が RTT に反比例するため不公平	T が RTT に依存しないため公平
$\beta - \alpha$	1.2	1	$2 \rightarrow 1.5 \rightarrow 1$

5.5 評価

本章では，シミュレーションの環境，結果および考察について述べる．

5.5.1 シミュレーション環境

TCP Sapipa の有効性を確認するために，コンピュータシミュレーションによる評価を行った．

5.5.1.1 実装環境

今回のシミュレーションでは評価方法として，現在当該分野のシミュレーションに広く利用されている Network Simulator version 2 (ns-2) [39] を用いて評価した．TCP Reno, HSTCP，STCP と比較して，TCP Sapipa が効率のよい転送を行っているか，TCP Reno との親和性があるか，RTT の異なるコネクション間で公平な帯域分配を行っているかについて，評価する．

5.5.1.2 シミュレーションモデル

図 5.9 に本評価のシミュレーショントポロジーを示す．このモデルにおいて転送するコネクションの数，ボトルネックの帯域，コネクションの RTT を変動させ，sender から reciever に複数の TCP バルクデータをそれぞれ転送し，シミュレーションを行う．

5.5.1.3 シミュレーション中に用いたパラメータ

TCP のパラメータ, オプションとして，以下の項目は共通に用いた．

- TCP タイムスタンプオプション [40] を利用する．
- TCP ECN[12]，SACK[63] 機能は未使用である．

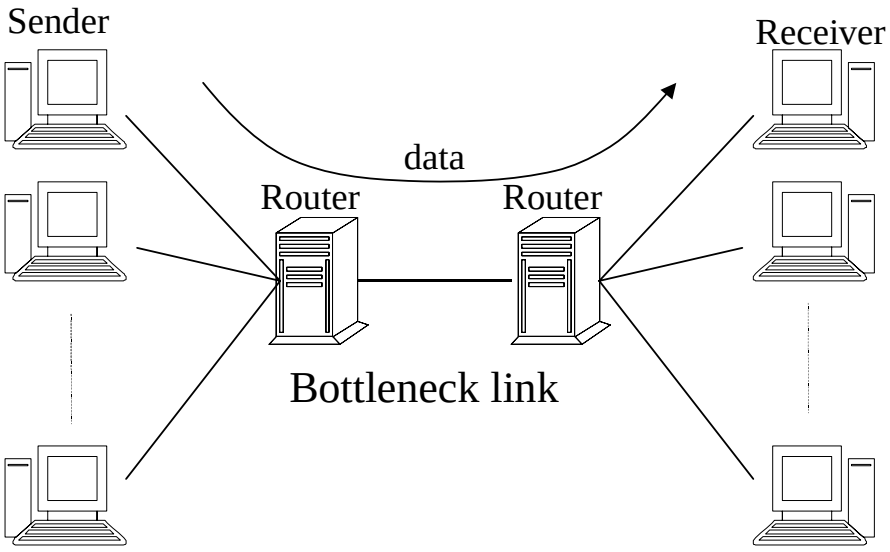

図 5.9: シミュレーションモデル

- 受信側でのパケットの処理時間は 0 秒である．
- TCP セグメントサイズ：1000 bytes

ルータに用いたパラメータは以下の通りである．

- バッファサイズは 4000 packets
- キュー管理アルゴリズムは RED[64] を使用する．
  - $max_{th} = 4000 packets$
  - $min_{th} = 400 packets$
  - $max_p = 0.2\%$

SACK オプションにより，効率のよい再送制御を実現し，各輻輳制御機構の性能の比較を行いやすい条件にした．また，RED の廃棄アルゴリズムを使用することで，複数の TCP コネクションを同時に転送した場合にも，global synchronization が起こらないようにした．

5.5.1.4 測定値

定常状態における性能を測定するために, シミュレーション開始から 20 秒が経過した時点で測定を開始した．この時点では, TCP の輻輳制御機構が安定してから十分な時間が経過していると考えられる．また，測定時間は十分転送を行ったと考えられる 500 秒間で行い，各々のコネクションにおけるスループットを測定した．

表 5.3: コネクション数の変化に対する帯域利用率の評価におけるシミュレーション条件

コネクション数	1〜8
ボトルネックの帯域	100 Mbit/s および 1 Gbit/s
RTT	100 ms

5.5.1.5 評価項目

評価項目は以下の通りである．

(1) 帯域利用率 ：コネクション数，ボトルネックの帯域を変化させて，その帯域に利用率を評価する．

(2) TCP Reno との親和性 ：TCP Reno と同時に転送した場合に，TCP Reno と帯域を公平に分配しているか，TCP Reno との公平であるかを評価する．

(3) RTT の異なるコネクション間での帯域分配 ：RTT の異なるコネクション間で帯域が公平に分配されているかを評価する．

5.5.2 帯域利用率の評価

TCP Sapipa の帯域利用率を確認するために，TCP Reno，HSTCP，STCP と比較して，シミュレーションを行った．

5.5.2.1 フロー数の変化に対する帯域利用効率

TCP Sapipa が効率のよい転送を行えていることを確認するために，ボトルネックの帯域利用効率を評価した．シミュレーション条件を表 5.3 にまとめる．

以上の環境の下，TCP Reno，HSTCP，STCP と TCP Sapipa の輻輳制御方式で，それぞれ，各コネクション数で転送を行い，帯域利用率を測定した．その結果を図 5.10，図 5.11 にそれぞれボトルネックの帯域 100 Mbit/s，1 Gbit/s の場合について示す．

図 5.10 より，ボトルネックが 100 Mbit/s の場合は，TCP Sapipa はフロー数が 1 コネクションの時には TCP Reno より帯域利用効率が高いのに対して，HSTCP，STCP と比較して低い．これは図 5.7 において，あるスループットにおけるパケット廃棄率を比較したとき，パケット廃棄率の高い TCP は輻輳状態に達しやすく，その輻輳ウィンドウにおいて帯域を有効利用できるからである．つまり，図 5.10 において，図 5.7 中でスループットが 100 Mbit/s のときにパケット廃棄率が高い TCP は，帯域を有効に利用していることがわかる．またフロー数が増大するにつれて，TCP Sapipa，TCP Reno と HSTCP の差異がなくなってきている．輻輳ウィンドウを最も多く増加させる STCP が効率よく転送している．TCP Sapipa は，フロー数に関係なく，TCP Reno と同等の帯域利用率を実現

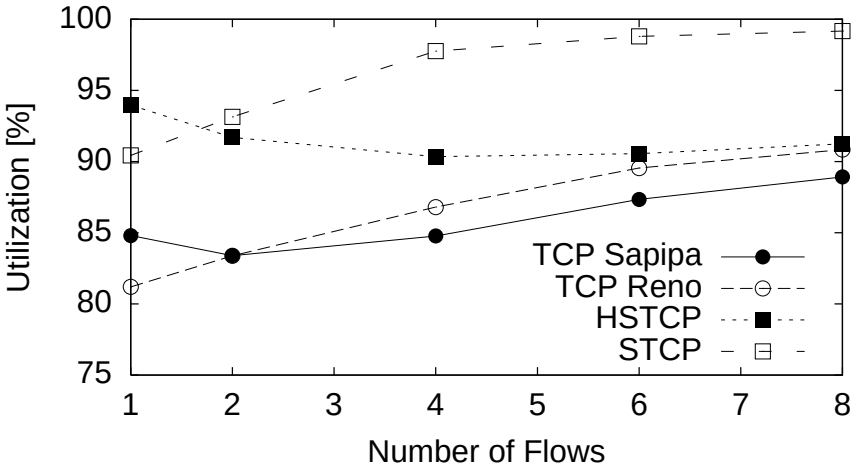

図 5.10: コネクション数における 100 Mbit/s での帯域利用率

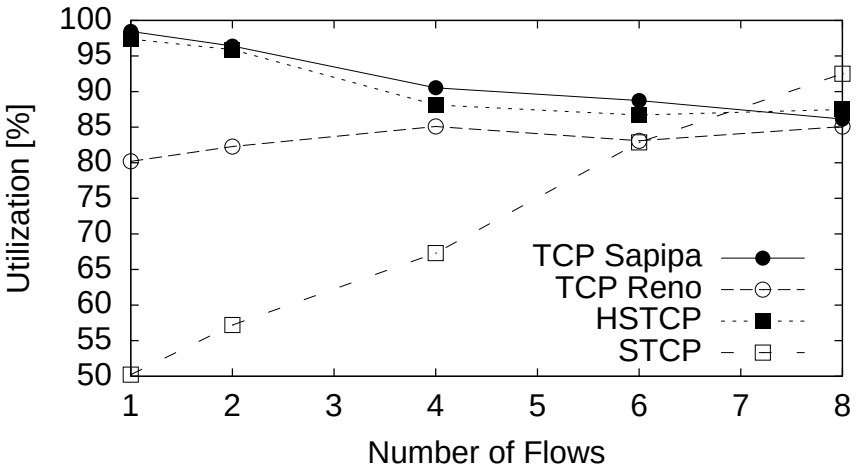

図 5.11: コネクション数における 1 Gbit/s での帯域利用率

表 5.4: ボトルネックの変化に対する帯域利用率の評価におけるシミュレーション条件

コネクション数	1
ボトルネックの帯域	10 Mbit/s〜1 Gbit/s
RTT	100 ms

していることから，100 Mbit/s という帯域の狭い場合においては帯域利用効率ではなく，TCP Reno との親和性を重視していることがわかる．

図 5.11 より，ボトルネックが 1 Gbit/s の時は，TCP Sapipa はフロー数に関係なく HSTCP と同様の帯域利用効率を実現し，TCP Reno よりも最大 15% 帯域を有効利用できている．TCP Sapipa が 1 Gbit/s においては，TCP Reno との親和性ではなく，広帯域転送に適した HSTCP と同程度の帯域利用効率を重視していることがわかる．このことは図 5.7 において，p が小さい値のとき，TCP Sapipa が HSTCP 相当のスループットを示していることからもわかる．一方，STCP は広帯域データ転送では増加量が大きく，廃棄パケット数も増大し廃棄パケットの再送制御に時間がかかるため，性能が悪化してしまう．これはコネクションが少ない場合に顕著にその兆候が見られる．以上より，ボトルネックの帯域が 100 Mbit/s の場合と，1 Gbit/s の場合で，帯域獲得の結果が大きく異なっている．

本評価では，TCP 制御方式の特性を評価するために，8 コネクションまでのコネクションを同時に転送した場合のシミュレーションを行った．実際の環境では，膨大なコネクションが共存しているが，1 つのコネクションに着目した場合，ネットワークの帯域は，他のコネクションが使用している帯域と，どのコネクションも使用していない帯域に分けられる．その空き帯域を効率よく使用する点では，コネクションが 1 本のときの帯域使用効率が高い TCP Sapipa は，膨大なコネクションが共存している実ネットワークにおいても帯域使用効率向上効果があると考えられる．

5.5.2.2 ボトルネック帯域の変化に対する帯域利用効率

つぎに，ボトルネックの帯域を変化させた際の TCP Sapipa の帯域利用効率の評価を行った．シミュレーション条件を表 5.4 にまとめる．

以上の環境の下，TCP Reno，HSTCP，STCP と TCP Sapipa の輻輳制御方式で，それぞれ，ボトルネックの各帯域で転送を行い，帯域利用率を測定した．図 5.12 がその結果を示したグラフである．

図より広帯域転送において TCP Sapipa は効率よく転送が行われている．図 5.7 において輻輳ウィンドウが大きくなるにつれて，TCP Sapipa は HSTCP の廃棄率に近づいているため，実際の転送量も HSTCP に近い値を実現している．

以上のことから，TCP Sapipa は転送量が大きくなるにつれて帯域利用率が向上し，コネクション数が少ないほうがその効果が顕著に現れることがいえる．転送効率は HSTCP,

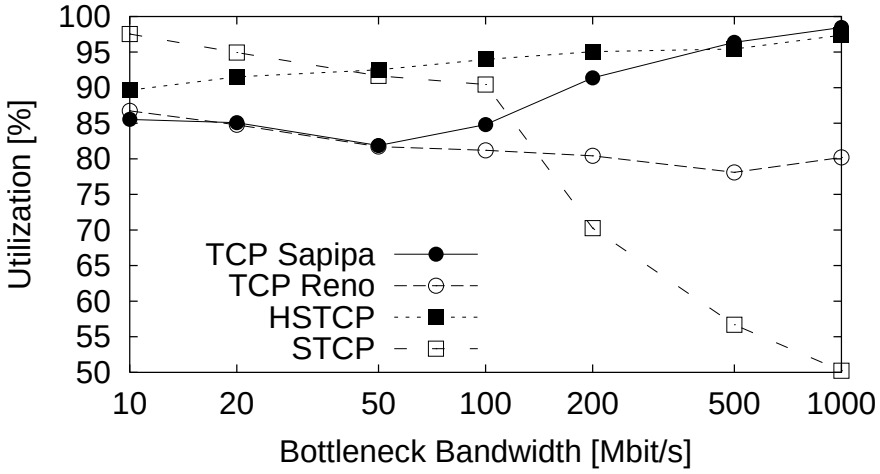

図 5.12: コネクション 1 本における帯域と帯域利用率の関係

表 5.5: TCP Reno との親和性の評価におけるシミュレーション条件

コネクション数	2 (TCP Reno および他 TCP)
ボトルネックの帯域	10 Mbit/s〜1 Gbit/s
RTT	100 ms

STCP ほど常に高くはない場合があるが，広帯域転送を実現できるプロトコルであると考えられる．

5.5.3 TCP Reno との親和性の評価

TCP Sapipa の TCP Reno との親和性を確認するために，HSTCP，STCP と比較して，シミュレーションを行った．

シミュレーション条件を表 5.5 にまとめる．

以上の環境の下，HSTCP，STCP と TCP Sapipa の輻輳制御方式で，それぞれ，ボトルネックの各帯域で転送を行い，TCP Reno との親和性を測定した．

その結果を図 5.13 に示す．ここで，親和性を示す指標として TCP Reno に対してのその TCP コネクションの帯域の比を用いる．親和性の値が 1 のとき，TCP Reno との帯域が等しく，公平な状態と定義する．

図より帯域が小さいときには HSTCP，STCP と TCP Sapipa ともに TCP Reno との帯域がの比が少なく，親和性が高い．一方，帯域が大きくなるにつれて HSTCP，STCP は指標が大きく増大しているのに対して，TCP Sapipa は比が大きくはなっているが，HSTCP，STCP と比べて相対的に比が少なくなっているのが分かる．また，これは，図 5.8 より，計算の結果と一致している．以上のことから，TCP Sapipa は HSTCP，STCP と比較して TCP Reno との親和性が改善されている．

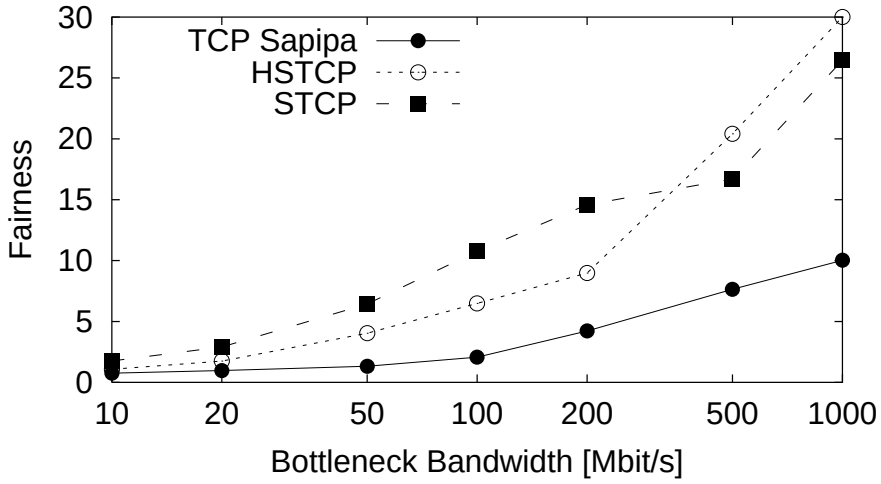

図 5.13: 帯域の変化に対する TCP Reno との親和性

表 5.6: 異なる RTT のコネクション間の公平性の評価におけるシミュレーション条件

コネクション数	8
ボトルネックの帯域	100 Mbit/s および 1 Gbit/s
RTT	40 ms, 70 ms, 100 ms, 130 ms, 160 ms, 190 ms, 220 ms, 250 ms

5.5.4 異なる RTT のコネクション間の公平性の評価

TCP Sapipa の異なる RTT のコネクション間の公平性を確認するために，TCP Reno, HSTCP，STCP と比較して，シミュレーションを行った．

シミュレーション条件を表 5.6 にまとめる．

以上の環境の下，TCP Reno，HSTCP，STCP と TCP Sapipa の輻輳制御方式で転送を行い，帯域利用率を測定した．その結果を，図 5.14，図 5.15 にそれぞれボトルネックの帯域が 100 Mbit/s，1 Gbit/s の時のそのコネクションの帯域を示したグラフを示す．また，表 5.7，表 5.8 にコネクションの帯域，コネクション間の公平性を示した．ここで，コネクション番号 1〜8 の RTT をそれぞれ $40, 70, 100, 130, 160, 190, 220, 250ms$ で転送した．

帯域が 100 Mbit/s の場合も 1 Gbit/s の場合も TCP Reno，HSTCP，STCP では RTT が短いコネクションの帯域が大きく，RTT の長いコネクションの帯域が小さく，コネクション間の帯域が分散し，F が 0.34〜0.53 を示している．一方，TCP Sapipa では帯域に関係なくコネクション間の分散こそあれ，F が 0.85 を示している．また，帯域が 100 Mbit/s の場合も 1 Gbit/s の場合も，TCP Reno，HSTCP，STCP では，最もスループットの高いコネクションと低いコネクションの比が，9 倍以上を示しているのに対して，TCP Sapipa は，最もスループットの高いコネクションと低いコネクションの比が，2 倍以下を示している．これは式 (2.5)，式 (2.9)，式 (2.11) より，TCP Reno，HSTCP，STCP のスループットは RTT に依存する値になっているのに対して，式 (5.28) より TCP Sapipa のスループットは RTT に依存しない値になっているからである．以上のことから，TCP

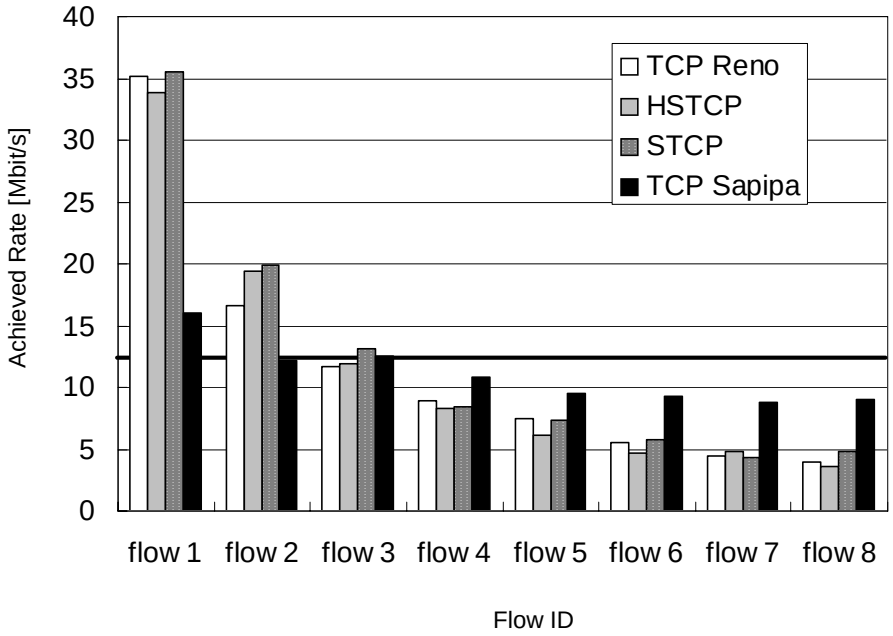

図 5.14: RTT の異なるコネクションにおける 100Mbit/s でのスループット

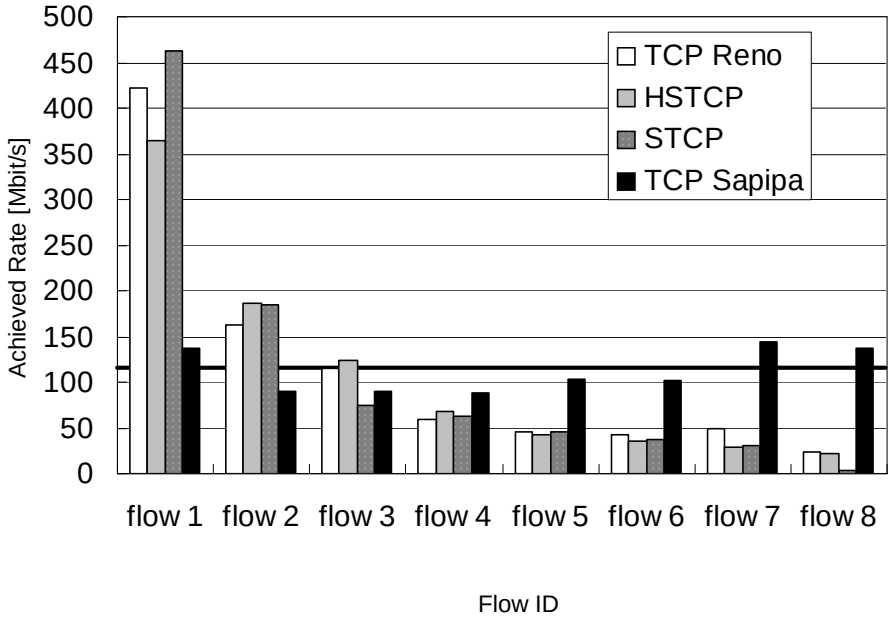

図 5.15: RTT の異なるコネクションにおける 1Gbit/s でのスループット

表 5.7: 100 Mbit/s の帯域におけるコネクションの帯域と公平性 F

flow #	RTT (ms)	公平な帯域	TCP Reno	HSTCP	STCP	TCP Sapipa
1	40	12.50	35.13	33.85	35.49	16.00
2	70	12.50	16.61	19.35	19.84	12.18
3	100	12.50	11.72	11.98	13.16	12.54
4	130	12.50	8.93	8.28	8.42	10.87
5	160	12.50	7.53	6.18	7.30	9.51
6	190	12.50	5.55	4.73	5.73	9.28
7	220	12.50	4.51	4.88	4.35	8.82
8	250	12.50	3.95	3.63	4.80	9.00
total			93.93	92.89	99.09	88.20
F			0.4115	0.4391	0.3447	0.8546

表 5.8: 1 Gbit/s の帯域におけるコネクションの帯域と公平性 F

flow #	RTT(ms)	公平な帯域	TCP Reno	HSTCP	STCP	TCP Sapipa
1	40	125.0	422.4	365.1	462.7	137.3
2	70	125.0	163.5	186.5	184.7	89.7
3	100	125.0	114.9	124.1	74.3	90.3
4	130	125.0	58.5	68.3	63.4	87.9
5	160	125.0	45.1	43.2	45.0	104.0
6	190	125.0	42.0	34.9	36.9	102.2
7	220	125.0	49.9	28.0	30.8	143.5
8	250	125.0	23.9	22.1	3.9	137.6
total			920.2	872.2	901.7	892.5
F			0.5298	0.5239	0.5387	0.8514

Sapipa は RTT 間の公平性が改善されていることが分かる．

図 5.14，図 5.15 から確認できる提案方式におけるコネクション間の帯域の分散は，収束性の問題であると考えられる．提案方式は，広帯域転送で効率よく帯域獲得できる仕組みであるため，広帯域転送を行うにしたがい輻輳ウィンドウの増加量が大きくなる．そのため，複数のコネクションが共存し，それらの帯域が分散しているとき，その帯域の分散が収束することが難しい．コネクションの開始時刻の違い等コネクションの状況の違いにより，帯域獲得が不公平になることがあり，その状況が確認できたと考えられる．

5.6 まとめ

現行の TCP，TCP Reno は TCP Reno の広帯域ネットワークで性能が悪くなる問題，RTT の異なるコネクションが競合するときにスループットが不公平に分配されるという問題が指摘されている．これらの問題に対して，様々な提案がなされてきたが，広帯域ネットワークでの帯域利用効率，TCP Reno との親和性，RTT の異なるコネクションの公平性の 3 つの問題を同時に解決している研究がなされていないため，本章では同時に解決する TCP Sapipa を提案した．TCP Sapipa は輻輳の有無に対して輻輳ウィンドウの増減に変更を行うだけであり，スケーラビリティが高い．輻輳ウィンドウの増加をその輻輳ウィンドウの平方根に反比例させ，減少をその輻輳ウィンドウの平方根に比例させることで，高速データ転送に適したウィンドウ増減を実現している．また，輻輳ウィンドウの増加を RTT に比例させることでコネクション間に存在する RTT の差異による不公平な帯域割当ての問題の改善を行っている．さらに，輻輳ウィンドウの増減の係数を設定することで，TCP Reno との親和性の問題も改善を行っている．

TCP Sapipa のスループットを解析的に求め，また，コンピュータシミュレーションを用いた評価および TCP Reno，HSTCP と STCP との比較の結果，以下のことがわかった．TCP Sapipa は 1 Gbit/s の広帯域転送時，HSTCP と同様の効率でデータ転送を実現し，TCP Reno よりも最大 15%帯域を有効利用できる．TCP Reno と同時に転送した場合，帯域が大きくなるにつれて，TCP Reno との親和性を改善でき，1 Gbit/s の広帯域転送時には HSTCP，STCP と比較して TCP Reno との親和性を 2.5〜3 倍改善している．コネクションごとに RTT の異なる環境で転送を行った場合，公平性を示す指標が TCP Sapipa では 0.85 で，他方式の 0.34〜0.53 と比較して，RTT の値に依存しない帯域分配が行われている．以上より，TCP Sapipa は帯域が狭い場合には，TCP Reno との問題を改善し，広帯域になるにつれて，広帯域データ転送に適し，帯域量に関係なく，コネクション間に存在する RTT の差異による不公平な帯域割当ての問題を改善する有効な輻輳制御機構である．

第6章　結論

本論文は，帯域が広く，伝播遅延が大きい帯域遅延積の大きなネットワークにおいて，ネットワーク内のコネクションに対して高いサービス品質を提供可能な帯域制御技術に関する研究の成果をまとめたものである．

1 章では，本研究の目的として，帯域遅延積の大きなネットワークにおいて，ネットワーク内のコネクションに対して高いサービス品質を実現する帯域制御技術を提案することを示した．本研究の目的を達成する研究の位置付けとして，コネクション間の公平性の課題と帯域使用効率の課題を取り上げ，ネットワークの改善によるアプローチ，エンドホストとネットワークルの連携による改善アプローチ，エンドホストの改善によるアプローチの 3 つのアプローチからの解決を図ることを示した．

2 章では，効率的かつ公平なデータ転送を実現するために必要な帯域制御技術にかかわる課題を明らかにした．具体的には，拡大する帯域，分散する伝播遅延の帯域保証ネットワークにおける，公平性，帯域使用効率の観点から指摘されている課題として，(1) 異なる RTT のコネクション間の公平な帯域割当て，(2) 広帯域ネットワークにおける帯域使用効率の向上の課題について，既存方式の現状について述べた．

3 章では，伝播遅延の異なるコネクションがネットワーク上で共存した場合の帯域割当ての課題に着目し，ネットワークルータによる解決を図った．この課題に対して，AF PHB を利用した Diffserv 帯域保証サービスにおける集約コネクション中での適用に焦点を絞った．

その解決策として，以下の 2 つの提案を行った．1 つ目は，エッジルータへの FR の通知である．集約コネクション内のコネクションが公平とされる帯域として，FR を定義し，契約ホストにおいて契約帯域と集約コネクション数から FR を計算し，ポリシーサーバを介してエッジルータに FR を通知方法を提案した．2 つ目は，エッジルータによる FR に基づいた，公平な帯域割当て量を実現するマーキング方法として，個別のコネクションにマーキングを行う手法，AI マーキングと IA マーキングを提案した．本方式は，ネットワークのエッジノードのみの改善によってコネクション間の公平性を実現する技術であり，現ネットワークへの適用が容易な方式である．また，ネットワークの改善であるために，異種混在環境になることは無いため，使用プロトコル間においての公平性についての問題もない．

コンピュータシミュレーションを用いたその有効性の検証の結果，既存マーキング方式よりも提案方式の IA マーキングのマーキングを用いた場合において，伝播遅延の異なるコネクションに対しての帯域割当て量を公平にすることができた．公平な帯域割当ては，伝播遅延の差が少ない場合に効果が発揮され，伝播遅延の差が大きい場合は，公平性の改善の程度が少なく，目標としていた FR を達成できない場合があった．また，提案方式，AI マーキング は余剰帯域を効率よく利用できた．以上のことから，本課題に対するネットワークにおける解決アプローチとして，提案方式の AI マーキングは公平な帯域保証サービスを実現する一助となると考える．

4 章では，3 章と同様に，RTT の異なるコネクション間で不公平な帯域割当て問題に注

目し，この課題に対して，AF PHB を利用した Diffserv 帯域保証サービスにおける集約コネクション中での適用に焦点を絞った．この課題に対して，3 章と異なるアプローチのネットワークとエンドホストの連携による解決を図り，FR-TCP を提案した．

FR-TCP 輻輳制御機構では 2 点の提案を行った．1 点目として，FR を送信側ホストへ通知する機構についての提案を行った．この通知機構では，契約ホストにおいて FR が計算され，ポリシーサーバを介してエッジルータに FR が通知される．エッジルータにおいて通知された FR をパケットヘッダに記入し，その値を ACK パケットにコピーすることによって送信側へ通知する仕組みとなっている．2 点目として，送信側ホストにおいて通知された FR を考慮した送信帯域の制御を行う輻輳ウィンドウ制御アルゴリズムを提案した．このアルゴリズムでは，通知された FR に対応する値に *ssthresh* を設定し，更に ECN 機構を利用することで，FR 以上の帯域での安定したデータ送信を実現している．以上のように，本提案はネットワークノードとエンドホストが連携することにより，帯域割当ての公平性を向上させる提案である．

更に提案機構について，コンピュータシミュレーションを用いてその有効性を検証した．帯域割当ての公平性についての評価を行った結果，FR-TCP は集約されているコネクションの RTT，またリンク全体の帯域幅に関わらず常に各コネクションに含まれる TCP コネクションが FR 以上の帯域でデータ送信を行い，既存の TCP Reno と比較して帯域割当ての公平性を大きく改善することがわかった．FR-TCP によるネットワークとエンドホストの連携による解決は，3 章で示した IA マーキングのエンドホストによる解決と比較して，より公平性を実現していることがわかった．以上のように， FR-TCP は従来の TCP の場合と比較して，想定したあらゆる場合について集約全体の帯域使用量を抑える事なく，かつ帯域割当ての公平性について大きな改善が得られることがわかった．よって，提案機構は AF PHB を利用する帯域保証サービスを集約コネクションに適用する場合に，集約コネクション内の RTT に関わらずより公平なデータ転送を行うことを可能とし，同一のサービスを利用するコネクションにより均等にサービスを享受させることに関して非常に有効な方法であるといえる．

5 章では，伝播遅延が大きくなる，ネットワークの帯域が拡大するネットワークにおいて，効率的な帯域獲得の課題に焦点を絞り，この課題に対してエンドネットワークにおいて提案を行った．本提案においては，一般的なネットワークでの適用での使用を前提とし，考察を行い，効率的かつ公平な帯域割当てを実現する TCP Sapipa を提案した．本方式は広帯域データ転送に適すること，TCP Reno との親和性との課題を解決すること，コネクション間に存在する RTT の差異による不公平な帯域割当ての課題を解決することを目的としている．

TCP Sapipa は輻輳の有無に対して輻輳ウィンドウの増減に変更を行うことで実現でき，スケーラビリティが高い．輻輳ウィンドウの増加をその輻輳ウィンドウの平方根に反比例させ，減少をその輻輳ウィンドウの平方根に比例させることで，高速データ転送に適したウィンドウ増減を実現している．また，輻輳ウィンドウの増加を伝播遅延に比例させることでコネクション間に存在する RTT の差異による不公平な帯域割当ての課題の解決をし

ている．さらに，輻輳ウィンドウの増減の係数を設定することで，TCP Reno との親和性との課題も解決をしている．

コンピュータシミュレーションによって評価を行い，以下のことがわかった．TCP Sapipa は 1 Gbit/s の広帯域転送時，HSTCP と同様の効率でデータ転送を実現し，TCP Reno よりも最大 15%帯域を有効利用できる．TCP Reno と同時に転送した場合，帯域が大きくなるにつれて，TCP Reno との親和性を改善でき，1 Gbit/s の広帯域転送時には HSTCP, STCP と比較して TCP Reno との親和性を 2.5〜3 倍改善できる．コネクションごとに RTT の異なる環境で転送を行った場合，公平性を示す指標 F が TCP Sapipa では 0.85 で，他方式の 0.34〜0.53 と比較して，RTT の値に依存しない帯域分配が行われている．一方で，RTT に依存しないがコネクションの帯域獲得に分散が生じており，TCP Sapipa の収束性に問題があることも確認した．

以上より，TCP Sapipa は帯域が狭い場合には，TCP Reno との課題を解決し，広帯域になるにつれて，広帯域データ転送に適し，帯域量に関係なく，コネクション間に存在する RTT の差異による不公平な帯域割当ての課題を解決する有効な輻輳制御機構である．

本研究では，公平性の課題と帯域使用効率の課題に取り組んでいる．

公平性の改善は，3，4，5 章による提案より，ネットワークにおける解決，ネットワークとエンドホストの連携による解決，エンドホストにおける解決の 3 手法で実現できた．ネットワークとエンドホストの連携による解決では，達成すべき帯域の FR を常に実現できることを示し，性能の高さが際立っていた．この性能の高さは，ネットワークとエンドホストの両方から改善を行ってるからであると考えられる．また，ネットワークにおける解決においても，状況により目標としていた帯域に達成できていない場合もあったが，公平性の改善は見られた．公平性とはコネクション間の性能のため，この課題を解決するには，複数のコネクションが観測できるネットワークにおいて制御をすることが適していると考えられる．エンドホストの解決では，改善はみられたものの他のコネクションの存在を認識することなく公平性を実現することが難しいことが確認できた．

帯域使用効率に関しては，また，5 章および関連研究による提案より，ネットワークとエンドホストの連携による解決，エンドホストにおける解決の 2 手法で改善を図ることを示した．コネクション単体の性能のため，ネットワークにおける解決では困難であると考えられる．性能に関しては，ネットワークとエンドホストの連携による関連研究の解決で最も改善が実現できていたが，エンドホストの解決においても，転送効率の効果が十分にあることが示せた．コネクション単体の性能である効率性に関しても，ネットワークとエンドホストの両方から改善を行うことで，性能の高さを確認できた．

本研究では，公平性の課題と帯域使用効率の課題に対して 3 つの解決アプローチを提案し，それぞれのアプローチがそれぞれの特性を持っていた．ネットワークにおける解決で，公平性等の複数のコネクション間に関係するサービス品質の改善が効果的であることを示し，エンドホストにおける解決で，転送効率等のコネクション単体におけるサービス品質において改善が効果的であることを示した．また，ネットワークとエンドホストの連携による解決で，複数のコネクション間に関係するサービス品質，コネクション単体にお

けるサービス品質において，高い性能向上が図れることを示した．

ネットワークとエンドホストが連携することで，サービス品質の向上効果は，飛躍的に改善されるが，スケーラビリティの点で，現ネットワークへの適用に問題を抱えている．今後，ネットワークが遅延が大きくなり，帯域が拡大していくことを考慮し，そのネットワークへ適用する場合，エンドホストでコネクション単体の転送性能の向上を図り，ネットワークにおいてコネクション間の遅延，プロトコル等の公平性および差別化を実現することが適していると考えられる．つまり，エンドホスト，ネットワークのそれぞれで，コネクション単体の性能と，コネクション間の性能の向上を図り，それらの方式を組み合わせることで，実用的かつ効果的な改善が見込まれる．

今後の課題として，より効率的かつ公平なデータ転送を実現するために以下の点が 2 点が課題となるであろう．

1 点目は，Diffserv 網における未契約の帯域の公平な帯域割当てである．3 章で提案したネットワークルータによる解決も，4 章で提案したネットワークとエンドホストの連携による解決も，契約帯域内において，集約された各コネクションを公平に割当てることを目的している．しかし，両提案方式とも集約フロー内の完全な公平性を実現することができていない．特に他のコネクションが使用していない未使用帯域の割当てに関して，帯域を獲得したいコネクション間で公平に割当ててることができていなかった．そのため，契約帯域だけでなく未契約の余剰帯域の公平性を向上させる何らかの提案が必要であると考えられる．

2 点目は，TCP 輻輳制御機構における帯域の収束性である．5 章で提案した TCP 輻輳制御機構は，広帯域転送で効率よく帯域獲得できる仕組みであるため，広帯域転送を行うにしたがい輻輳ウィンドウの増加量が大きくなる．そのため，複数のコネクションが共存し，それらの帯域が分散しているとき，その帯域の分散が収束することが難しい．特に，コネクションの開始時刻の違いにより，帯域獲得が不公平になることがある．この問題を解決し，同一プロトコルの TCP コネクションは，開始時刻が異なっていても，コネクションの獲得帯域が短時間で収束する提案が必要となる．

謝 辞

本論文をまとめるにあたって，終始格別なご指導を賜りました，慶應義塾大学 情報工学科 岡田 謙一 教授に謹んで深い感謝の意を表します．また，有益なご討論，ご指導を賜りました慶應義塾大学 情報工学科 山中 直明 教授，慶應義塾大学 情報工学科 寺岡 文男 教授，慶應義塾大学 情報工学科 重野 寛 助教授，東京工科大学 コンピュータサイエンス学部 学部長 松下 温 教授 (元慶應義塾大学 情報工学科 教授) に心から感謝申し上げます．

本論文の執筆の機会を与えて頂いた NTT アドバンステクノロジ (株) 国際事業推進本部 前田 洋一 部長 (元 NTT アクセスサービスシステム研究所 部長)，NTT アクセスサービスシステム研究所 中島 隆 部長，NTT アクセスサービスシステム研究所 梶山 義夫 主幹研究員に心から感謝いたします．

本研究は，慶応大学理工学部 情報工学科 岡田研究室で行われました．研究を共に行い貴重なご意見を頂いた先輩である，坂井 達彦 氏，野田 陽子 氏，また，議論を行い共に励ましあった，上田 真太郎 氏，福原 政彦 氏，廣瀬 史明 氏，それぞれの方に深く感謝をいたします．

最後に，私を常に暖かく見守り支えて下さいました私の両親，そして私の研究活動を理解し，精神的に支えてくださいました私の妻 真理に心から感謝します．

参考文献

[1] 平成 18 年版　情報通信白書.
URL: http://www.johotsusintokei.soumu.go.jp/whitepaper/ja/h18/.

[2] 戸田巌. ネットワーク QoS 技術. オーム社, 2001.

[3] 井上 一朗三宅 功. 通信インフラとしての IP 技術. 信学会誌特集号, Vol. 82, No. 10,12, Apr. 2000.

[4] F. Baker, J. Krawczyk, and A. Sastry. Integrated Services Management Information Base using SMIv2. RFC 2213 (Proposed Standard), September 1997.

[5] F. Baker, J. Krawczyk, and A. Sastry. Integrated Services Management Information Base Guaranteed Service Extensions using SMIv2. RFC 2214 (Proposed Standard), September 1997.

[6] K. Nichols, S. Blake, F. Baker, and D. Black. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. RFC 2474 (Proposed Standard), December 1998.

[7] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Service. RFC 2475 (Informational), December 1998.

[8] D. Grossman. New Terminology and Clarifications for Diffserv. RFC 3260 (Informational), April 2002.

[9] B. Davie, A. Charny, J.C.R. Bennet, K. Benson, J.Y. Le Boudec, W. Courtney, S. Davari, V. Firoiu, and D. Stiliadis. An Expedited Forwarding PHB (Per-Hop Behavior). RFC 3246 (Proposed Standard), March 2002.

[10] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. Assured Forwarding PHB Group. RFC 2597 (Proposed Standard), June 1999.

[11] J. Postel. Transmission Control Protocol. RFC 793 (Standard), September 1981.

[12] K. Ramakrishnan, S. Floyd, and D. Black. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3168 (Proposed Standard), September 2001.

[13] Van Jacobson. Congestion Avoidance and Control. In *ACM SIGCOMM*, pp. 314–329, August 1988.

[14] K. K. Ramakrishnan, D. Chiu, and R. Jain. Congestion Avoidance in Computer Networks with a Connectionless Network Layer, Part IV: A Selective Binary Feedback Scheme for General Topologies. Technical report, 1987.

[15] R. Jain and K. K. Ramakrishnan. Congestion Avoidance in Computer Networks with a Connectionless Network Layer: Concepts. *Proceedings of the Computer Networking Symposium; IEEE; Washington, DC*, pp. 134–143, 1988.

[16] M. Allman, V. Paxson, and W. Stevens. TCP Congestion Control. RFC 2581 (Proposed Standard), April 1999.

[17] Wenjia Fang. *Differentiated Services: Architecture, Mechanisms and an Evaluation.* PhD thesis, Princeton University, 2000.

[18] K. Nichols and B. Carpenter. Definition of Differentiated Services Per Domain Behaviors and Rules for their Specification. RFC 3086 (Informational), April 2001.

[19] Z. Wang and J. Crowcroft. Eliminating Periodic Packet Losses in 4.3-Tahoe BSD TCP Congestion Control. In *ACM Computer Communication Review*, pp. 9–16, 1992.

[20] M. Perloff and K. Reiss. Improvements to TCP performance in high-speed ATM networks. In *Communications of the ACM*, pp. 9–16, 1992.

[21] S. Floyd, T. Henderson, and A. Gurtov. The NewReno Modification to TCP's Fast Recovery Algorithm. RFC 3782 (Proposed Standard), April 2004.

[22] Jitedra Padhye, Victor Firoiu, Don Towsley, and Jim Krusoe. Modeling TCP Throughput: A Simple Model and its Empirical Validation. In *ACM SIGCOMM '98*, pp. 303–314, 1998.

[23] S. Floyd. Limited Slow-Start for TCP with Large Congestion Windows. RFC 3742 (Experimental), March 2004.

[24] Tom Kelly. Scalable TCP: Improving Performance in HighSpeed Wide Area Networks. In *ACM SIGCOMM Computer Communication Review*, pp. 9–16, 1992.

[25] Fang, W. and Peterson L. TCP mechanisms for Diff-Serv Architecture. In *Princeton University, CS Dept., Technical Report TR-605-99*, September 1999.

[26] T. Hamann and J. Walrand. A New Fair Window for ECN Capable TCP (New-ECN). In *Tech. Rep. Memorandum No. UCB/ERL M99/35, Electronics Research Laboratory*, College of Engineering, UC Berkeley, 1999.

[27] Wei Lin, Rong Zheng, and Jennifer C. Hou. How to Make Assured Service More Assured. In *IEEE ICNP*, pp. 182–191, 1999.

[28] Ikjun Yeom and A. L. Narasimha Reddy. Realizing Throughput Guarantees in a Differentiated Services Network. In *ICMCS*, Vol. 2, pp. 372–376, 1999.

[29] Biswajit Nandy, Nabil Seddigh, Peter Pieda, and Jeremy Ethridge. Intelligent Traffic Conditioners for Assured Forwarding Based Differentiated Services Networks. In *NETWORKING*, pp. 540–554, 2000.

[30] David D. Clark and Wenjia Fang. Explicit Allocation of Best Effort Packet Delivery Service. *IEEE/ACM Transactions on Networking*, Vol. 6, No. 4, pp. 362–373, 1998.

[31] D. Chiu and R. Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. In *Computer Networks and ISDN Systems*, Vol. 17, pp. 1–14, 1989.

[32] 秦野智也, 坂井達彦, 野田陽子, 重野寛, 松下温. Diffserv AF 集約フロー内の公平な帯域割当を意識したエッジルータの改善. 情報処理学会第 64 回全国大会, pp. 627–628, Mar. 2002.

[33] 秦野智也, 野田陽子, 坂井達彦, 重野寛, 岡田謙一, 松下温. 公平な帯域分配を実現するエッジルータの改善. 情報処理学会シンポジウムシリーズ, Vol. 2002, No.15, pp. 195–200, Oct. 2002.

[34] 秦野智也, 野田陽子, 重野寛, 岡田謙一, 松下温. Diffserv 網における公平な帯域分配のためのマーキング手法. 情報処理学会論文誌, Vol. 44, No. 12, pp. 3152–3160, Dec. 2003.

[35] K. Nichols, V. Jacobson, and L. Zhang. A Two-bit Differentiated Services Architecture for the Internet. RFC 2638 (Informational), July 1999.

[36] J. Walker and A. Kulkarni. Common Open Policy Service (COPS) Over Transport Layer Security (TLS). RFC 4261 (Proposed Standard), December 2005.

[37] Cisco Systems Inc. URL: http://www.cisco.com/.

[38] Spirent Communications. URL: http://www.spirentcom.com/.

[39] The Network Simulator - ns-2. URL: http://www.isi.edu/nsnam/ns/.

[40] V. Jacobson, R. Braden, and D. Borman. TCP Extensions for High Performance. RFC 1323 (Proposed Standard), May 1992.

[41] R. Jain, D.M. Chiu, and H. Hawe. A quantitative measure of fairness and discrimination for resource allocation in shared systems. Tech.rep., Digital Equipment Corporation, 1984.

[42] 野田陽子, 秦野智也, 坂井達彦, 重野寛, 岡田謙一, 松下温. Diffserv ネットワークにおける帯域割当の公平性を向上させる TCP 輻輳制御機構. 情報ネットワーク研究会, 信学技報, Vol. 102, No. 214, pp. 25–30, Jul. 2002.

[43] T. Hatano, H. Shigeno, K. Okada, and Y. Matsushita. Fair Rate TCP for Fair Share of Bandwidth over Diffserv Networks. In *PACRIM '03*, Vol. 1, pp. 401–404, Aug. 2003.

[44] Tomoya Hatano, Hiroshi Shigeno, Ken ichi Okada, and Yutaka Matsushita. Fair Bandwidth Allocation in Diffserv Network. 情報処理学会論文誌, Vol. 45, No. 12, pp. 2773–2781, Dec. 2004.

[45] ATM Forum TM. Traffic Management Specification Version 4.0. Technical report, The ATM Forum, Apr. 1996.

[46] S.PISANPATTANAKUL and M.YAMAMOTO and H.IKEDA. ABR Traffic Control Scheme for Bursty Traffic Considering the Number of Hops. *IEICE Trans. on Commun.*, Vol. E81-B, No. 11, pp. 2056–2063, Nov. 1998.

[47] W. Fang, N. Seddigh, and B. Nandy. A Time Sliding Window Three Colour Marker (TSWTCM). RFC 2859 (Experimental), June 2000.

[48] 秦野智也, 福原政彦, 重野寛, 岡田謙一. TCP Reno との公平性を意識した高速転送のための TCP 輻輳制御機構. 情報ネットワーク研究会, 信学技報, Vol. 103, No. 198, pp. 19–24, Jul. 2003.

[49] Tomoya HATANO, Masahiko FUKUHARA, Hiroshi SHIGENO, and Ken ichi OKADA. TCP-friendly SQRT TCP for High Speed Networks. In *APSITT*, pp. 455–460, Nov. 2003.

[50] Masahiko Fukuhara, Fumiaki Hirose, Tomoya Hatano, Hiroshi Shigeno, and Ken ichi Okada. SRF TCP: A TCP-friendly and Fair Congestion Control Method for High Speed Networks. In *OPODIS*, pp. 105–114, Dec. 2004.

[51] 秦野智也, 廣瀬史明, 福原政彦, 重野寛, 岡田謙一. TCP Reno との公平性を意識した広帯域データ転送のための TCP 輻輳制御機構. 電子情報通信学会誌, Vol. J88-B, No. 11, pp. 2263–2274, Nov. 2005.

[52] Tomoya Hatano, Hiroshi Shigeno, and Ken ichi Okada. TCP-friendly Congestion Control for HighSpeed Network. In *IEEE SAINT*, Jan. 2007.

[53] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. TCP Vegas: New Techniques for Congestion Detection and Avoidance. In *ACM SIGCOMM*, pp. 24–35, 1994.

[54] J. S. Ahn, P. B. Danzig, Z. Liu, and L. Yan. Evaluation with TCP Vegas: Emulation and experiment. In *ACM SIGCOMM*, pp. 185–195, 1995.

[55] Lawrence S. Brakmo and Larry L. Peterson. TCP Vegas: End to End Congestion Avoidance on a Global Internet. *IEEE Journal on Selected Areas in Communications*, Vol. 13, No. 8, pp. 1465–1480, 1995.

[56] David X. Wei Cheng Jin and Steven H. Low. FAST TCP for High-speed Long-distance Networks. Internet draft draft-jwl-tcp-fast-01, IETF, 2003.

[57] M. Handley, S. Floyd, J. Padhye, and J. Widmer. TCP Friendly Rate Control (TFRC): Protocol Specification. RFC 3448 (Proposed Standard), January 2003.

[58] Deepak Bansal and Hari Balakrishnan. Binomial Congestion Control Algorithms. In *IEEE INFOCOM*, pp. 631–640, 2001.

[59] Y. Yang and S. Lam. General AIMD Congestion Control, 2000.

[60] Sally Floyd, Mark Handley, Jitendra Padhye, and Jorg Widmer. Equation-based congestion control for unicast applications. In *ACM SIGCOMM*, pp. 43–56, Stockholm, Sweden, August 2000.

[61] HighSpeed TCP Web Page. In *URL:http://www.icir.org/floyd/hstcp.html*.

[62] S. Floyd. HighSpeed TCP for Large Congestion Windows. RFC 3649 (Experimental), December 2003.

[63] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP Selective Acknowledgment Options. RFC 2018 (Proposed Standard), October 1996.

[64] Sally Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, Vol. 1, No. 4, pp. 397–413, 1993.

論文目録

主論文に関連する原著論文

1. 秦野智也, 野田陽子, 重野寛, 岡田謙一, 松下温. Diffserv 網における公平な帯域分配のためのマーキング手法. 情報処理学会論文誌, Vol. 44, No. 12, pp. 3152-3160, Dec. 2003.
2. Tomoya Hatano, Hiroshi Shigeno, Ken-ichi Okada, and Yutaka Matsushita. Fair Bandwidth Allocation in Diffserv Network. 情報処理学会論文誌, Vol. 45, No. 12, pp. 2773-2781, Dec. 2004.
3. 秦野智也, 廣瀬史明, 福原政彦, 重野寛, 岡田謙一. TCP Reno との公平性を意識した広帯域データ転送のための TCP 輻輳制御機構. 電子情報通信学会誌, Vol. J88-B, No. 11, pp. 2263-2274, Nov. 2005.

国際会議発表

1. T. Hatano, H. Shigeno, K. Okada, and Y. Matsushita. Fair Rate TCP for Fair Share of Bandwidth over Diffserv Networks. In *PACRIM '03*, Vol. 1, pp. 401-404, Aug. 2003.
2. Tomoya HATANO, Masahiko FUKUHARA, Hiroshi SHIGENO, and Ken-ichi OKADA. TCP-friendly SQRT TCP for High Speed Networks. In *APSITT*, pp. 455-460, Nov. 2003.
3. Masahiko Fukuhara, Fumiaki Hirose, Tomoya Hatano, Hiroshi Shigeno, and Ken-ichi Okada. SRF TCP: A TCP-friendly and Fair Congestion Control Method for High Speed Networks. In *OPODIS*, pp. 105-114, Dec. 2004.
4. Fumiaki Hirose, Masahiko Fukuhara, Tomoya Hatano, Hiroshi Shigeno, and Ken-ichi Okada. A Two-Level ECN Marking for Fair Bandwidth Allocation between HSTCP and TCP Reno. In *IEEE ICDCS Workshops (MNSA)*, pp. 763-769, Jun. 2005.
5. Tomoya Hatano, Hiroshi Shigeno, and Ken-ichi Okada. TCP-friendly Congestion Control for HighSpeed Network. In *IEEE SAINT*, Jan. 2007.

研究会発表

1. 野田陽子, 秦野智也, 坂井達彦, 重野寛, 岡田謙一, 松下温. Diffserv ネットワークにおける帯域割当の公平性を向上させる TCP 輻輳制御機構. 情報ネットワーク研究会, 信学技報, Vol. 102, No. 214, pp. 25-30, Jul. 2002.
2. 秦野智也, 野田陽子, 坂井達彦, 重野寛, 岡田謙一, 松下温. 公平な帯域分配を実現するエッジルータの改善. 情報処理学会シンポジウムシリーズ, Vol. 2002, pp. 195-200, Oct. 2002.
3. 福原政彦, 秦野智也, 野田陽子, 重野寛, 岡田謙一. Diffserv ネットワークにおける ECN を用いた受信者主導の輻輳制御方式. 第 113 回マルチメディア通信と分散処理研究会, 2003-DPS-113, pp. 51-56, Jun. 2003.
4. 秦野智也, 福原政彦, 重野寛, 岡田謙一. TCP Reno との公平性を意識した高速転送のための TCP 輻輳制御機構. 情報ネットワーク研究会, 信学技報, Vol. 103, No. 198, pp. 19-24, Jul. 2003.
5. 廣瀬史明, 福原政彦, 秦野智也, 重野寛, 岡田謙一. 2 種類の ECN マーキングによる TCP Reno と HS TCP の公平な転送の実現. 情報ネットワーク研究会, 信学技報, Vol. 104, No. 73, pp. 55-60, May 2004.

口頭発表

1. 秦野智也, 坂井達彦, 野田陽子, 重野寛, 松下温. Diffserv AF 集約フロー内の公平な帯域割当を意識したエッジルータの改善. 情報処理学会第 64 回全国大会, pp. 627-628, Mar. 2002.
2. 福原政彦, 秦野智也, 野田陽子, 重野寛, 岡田謙一. Diffserv AF PHB における受信者主導の輻輳制御機構. 情報処理学会第 65 回全国大会, Vol. 3, pp. 481-482, Mar. 2003.

受賞

1. 第 10 回 マルチメディア通信と分散処理 (DPS) ワークショップ. Best Paper Award. 公平な帯域分配を実現するエッジルータの改善. Oct. 2002.