様式博３

博士学位論文

題　　目

無線 LAN における次世代トランスポート層プロトコル
Compound TCP に関する研究

担当指導教員名　登尾 啓史　㊞

申請年月日　2014 年 2 月 3 日

申請者専攻名　コンピュータサイエンス専攻

学生番号　DT11A001

氏名　織田 弘樹　㊞

大阪電気通信大学大学院

目次

図目次

表目次

1　はじめに

現在のインターネットの前身であり，パケット通信による世界初のネットワークである ARPANET は，1969 年に ARPA (Advanced Research Project Agency) によって構築された．構築された当初の ARPANET は，カリフォルニア大学ロサンゼルス校，カリフォルニア大学サンタ・バーバラ校，ユタ大学の 3 つの大学と，スタンフォード研究所からなる 4 ノードによって構成されるネットワークであった．ARPANET 構築時点では，現在のインターネットにおいて広く用いられている TCP/IP はまだ提案されておらず，NCP (Network Control Protocol) と呼ばれる通信プロトコルが用いられていた．

その後，ARPANET は，米国外の大学や研究機関へと接続されるなど大規模化し，民間への解放されはじめる．1983 年には，ARPANET の軍事部分が MILNET (Military Network) として分離し，ARPANET は研究用ネットワークに移行する．ARPANET が研究用ネットワークへと移行すると同時に，ARPANET で用いられる通信プロトコルは NCP から TCP/IP に移行される．また，1986 年には，NSF (National Science Foundation) によって，全米 6 カ所のスーパーコンピュータを 54 [Kbit/s] の専用線で接続した．TCP/IP を用いるネットワークである NFSNet の運用が開始される．しかし，この NSFNet は，AUP (Acceptable Use Policy) と呼ばれる，個人の営利目的に使用することができないなど利用制限が定められていた．しかし，この利用制限は，1991 年に連邦ネットワーク協議会が，ネットワークに接続される端末の制限を撤廃する．これにより，誰でもコンピュータと通信回線があればネットワークに接続することが可能となった．また同年に NFSNet に ARPANET が吸収される．その後，1995 年に NSF が NSFNet の運営を民間企業である MCI などに移管され，民間のインターネット接続業者が登場するなど，インターネットは民間主導型のネットワークとなった．

このような経緯を経て現在に至るインターネットは，現在，全世界で 24 億人が利用していると推測されており，毎年インターネットを利用する人は増加し続けている [1]．今後もインターネットを利用する人は，さらに増加し続けると考えられる．また，現在のインターネットにおいて，トランスポート層プロトコルに TCP を用いた通信は全トラヒックの 80% を占める [2]．TCP を用いた通信は，今後もインターネットにおいて，広く行われていくことが考えられる．

TCP が提案された当時，狭帯域な回線において少数の限られたノード間で通信が行われていた．そのため，現在のインターネットのように，非常に多くのノードが存在すること，また，広帯域な回線において多種多様なアプリケーションが，TCP を用いて通信することは想定されていなかった．そのため，様々な問題が発生することが明らかになっている．それに対して，これまでに数多くの TCP の改善手法が提案されてきた [3–12]．

例えば，[3–6] では，TCP の性能を解析した研究が行われている．現在のインターネットにおいて，トランスポート層プロトコルに TCP を用いた通信はトラヒックの大部分を占める．しかし，その性能は明らかにされていなかった．そこでこれらの研究では，数学的に TCP の性能を解析している．[3, 4] では，TCP Reno の簡単なモデル化を行い，パケット棄却率が一定であるネットワークにおいて，TCP の平均ウィンドウサイズやスループットを導出している．また，[5] では，TCP の輻輳制御とネットワークをあわせたフィードバックシステムとしてモデル化し，TCP の過渡特性を解析にしている．解析の結果，TCP の輻輳回避フェーズにおける過渡特性は，ラウンドトリップ時間に依存し，バックグラウンドトラヒックやボトルネックルータの影響は少ないことを示している．さらに，[6] では，既存の流体近似法および待ち行列理論を組み合わせた解析手法を応用することにより，TCP の定常特性および過渡特性を解析している．その結果，ラウンドトリップ時間が TCP の安定性や過渡特性に影響を与えることや，コネクション数やバックグラウンドトラヒックの増加に伴って，ネットワークが安定することを示している．

また, [7,8] では，TCP の輻輳制御を改良する研究が行われている．[7] では．FEC (Forward Error Correction) 技術を用いて要求された帯域を確保する手法を提案している．TCP は，パケット棄却が発生した場合，確認応答なしにネットワークに送出することができるパケット数を半減させる．そのため，動画像ストリーミングのように常に一定のスループットを得る必要がある通信に，TCP は不向きである．[7] では，TCP 層の下部に FEC 機構を設け，パケット棄却が発生した場合でもパケット棄却を隠蔽することにより送信レートの低下を回避する．また，[8] では，インライン計測に基づく TCP の輻輳制御に関する研究が行われている．TCP の輻輳制御は，パケット棄却が発生するまで輻輳ウィンドウを増加させる単純なものであるため，効率的ではない．そこで利用可能帯域を計測し，それに基づいて輻輳ウィンドウを増減させることで，TCP と比較して効率の良い輻輳制御を行う．

さらに TCP は，パケット棄却を検出しない限り 1 ラウンドトリップ時間ごとに，1 パケットずつ確認応答なしに送信するパケット数を増加させる．そのため，高遅延なネットワークにおいて，高いスループットを得るためには時間を要する．そこで，[9–12] では，高遅延なネットワークにおける研究が行われている．[9,10] では，衛星ネットワークにおける TCP の改善方式を提案している．衛星回線は，山間部や離島といった通信回線の敷設が困難である地域における，ネットワークに接続する手段の 1 つとして期待されている．しかし，衛星回線は，ラウンドトリップ時間が大きく，パケット棄却率が高いという特性を持つ．[9,10] では，このような特性を持つ衛星回線において，スループットを向上させる手法を提案している．また，[11,12] では，TCP プロキシと呼ぶ，コネクションを分割させることで，高遅延なネットワークにおいて高いスループットを得る手法を提案している．通常 TCP は，エンドホスト間で単一のコネクションによってデータ転送を行う．TCP プロキシは，エンドホスト間の TCP コネクションをネットワーク中のノードにおいて分割し，複数の TCP コネクションによってデータ転送を行う．TCP コネクションを分割することで，フィードバックループを小さくできるため，スループットを向上させることができる．

また，広帯域なネットワークにおいて TCP を用いた場合，帯域を最大限活用できないことが明らかになっている．これは，TCP の輻輳制御は，輻輳ウィンドウと呼ばれる確認応答なしに送出することができるパケット数を増減させることで，データ転送速度を調節している．しかし，TCP の輻輳制御は，輻輳ウィンドウの増加幅が小さく，パケット棄却検出時における輻輳ウィンドウの減少幅が小さい．したがって，広帯域なネットワークにおいて，帯域を有効に活用することが困難である．ネットワークの広帯域化は今後も進み，将来的には帯域が 10 [Gbit/s] であるネットワークが普及すると考えられる．そのため，広帯域なネットワークにおける TCP の研究は，今後重要になると考えられる．

広帯域なネットワークにおける TCP の研究は，これまでに数多く行われている [13–22]．既存の改良手法は，損失ベースの輻輳制御，遅延ベースの輻輳制御，ハイブリッド型の輻輳制御の 3 種類に分類される．まず，HighSpeed TCP をはじめとする，損失ベースの輻輳制御 [13–16] は，TCP と同様にパケット棄却を輻輳の指標とする．損失ベースの輻輳制御は，TCP と比較して輻輳ウィンドウの増加速度を大きくし，輻輳を検出した場合の輻輳ウィンドウの減少幅を小さくすることで広帯域なネットワークにおいて，帯域を有効に活用する．しかし，TCP と競合した場合，TCP の帯域を奪う問題がある．また，FAST TCP をはじめとする遅延ベースの輻輳制御 [17–19] は，ラウンドトリップ時間の変化を輻輳の指標とした輻輳制御を行う．遅延ベースの輻輳制御はラウンドトリップ時間からネットワークの輻輳状態を推測し，空き帯域があると判断した場合は，急速に輻輳ウィンドウを増加させることで，広帯域なネットワークにおいて，帯域を有効に活用する．しかし，遅延ベースの輻輳制御が損失ベースの輻輳制御と競合した場合，帯域が不当に奪われる問題がある．最後に，Compound TCP をはじめとするハイブリッド型の輻輳制御手法 [20–22] は，損失ベースの輻輳制御と遅延ベースの輻輳制御を組み合わせた輻輳制御手法である．損失ベースの輻輳制御と遅延ベースの輻輳制御を

組み合わせることによって，それぞれの輻輳制御手法が持つ問題を解決している．ハイブリッド型の輻輳制御は，ネットワークに空き帯域があると考えられる場合には，輻輳ウィンドウを急速に増加させ，帯域を使い切る．また，TCP と競合した場合には，競合する TCP の帯域を奪わない TCP-Friendly な制御を行う．ハイブリッド型の輻輳制御手法の中でも，Compound TCP は，Windows Vista Service Pack 1 以降に搭載されている．そのため，Compound TCP は，今後広く用いられることが考えられる．

また，近年無線 LAN の普及が進んでいる．無線 LAN は，有線 LAN のようにケーブルが存在しないため，設置の自由度が高く，また，利用端末も自由に移動することができる利点をもつ．無線 LAN 対応のモバイル端末出荷台数は，2009 年度は 1161 万台であったのに対し，2013 年度は 3415 万台と急速に増加している [23]．今後も無線 LAN は，普及が進んでいくと考えられる．

これらの無線 LAN 機器の大部分は IEEE (Instiute of Electrical and Electronics Engineers) 802 委員会のワーキンググループ 11 が策定した規格に準拠している．1997 年，IEEE 802.11 ワーキンググループは，最初の標準規格である 802.11 を策定した．IEEE 802.11 は，2.4 [GHz] 帯の周波数帯を用いる規格である．IEEE 802.11 には伝送速度 1 [Mbit/s] である規定と，伝送速度が 2 [Mbit/s] である規定の 2 種類がある．その後，IEEE 802.11 ワーキンググループは，IEEE 802.11 の伝送速度を高速化した IEEE 802.11b を 1999 年に策定した．IEEE 802.11b は，CCK (Complementary Code Keying) と呼ばれる方式を採用した 2.4 [GHz] 帯の周波数帯を用いる伝送速度 11 [Mbit/s] の規格である．IEEE 802.11b の策定と同時に，OFDM (Orthogonal Frequency Division Multiplexing) 方式 を採用した 5 [GHz] の周波数帯域を用いる，最大伝送速度 54 [Mbit/s] の規格である IEEE 802.11a が策定された．IEEE 802.11b と IEEE 802.11a 間には互換性はない．その後，IEEE 802.11b と互換性があり，最大伝送速度 54 [Mbit/s] の規格である IEEE 802.11g や，最大伝送速度 600 [Mbit/s] の規格である IEEE 802.11n が策定されている．このように無線 LAN において広帯域化が進んでおり，今後もさらに広帯域化していくことが考えられる．

有線 LAN では，CSMA/CD (Carrier Sense Multiple Access/Collision Detection) と呼ばれるアクセス制御が行われている．CSMA/CD は，信号を監視し，衝突による信号の変化によって衝突を検出する．しかし，無線 LAN においては送信信号がノイズの影響により大きく変化するため，CSMA/CD では信号の衝突を検出することができない．そこで，IEEE 802.11 では，パケットの衝突を回避するために CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) 方式が用いられている．CSMA/CA は，各端末がパケットを送信する際，他の無線端末が通信を行っていないか確認し，通信を行っていない場合は通信を開始する．また，他の無線端末の通信を検出した場合は，通信が終了するまで待機し，さらにランダムな時間待機した後パケットを送信することで，他の無線端末との衝突を回避する．

これまでに無線 LAN に関する研究が数多く行われてきている [24–31]．[24–27] では，アドホックネットワークにおける経路制御手法が提案されている．アクセスポイントを介さずに無線端末同士が無線によって直接通信アドホックネットワークでは，目的となる端末までの経路を探索する必要がある．そのため，アドホックネットワークにおける経路制御手法がこれまでに提案されている．

また，[28–31] では，省電力化に関する研究が行われている．[28] では，接続される無線端末数が少なく，近距離に複数のアクセスポイントが存在する場合，1 台を残して他のアクセスポイントをスリープさせることで，全体の消費電力を削減させる手法が提案されている．また，端末の消費電力を減少させる手法が提案されている．[29, 30] では，パケットの送受信を行う時間をアクセスポイントが定め，その時間以外は無線機器をスリープさせることで省電力化している．さらに，[31] では，SCTP を利用することによって，無線 LAN における TCP データ転送を省電力化する手法が提案されている．

さらに，無線 LAN においてコネクション間のスループット公平性が失われることが明らかになってい

る [32–34]．この問題は，1 つのアクセスポイントに多数の無線端末が接続した場合に発生する．この問題は，無線 LAN のアクセス制御である CSMA/CA と，TCP の輻輳制御に起因する．今後，無線 LAN の普及はさらに進むと考えられる．また，公衆無線 LAN など，少数のアクセスポイントに対して不特定多数の無線端末が接続される機会が増加すると考えられる．従って，無線 LAN における公平性の問題は重要になると考えられる．

無線 LAN において，コネクション間のスループット公平性が失われる問題に対して，多くの改善手法が提案されている．既存の改善手法は，アクセスポイントに改良を加える手法 [35–38] と，無線端末に対して改良を加える手法 [39] の 2 種類に分類される．アクセスポイントにおける改良手法は，アクセスポイントに送信の優先権を与える手法と，バッファ内パケット数に応じてデータパケットを棄却させる手法が提案されている．しかし，既存のアクセスポイントの多くは，制御機構をハードウェア上に実装している．そのため，既存アクセスポイントに対して新たに改良を加えることは，困難であると考えられる．また，ネットワーク中で複雑な制御を行わない End-to-End の原則に反するため望ましくない．一方，無線端末に対して改良を加える手法は，送信ホストの輻輳制御を改良することで公平性を維持する．しかし，既存の無線端末に対して改良を加える手法は，スループットが低下する問題がある．

本論文では，広帯域なネットワークにおいて今後広く用いられることが考えられる Compound TCP を対象に，無線 LAN における Compound TCP の性能評価を行う．Compound TCP は，TCP と同様の輻輳制御を行う，損失ベースの輻輳制御方式を含んでいる．そのため，無線 LAN 環境下で Compound TCP を用いた場合，TCP と同様にコネクション間のスループット公平性が失われることが考えられる．そこで本論文では，まずシミュレーションにより，無線 LAN 環境下における Compound TCP の性能評価を行う．その結果，無線 LAN 環境において，Compound TCP は TCP と同様に，コネクション間でスループットが不公平になる問題が発生することを明らかにする．さらに本論文では，無線 LAN における Compound TCP の問題を解決する Compound TCP+ を提案する．最後に，Compund TCP+ をシミュレーションおよび実験ネットワークにおいて評価し，無線 LAN 環境下においてスループットの公平性が得られることを示す．さらに，広帯域高遅延なネットワーク環境において，高いスループットを得ることが可能であることを示す．

本論文の構成は以下の通りである．まず，2 章では，現在のインターネットにおいて，標準的に用いられているトランスポート層プロトコルである，TCP について説明する．3 章では，無線 LAN におけるアクセス制御について説明する．4 章では，Compound TCP の輻輳制御アルゴリズムと，無線 LAN における Compound TCP の性能評価について説明する．5 章では，無線 LAN における Compound TCP の問題を解決する，Compound TCP+ の輻輳制御について説明する．さらに，シミュレーションおよび実験ネットワークにおける Compound TCP+ の性能評価を行う．6 章では，広帯域なネットワークにおける Compound TCP+ の性能評価をシミュレーションによって行う．最後に 7 章では，本論文のまとめと今後の課題を述べる．

2 Transmission Control Protocol

現在のインターネットにおける通信の大部分は，Transmission Control Protocol (TCP) [40] をトランスポート層プロトコルに用いた通信である．この TCP はコネクション指向型で信頼性のある通信を提供する．本章では，まず再送制御やフロー制御，輻輳制御といった TCP が行う制御を述べる．その後，広帯域高遅延なネットワークにおいて TCP を用いた時，帯域を有効に活用することができない問題が発生することを述べ，その問題を解決する既存の手法を述べる．

2.1 TCP の概要

TCP は，現在のインターネットにおいて，標準的に用いられているトランスポート層プロトコルである．トランスポート層プロトコルには，TCP のほかに User Datagram Protocol (UDP) がある．UDP は，コネクションレス型で，信頼性のない通信を提供するプロトコルである．それに対して，TCP は，コネクション指向型で信頼性のある通信を提供するプロトコルである．TCP の制御には，受信ホストの性能に応じてパケットの送出量を調節するフロー制御，ネットワークの輻輳状況に応じてパケットの送出量を調節する輻輳制御，送信したパケットが棄却された場合に再送する再送制御がある．

TCP/IP によるパケット通信では，トランスポート層やインターネット層など，各階層においてデータにヘッダが付けられた後，パケットが送信される．トランスポート層プロトコルとして TCP を用いる場合は，トランスポート層において TCP ヘッダが付与される．図 1 に，TCP のヘッダフォーマットを示す．以下に TCP のヘッダフィールドについて，それぞれ説明する．

- 始点ポート番号，終点ポート番号
  16 ビット長のフィールドを持ち，送信元および受信元のポート番号を表す．ポート番号は，アプリケーション毎に利用するポート番号が定められている．FTP や HTTP などの標準的なサービスのポート番号は，0 から 255 の範囲で予約されている．これらのポート番号は，well-known ポート番号と呼ばれている．

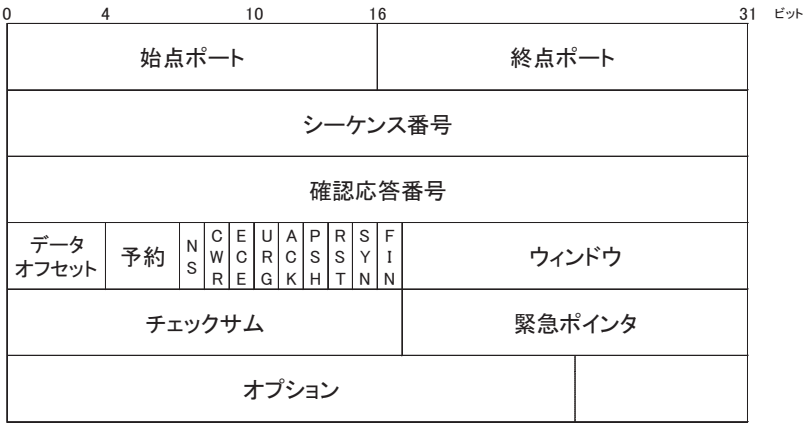


図 1　TCP ヘッダフォーマット

- シーケンス番号
  32 ビット長のフィールドを持ち，シーケンス番号を表す．シーケンス番号は，送信したデータの位置をバイト数で表したものである．データを送信する度に，送信したデータのバイト数だけシーケンス番号が加算される．コネクション確立時や，コネクション切断時に用いる SYN や FIN は，データを含んでいない場合でもシーケンス番号を 1 増加させる．
- 確認応答番号
  32 ビット長のフィールドを持ち，確認応答番号を表す．確認応答番号の値は，この値から 1 を引いた値までのデータを受信したことを意味する．よって，次に送るデータのシーケンス番号と，返された確認応答番号が同じ場合，正常に通信が行われていることを意味する．
- データオフセット
  データオフセットは，ヘッダの先頭からデータ部までのオフセットを 32 ビット単位で表す．データオフセットは，オプションがない場合，5 となる．
- 予約
  予約フィールドは，6 ビット長の将来の拡張のために用意されているフィールドである．
- ウィンドウ
  ウィンドウフィールドは，送信ホストに対して，受信ホストにおけるバッファの空き容量を通知するためのフィールドである．ウィンドウフィールドは，TCP のフロー制御に用いられ，受信ホストの受信バッファの空き容量が設定される．送信ホストは，このフィールドで定められているデータ量以上のデータは送信してはならない．
- チェックサム
  チェックサムフィールドは，TCP のヘッダとデータが破損していないことを確認するために用いる．
- 緊急ポインタ
  緊急ポインタフィールドは，16 ビット長のフィールドを持つ，緊急を要するデータの格納場所を示す．緊急ポインタは，コントロールビットの URG が 1 である場合に有効となる．
- オプション
  オプションフィールドは，TCP による通信の性能を向上させるためのフィールドである．オプションの長さは，最大で 40 バイトである．
- 制御ビット
  制御ビットは，NS，CWR，ECE，URG，ACK，PSH，RST，SYN，FIN の 9 ビットがある．まず，NS は，ECN-nonce 輻輳保護を表すビットである．CWR は，輻輳ウィンドウ縮小を表すビットである．CWR は，ECE フラグがセットされたパケットを受信した場合，輻輳制御で応答する場合にセットされる．ECE は，ECN-Echo を表すビットである．ECE フラグがセットされると同時に SYN フラグが 1 である場合は，ECN が利用可能であることを表す．SYN フラグが 0 でない場合は，IP ヘッダに Congestion Experienced フラグがセットされたパケットを受信したことを表す．URG は，緊急に処理すべきデータが含まれているかを表すビットである．このビットが 1 である場合は，そのデータが緊急に処理すべきデータであることを意味する．ACK は，確認応答番号のフィールドが有効であることを意味する．PSH は，受信したデータをバッファリングの可否を定めるビットである．PSH が 1 に設定されているパケットを受信した場合，データをバッファリングせず，アプリケーションに渡す．一方，PSH が 0 となっているパケットは，バッファリングされる．RST は，コネクションを強制的に初期化する際に用いられる．このビットが 1 である場合，コネクションが強制的に切断される．SYN は，コ

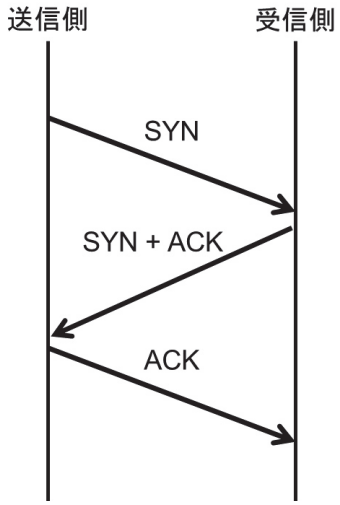


図 2: TCP コネクションの確立

ネクション確立時に用いる．コネクションの確立を行う場合には，SYN を 1 に設定し，シーケンス番号のフィールドにシーケンス番号の初期値を入れる．FIN は，通信終了であることを意味する．今後送信するデータがない場合は，FIN を 1 に設定する．

トランスポート層プロトコルに TCP を用いてデータを送信する場合は，上述したフォーマットに従ったヘッダを付加する．トランスポート層においてヘッダを付加した後，トランスポート層の下位層であるネットワーク層にパケットが渡される．一方，データを受信した際には，ヘッダを取り除きトランスポート層の上位層であるアプリケーション層にデータを渡す．

TCP は，通信相手とのコネクションを確立してから，データの送受信を行うコネクション型のプロトコルである．送受信ホスト間でパケットの送受信を 3 回行う 3 Way handshake と呼ばれる方法によって，TCP はコネクションを確立する．図 2 に，TCP コネクションの確立の流れを示す．送信側ホストはコネクション確立要求 (SYN) フラグが設定されたパケットを，受信側のホストに対して送信する．受信側ホストが SYN を受信した時，SYN に対する確認応答 (ACK) パケットと，送信側ホストに対する SYN を受信側ホストは送信する．送信側ホストが，受信側ホストからの SYN を受信した時，それに対する ACK を受信側ホストに対して送信し，コネクションが確立される．

次に，図 3 に，TCP コネクションの切断の流れを示す．データの送信が終了した時，送信側ホストはコネクション開放要求 (FIN) フラグがセットされたパケットを受信側ホストに対して送信する．受信側ホストは FIN を受信した時，それに対する ACK パケットを送信し，その後，送信側ホストに対する FIN を送信する．送信側ホストが，受信側ホストからの FIN を受信した時，それに対する ACK パケットを受信側ホストに対して送信し，コネクションが切断される．

2.2 再送制御

TCP は，送出したデータパケットが送信先のホストに到着することを保証する，信頼性のある通信を行うプロトコルである．データパケットが送信先ホストに到着することを保証するために，TCP はパケットが棄

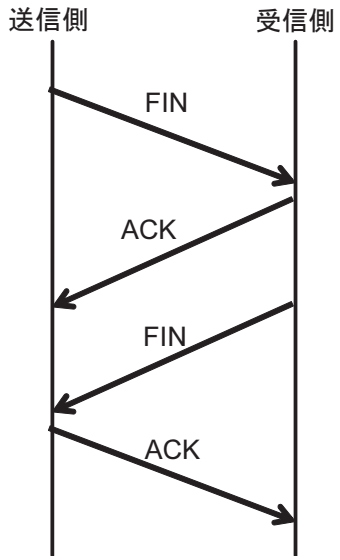


図 3: TCP コネクションの切断

却された場合，そのパケットを再送する再送制御を行う．これを実現するために，TCP は確認応答 (ACK) パケットを用いる．TCP は，受信ホストがデータパケットを受信した場合，そのデータパケットを受信したことを通知する ACK パケットを送信ホストに対して送信する．受信ホストからの ACK パケットを受信することによって，送信ホストは送信したパケットが到着したことを確認する．しかし，送信ホストに ACK パケットが到着しない場合は，何らかの原因によりデータパケットが棄却されたと考えられる．この時，TCP は再送制御によって，棄却されたパケットを再送する．具体的には，TCP は，以下のようにパケット棄却を検出し，パケットの再送を行う．

パケットが棄却された場合，送信ホストからの新たにデータパケットを受信した場合でも，受信ホストは同じシーケンス番号の ACK パケットを送信ホストに対して送信する．よって，送信ホストは，同じシーケンス番号の ACK パケットを複数回受信することとなる．この時，送信ホストにおいて同一シーケンス番号の ACK パケットを 3 個受信した場合，TCP はパケット棄却が発生したと判断し再送する．

また，送信ホストがデータパケットを送信した後，一定時間が経過した場合でも ACK パケットが到着しない場合は，パケット棄却が発生したと判断しパケットを再送する．この再送を行うまでの時間を再送タイムアウト値と呼ぶ．再送タイムアウト値を大きくした場合，パケット棄却が発生した時の待ち時間が長くなり性能が劣化する．一方，再送タイムアウト値を小さくしすぎた場合，確認応答パケットが到着する前にパケットを再送し，ネットワークに不要な負荷をかける．このように，通信の効率に大きな影響を与えるため，再送タイムアウト値を適切に設定することが重要である．

TCP では，再送タイムアウト値を，ラウンドトリップ時間に基づいて決定する．しかし，ラウンドトリップ時間は，ネットワークの状況によって常に変化する．したがって，再送タイムアウト値を適切に設定することは困難である．そこで，次式のようにラウンドトリップ時間を平滑化し，その値を用いる．

$$sRTT = \alpha RTT + (1 - \alpha) sRTT \tag{1}$$

α は平滑化のためのパラメータであり，その推奨値は 1/8 である．さらに $sRTT$ は，平滑化されたラウンド

トリップ時間を表し，RTT は現在のラウンドトリップ時間を表す．次に，平均偏差 D を求める．D は次式のように求められる．

$$D = (1 - \beta)D + \beta|sRTT - RTT| \tag{2}$$

偏差の係数 β の推奨値は 1/4 である．$sRTT$ および D を用いて，次式のように再送タイムアウト値を決定する．

$$RTO = sRTT + 4D \tag{3}$$

2.3 フロー制御

TCP は，ウィンドウ制御により，確認応答パケットなしに複数のパケットをネットワークに対して送出することが可能である．しかし，受信側ホストが持つバッファの状況を考慮せずに大量のパケットを送出した場合，バッファが溢れパケット棄却が発生する場合がある．TCP は，パケット棄却を検出した時，棄却されたパケットを再送するため，受信側ホストのバッファサイズを超えるパケット数を送出することは，データ転送の効率を低下させる．そのため，TCP では，フロー制御を行う事によって，送信側ホストは受信側ホストの状況に合わせてパケット送出量を調節している．

具体的には，受信ホストは広告ウィンドウと呼ばれる，受信ホストが持つバッファの空き容量を ACK パケットのヘッダに設定し，送信ホストへ送信する．送信ホストは，通知された広告ウィンドウの値を超えない範囲でパケットを送出する．これによって，受信ホストの処理能力を超えない速度でデータを送信することができる．

2.4 輻輳制御

TCP は，輻輳ウィンドウと呼ばれる 1 ラウンドトリップ時間あたりに送出できるパケット数を動的に増減させることで，データ転送速度を調節している．ここでは，標準的な TCP である，TCP NewReno の輻輳制御アルゴリズムを説明する．TCP NewReno は，パケットの棄却を輻輳の指標とする，損失ベースの輻輳制御手法を用いる．図 4 に TCP NewReno の輻輳ウィンドウの変化を示す．図 4 に示すように，TCP NewReno の輻輳制御方式は，スロースタートフェーズと輻輳回避フェーズと呼ばれる 2 つのフェーズから構成されており，それぞれのフェーズによって，輻輳ウィンドウの増加速度が異なる．

まず，TCP NewReno は，送受信ホスト間のコネクションが確立されデータの送受信が開始される時，スロースタートフェーズとなる．TCP NewReno は，スロースタートフェーズ時，ACK パケットを受信するごとに輻輳ウィンドウを次式のように増加させる．

$$cwnd = cwnd + 1 \tag{4}$$

図 5 に，スロースタートフェーズ時の通信の様子を示す．まず，TCP は，スロースタートフェーズ開始時，輻輳ウィンドウを 1 に設定し，データパケットを 1 つ送信する．そのデータパケットに対する確認応答パケットを受信した時，TCP は，輻輳ウィンドウを 1 増加させ 2 に設定する．そのため，データパケットを 2 つ送信する．さらにその確認応答パケットを受信した時，TCP は輻輳ウィンドウを 4 に設定し，パケットを 4 つ送信する．このように TCP NewReno は，スロースタートフェーズ時，輻輳ウィンドウを指数的に増加させる．

TCP はスロースタートフェーズ後，輻輳回避フェーズに移行する．TCP は，輻輳ウィンドウの値がスロースタート閾値と呼ばれる変数の値を超えると，スロースタートフェーズから輻輳回避フェーズへと移行する．

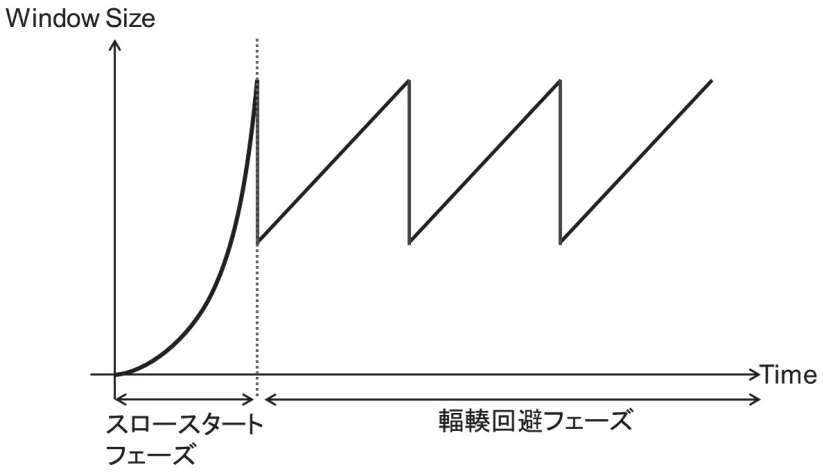


図 4: TCP NewReno の輻輳ウィンドウの変化

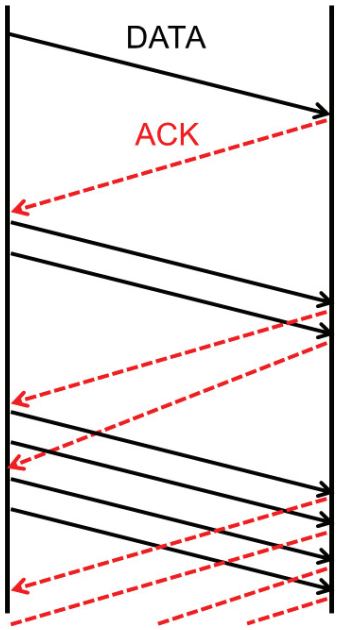


図 5: スロースタートフェーズ時における TCP NewReno の通信

スロースタート閾値の値は，TCP が輻輳を検出した際に設定される．TCP は，パケット棄却を検出した時，輻輳ウィンドウを減少させることで輻輳を回避する．この時，TCP は，パケット棄却を検出した時点における輻輳ウィンドウの値を半分にした値を，スロースタート閾値として設定する．TCP NewReno は，輻輳回避フェーズ時，1 ラウンドトリップ時間毎に輻輳ウィンドウを次式のように増加させる．

$$cwnd = cwnd + \frac{1}{cwnd} \tag{5}$$

輻輳回避フェーズでは輻輳ウィンドウを線形的に増加させる．図 6 に，輻輳回避フェーズ時の通信の様子を示

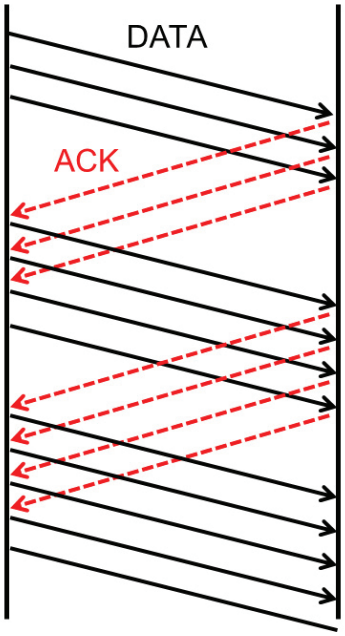


図 6: 輻輳回避フェーズ時における TCP NewReno の通信

す．ここでは，輻輳回避フェーズに移行した時の輻輳ウィンドウを 3 とする．輻輳ウィンドウが 3 であるため，確認応答パケットを受信する毎に輻輳ウィンドウが 3 分の 1 増加する．さらに，確認応答パケットを 3 つ受信することになるため，3 つめの確認応答パケットを受信した後に輻輳ウィンドウの値は 4 となる．

また，パケット棄却を検出した場合には，そのパケット棄却の検出方法によって，輻輳ウィンドウサイズの減少量が異なる．輻輳ウィンドウサイズの減少量は次式のように計算される．

$$cwnd = \begin{cases} \dfrac{cwnd}{2} & \text{(重複 ACK を受信した場合)} \\ 1 & \text{(タイムアウトが発生した場合)} \end{cases} \tag{6}$$

重複 ACK を受信した場合のパケット棄却は，ネットワークに軽度の輻輳が発生したと判断して輻輳ウィンドウを現在の値の半分に減少させる．一方，タイムアウトによるパケット棄却を検出した場合は，ネットワークに重度の輻輳が発生したと判断して，輻輳ウィンドウを 1 に減少させる．

2.5 広帯域ネットワークにおける TCP の問題

現在のインターネットにおいて，標準的に用いられいているトランスポート層プロトコルである TCP NewReno では，広帯域高遅延なネットワーク環境において，ネットワーク帯域を十分に活用することができない問題が明らかになっている [41].

この問題は，TCP NewReno の輻輳制御アルゴリズムに起因する．図 7 に，帯域が 10 [Gbit/s]，ラウンドトリップ時間が 100 [ms] である広帯域高遅延なネットワーク環境において，パケットサイズが 1500 [Byte] である TCP NewReno コネクションが通信を行ったときの輻輳ウィンドウの変化を示す．TCP NewReno の輻輳制御アルゴリズムでは，輻輳ウィンドウの増加幅が式 (5) のように，1 ラウンドトリップ時間毎に 1 パケットと非常に小さい．そのため，TCP NewReno を用いるコネクションが帯域を使い切ることを考えた場合，パケッ

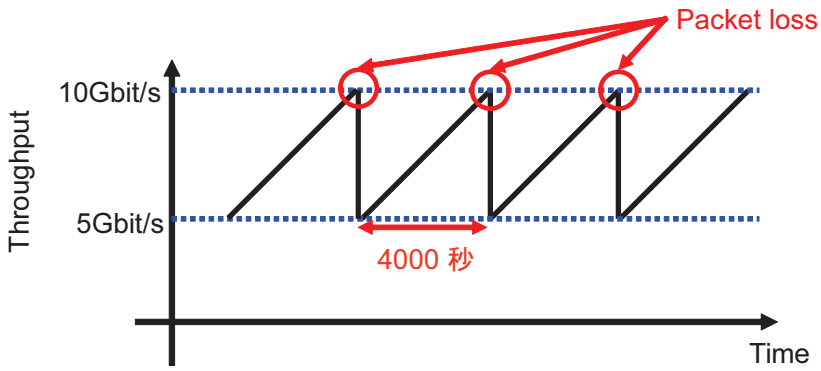


図 7: 広帯域高遅延環境における TCP NewReno の問題点

ト棄却率が 2×10^{-10} 以下である必要がある．現在の技術でこのパケット棄却率を実現することは難しい．

さらに，TCP NewReno は，式 (6) のように，パケット棄却を検出した時，非常に大きく輻輳ウィンドウ減少させる．図 7 のネットワークにおいて一度パケット棄却が発生した場合，再び帯域を使い切るために必要な輻輳ウィンドウサイズにまで回復するには，4000 秒の間パケット棄却を起こしてはならない．しかしながら，現在のネットワーク技術では，これを達成することは困難である．したがって，TCP NewReno を用いて，広帯域高遅延なネットワーク帯域を使い切ることは困難である．

2.6 既存の広帯域ネットワーク向けトランスポート層プロトコル

広帯域高遅延な環境において TCP NewReno を用いた場合，帯域を有効に活用することができない問題がある．そこで，これまでに広帯域高遅延なネットワークにおける TCP の問題を改善する手法が提案されている．本節では，その改善手法を説明する．

2.6.1 HighSpeed TCP

HighSpeed TCP [13] は，パケット棄却を輻輳の指標とする損失ベースの輻輳制御を用いる．図 8 に，HighSpeed TCP と TCP NewReno の輻輳ウィンドウの増減を示す．図 8 に示すように，HighSpeed TCP の輻輳制御アルゴリズムは，TCP NewReno と同様に，パケット棄却を検出するまで輻輳ウィンドウを線形的に増加させ，パケット棄却検出時には輻輳ウィンドウを減少させる．TCP NewReno は，輻輳ウィンドウの増加幅が 1 ラウンドトリップ時間ごとに 1 パケットと小さい．さらに，パケット棄却時には輻輳ウィンドウを半減させる．そのため，広帯域高遅延なネットワーク環境において，帯域を有効に使用することができない．そこで，HighSpeed TCP は，TCP NewReno と比べて，輻輳回避フェーズにおける輻輳ウィンドウの増加幅を大きくする．また，TCP NewReno と比較して，HighSpeed TCP はパケット棄却検出時における輻輳ウィンドウの減少幅を小さくする．具体的には，HighSpeed TCP は，次式に従って輻輳ウィンドウの増減を行う．

$$win = \begin{cases} win + \frac{a(win)}{win} & (\text{パケット棄却未検出時}) \\ (1 - b\,(win))\, win & (\text{パケット棄却検出時}) \end{cases} \tag{7}$$

win は現在の輻輳ウィンドウ，$a(win)$ は，1 ラウンドトリップ時間毎の輻輳ウィンドウの増加幅，$b(win)$ はパケット棄却検出時の輻輳ウィンドウの減少幅を表す．輻輳ウィンドウの増加幅 $a(win)$ および，輻輳ウィン

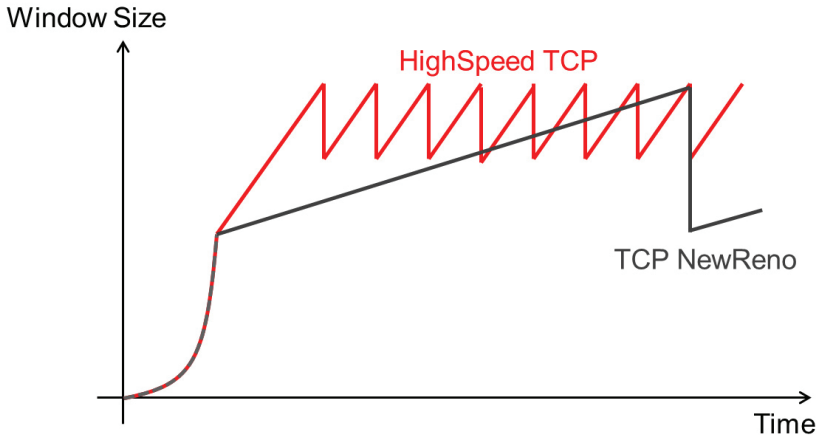


図 8: TCP NewReno と HighSpeed TCP の輻輳ウィンドウの増減

ドウの減少幅 $b(win)$ は以下の式に従って求められる．

$$a(w) = \frac{2w^2 \cdot b(w) \cdot p(w)}{2 - b(w)} \tag{8}$$

$$b(w) = \frac{\log(w) - \log(W_{low})}{\log(W_{high}) - \log(W_{low})} \tag{9}$$

$$p(w) = \exp\left(\frac{\log(w) - \log(W_{low})}{\log(W_{high}) - \log(W_{low})} \cdot (\log(P_{high}) - \log(P_{low})) + log(P_{low})\right) \tag{10}$$

P_{low} は，TCP NewReno において平均輻輳ウィンドウが W_{low} となる時のパケット棄却率を表す．また，$W_{low}, P_{high}, b_{high}$ は，HighSpeed TCP のパラメータであり，論文 [13] では，$W_{low} = 38, P_{high} = 10^{-7}, b_{high} = 0.1$ と定められている．

HighSpeed TCP は，TCP NewReno と比較して，輻輳回避フェーズにおける輻輳ウィンドウの増加幅を大きく，パケット棄却検出時における輻輳ウィンドウの減少幅を小さくする．これにより，広帯域高遅延なネットワーク環境において帯域を有効に活用することが可能である．しかし，HighSpeed TCP は TCP NewReno との公平性を考慮していない．そのため，TCP NewReno を用いるコネクションと HighSpeed TCP が競合した場合，HighSpeed TCP を用いるコネクションが，TCP NewReno を用いるコネクションの帯域を不当に占有する問題が存在する [42]．これは，以下の理由からである．パケット棄却が発生した場合，TCP NewReno は輻輳ウィンドウを半減させ，その後 1 RTT に輻輳ウィンドウを 1 増加させる．一方，HighSpeed TCP はパケット棄却が発生した場合でも TCP NewReno のように大きく輻輳ウィンドウを減少させず，また，輻輳ウィンドウの増加幅は TCP NewReno と比較して大きい．したがって，TCP NewReno が十分な輻輳ウィンドウサイズにまで回復する前に，HighSpeed TCP がより大きな輻輳ウィンドウサイズを得る．その結果，TCP NewReno は輻輳ウィンドウを増加させることができず，HighSpeed TCP に帯域を占有される．

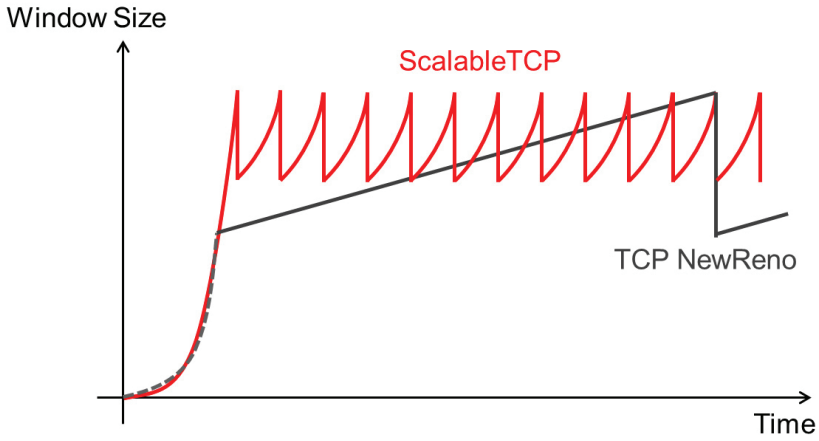


図 9: TCP NewReno と Scalable TCP の輻輳ウィンドウの増減

2.6.2 Scalable TCP

Scalable TCP [14] の輻輳制御は，パケット棄却を輻輳の指標とする損失ベースの輻輳制御である．図 9 に，TCP NewReno と Scalable TCP の輻輳ウィンドウの増減を示す．図 9 に示すように，Scalable TCP は，輻輳回避フェーズにおいて，輻輳ウィンドウを指数的に増加させる．具体的には，ACK パケットを受信する度に，以下の式に基づいて輻輳ウィンドウを増加させる．

$$win = win + \alpha \tag{11}$$

ここで，α は $0 < \alpha < 1$ の値をとる．パケット棄却が発生しない場合，1 ラウンドトリップ時間に win 個の ACK パケットを受け取る．従って，式 (11) から，1 ラウンドトリップ時間後のウィンドウサイズは，以下の式で与えられる．

$$win = (1 + \alpha)win \tag{12}$$

式 (12) から，Scalable TCP の輻輳ウィンドウサイズは，輻輳回避フェーズにおいて，1 ラウンドトリップ時間に，指数的に増加することがわかる．また，パケット棄却を検出した時，Scalable TCP は次式のように輻輳ウィンドウを減少させる．

$$win = win - \lceil b \times win \rceil \tag{13}$$

b は $0 < b < 1$ の値をとる．論文 [14] では，a を 0.01 とし，b を 0.125 としている．

Scalable TCP は，輻輳回避フェーズにおいて輻輳ウィンドウを指数的に増加させることで，広帯域高遅延なネットワーク環境においても高いスループットを得ることが可能である．Scalable TCP は，輻輳回避フェーズにおいて TCP NewReno よりも輻輳ウィンドウの増加幅を大きくすることで高いスループット得る HighSpeed TCP と類似した手法である．したがって，HighSpeed TCP と同様に，Scalable TCP と TCP NewReno が競合した場合，TCP NewReno を用いるコネクションの帯域を不当に奪う問題が発生すると考えられる．

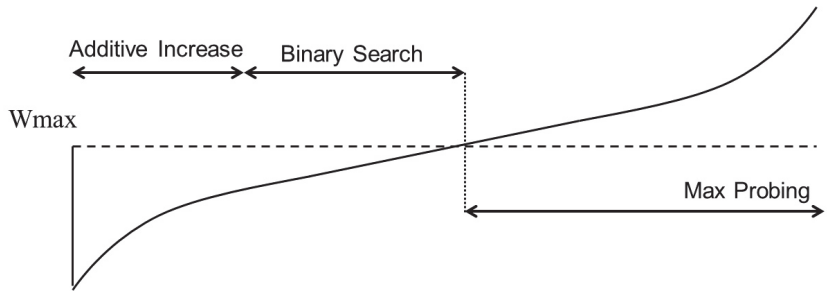


図 10: BIC TCP の輻輳ウィンドウの増加

2.6.3 BIC TCP

Binary Increase Control TCP (BIC TCP) [15] は，Linux Kernel 2.6.8 から 2.6.18 までデフォルトで用いられてきた輻輳制御手法である．BIC TCP の輻輳制御は，パケット棄却を輻輳の指標とする損失ベースの輻輳制御方式である．図 10 に，BIC TCP の輻輳ウィンドウの変化を示す．BIC TCP は，輻輳ウィンドウの増加方法として，Additive Increase モード，Binary Search モード，Max Probing モードの 3 つのモードを持つ．まず，輻輳ウィンドウ減少後の輻輳ウィンドウが，パケット棄却発生時の輻輳ウィンドウと大きく離れる場合，Additive Increase モードとなる．Additive Increase モードでは，次式のように輻輳ウィンドウを増加させる．

$$W_{inc} = cwnd + S_{max}/cwnd \quad (W_{inc} > S_{max}) \tag{14}$$
$$W_{inc} = S_{min}/cwnd \quad (W_{inc} < S_{min}) \tag{15}$$
$$cwnd = cwnd + W_{inc}/cwnd \tag{16}$$

S_{max} は，輻輳ウィンドウ増加の最大値で，論文 [15] では，32 と定められている．さらに，S_{min} は，輻輳ウィンドウ増加の最小値で，論文 [15] では，0.01 と定められている．BIC TCP は，Additive Increase モードである時，輻輳ウィンドウを線形的に増加させる．Binary Search モードでは，輻輳ウィンドウを次式のように増加させる．

$$W_{inc} = W_{max} - cwnd \quad (cwnd < W_{max}) \tag{17}$$
$$W_{inc} = cwnd - W_{max} \quad (cwnd \geq W_{max}) \tag{18}$$
$$cwnd = cwnd + W_{inc}/cwnd \tag{19}$$

W_{max} は，パケット棄却が発生した時の輻輳ウィンドウを表す．

BIC TCP は，パケット棄却発生時の輻輳ウィンドウを記録し，パケット棄却後に減少した輻輳ウィンドウの値との間で二分探索を用いて輻輳ウィンドウを増加させる．さらに，パケット棄却発生時の輻輳ウィンドウにまで回復した場合，Max Probing モードとなる．Max Probing モードでは，Binary Increase モードと，Additive Increase モードと対称となるように輻輳ウィンドウを増加させる．

BIC TCP は，広帯域高遅延なネットワーク環境において，高いスループットを獲得することが可能である．しかし，BIC TCP は，ラウンドトリップ時間が小さいネットワークや，低速なネットワーク環境においては，TCP NewReno よりも輻輳ウィンドウが大きく増加する．したがって，そのようなネットワークにおいて，BIC TCP と TCP NewReno が共存した場合，公平性が失われる問題がある [16].

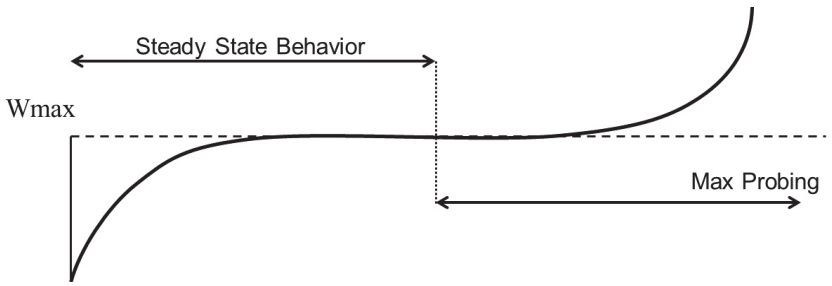


図 11: CUBIC TCP の輻輳ウィンドウの増加

2.6.4 CUBIC TCP

CUBIC TCP [16] は，Linux kernel 2.6.19 から現在に至るまで，デフォルトで用いられている輻輳制御である．CUBIC TCP の輻輳制御は，パケット棄却を輻輳の指標とする損失ベースの輻輳制御である．図 11 に，CUBIC TCP での輻輳ウィンドウの変化を示す．CUBIC TCP は，パケット廃棄時からの経過時間の 3 次間数で輻輳ウィンドウの増加幅が決定される Steady State Behavior モードと，指数関数的に輻輳ウィンドウを増加させる Max Probing モードの 2 つのモードが存在する．

CUBIC TCP は，ACK パケットを受信するごとに，以下の式に従って輻輳ウィンドウを増加させる．

$$cwnd = (Ct - K)^3 + cwnd_{max} \tag{20}$$

C は CUBIC TCP の定数であり，論文 [16] では，0.4 に定めている．さらに，$cwnd_{max}$ は，前回のパケット棄却発生時点の輻輳ウィンドウ，t は，前回のパケット棄却からの経過時間を表す．また，K は，次式より求められる．

$$K = \sqrt[3]{\frac{\beta cwnd_{max}}{C}} \tag{21}$$

β は，パケット棄却時の減少幅を表す．論文 [16] では，β を 0.2 と定めている．パケット棄却によって輻輳ウィンドウが減少する前の輻輳ウィンドウサイズを W_{max} としたとき，CUBIC TCP は，W_{max} まで回復させるように輻輳ウィンドウを増加させる．W_{max} まで輻輳ウィンドウが回復した時，CUBIC TCP は Max Probing モードに入り，さらに輻輳ウィンドウを大きく増加させる．また，パケット棄却を発生した時，CUBIC TCP は次式のように輻輳ウィンドウを減少させる．

$$cwnd = cwnd(1 - \beta) \tag{22}$$

パケット棄却発生からの経過時間をもとに，CUBIC TCP は輻輳ウィンドウの増加幅を決定する．そのため，ラウンドトリップ時間が異なるフローが競合する場合でも公平性が高い通信が可能である．しかし，CUBIC TCP を用いた輻輳ウィンドウが大きいコネクションと競合した場合，公平性のある通信を行うには多大な時間を要する問題があることが，シミュレーションによる評価から明らかになっている [43].

2.6.5 FAST TCP

FAST TCP [17] は，ラウンドトリップ時間を輻輳の指標とする，遅延ベースの輻輳制御である．FAST TCP は，現在のラウンドトリップ時間と，今までに観測された最小のラウンドトリップ時間を用いて，次式のよう

に輻輳ウィンドウを増加させる．

$$cwnd = min\left(2cwnd, (1-\gamma)\,cwnd + \gamma\left(\frac{baseRTT}{avgRTT}cwnd + \alpha\right)\right) \tag{23}$$

$baseRTT$ は，これまでに観測された最小のラウンドトリップ時間を表し，$avgRTT$ は現在のラウンドトリップ時間の平均を表す．また，γ は，0 から 1 までの間の値をとる FAST TCP のパラメータである．さらに，α は，ネットワーク中にバッファリングされているパケット数の目標値である．

FAST TCP は，同じ遅延ベースの輻輳制御を行う TCP Vegas と比較して，輻輳ウィンドウの増加幅が大きく，広帯域高遅延なネットワーク環境においても，その帯域を有効に活用することが可能である．しかし，ネットワークの状況は，常に変化する．そのため，α の適切な設定は困難であると考えられる．

2.6.6 TCP Westwood

TCP Westwood [18] は，ACK パケットの到着間隔からネットワークの利用可能帯域を推測し，その推測値をもとに輻輳制御を行う．具体的には，スロースタートフェーズにおいて輻輳ウィンドウを指数的に増加させ，輻輳回避フェーズにおいて輻輳ウィンドウを線形的に増加させる TCP NewReno と同一の制御を TCP Westwood は行う．一方，パケット棄却が発生した場合，TCP Westwood は次式のように輻輳ウィンドウとスロースタート閾値を設定する．

$$ssthresh = (BWE \cdot baseRTT)\,/seg_{size} \tag{24}$$

$$cwnd = ssthresh \qquad (cwnd > ssthresh) \tag{25}$$

BWE は利用可能帯域の推定値であり，$baseRTT$ はコネクション確立後に観測した最小のラウンドトリップ時間である．BWE は，次式に従って求められる．

$$BWE = d_k/\,(t_k - t_{k-1}) \tag{26}$$

d_k は，転送するデータのバイト数，t_k は，ACK パケットが到着した時間，t_{k-1} は，1 つ前の ACK パケットが到着した時間を表す．TCP NewReno のようにパケット棄却が発生した場合でも，TCP Westwood は輻輳ウィンドウを極端に減少させない．また，タイムアウトが発生した場合，TCP Westwood は，次式のように輻輳ウィンドウとスロースタート閾値を制御する．

$$thresh = (BWE \cdot baseRTT)\,/seg_{size} \tag{27}$$

$$ssthresh = 2 \qquad (thresh < 2) \tag{28}$$

$$ssthresh = thresh \qquad (thresh \geq 2) \tag{29}$$

$$cwnd = 1 \tag{30}$$

タイムアウトが発生した場合，TCP Westwood は，BWE を用いてスロースタート閾値を設定し，輻輳ウィンドウを 1 に設定する．

TCP Westwood は，パケット棄却が発生した場合，ACK パケットの到着間隔からネットワークの輻輳状況を推測し，その値をもとに輻輳ウィンドウの減少幅を決定する．そのため，TCP Westwood は不必要な輻輳ウィンドウの減少を回避することが可能である．しかし，一時的に ACK パケットの到着間隔が大きく変化した場合，ネットワークの輻輳状態を過大に評価する恐れがある．

2.6.7 TCP Westwood+

TCP Westwood+ [19] は，ラウンドトリップ時間を用いて利用可能帯域を推測し，その値にもとづいて輻輳制御を行う．まず，TCP Westwood+ では，TCP Westwood と同様に TCP Reno と同一の制御によって，輻輳ウィンドウを増加させる．パケット棄却を検出した場合，TCP Westwood+ は，次式のように輻輳ウィンドウとスロースタート閾値を設定する．

$$ssthresh = max\left(2, \left(BWE \cdot baseRTT\right)/seg_{size}\right) \tag{31}$$

$$cwnd = ssthresh \tag{32}$$

BWE は利用可能帯域の推定値であり，$baseRTT$ はコネクション確立後に観測した最小のラウンドトリップ時間である．TCP Westwood+ では，BWE を次式のように求める．

$$BWE = d_k/RTT \tag{33}$$

d_k は，RTT 間に受信したデータ量を表し，RTT は，現在のラウンドトリップ時間を表す．さらに，タイムアウトが発生した場合，TCP Westwood+ は次式のように輻輳ウィンドウとスロースタート閾値を設定する．

$$ssthresh = \left(BWE \cdot baseRTT\right)/seg_{size} \tag{34}$$

$$cwnd = 1 \tag{35}$$

TCP Westwood は，ネットワークの輻輳状態を ACK パケットが到着する毎に推測する．そのため，一時的に ACK パケットの到着間隔が大きく変化した時，ネットワークの輻輳状態を過大に評価する問題がある．それに対して，1 ラウンドトリップ時間毎に輻輳状態を推測することで，TCP Westwood の問題を解決している．しかし，ラウンドトリップ時間を輻輳の指標とするため，パケット棄却を輻輳の指標とする損失ベースの輻輳制御と競合した場合，帯域が不当に奪われる恐れがある．

2.6.8 TCP Fusion

TCP Fusion [20] は，パケット棄却を輻輳の指標とする損失ベースの輻輳制御と，ラウンドトリップ時間を輻輳の指標とする遅延ベースの輻輳制御を組み合わせた，ハイブリッド型の輻輳制御を行う．まず，TCP Fusion は，次式を用いてボトルネックとなるルータにバッファリングされているパケット数 *diff* を推測する．

$$\textit{diff} = cwnd\frac{(congRTT - baseRTT)}{congRTT} \tag{36}$$

$baseRTT$ は，最小のラウンドトリップ時間を表し，$congRTT$ は，パケット棄却が発生する直前のラウンドトリップ時間である．この $diff$ をもとに，TCP Fusion は次式のように輻輳ウィンドウを増加させる．

$$cwnd = \begin{cases} cwnd + W_{inc}/cwnd \\ \quad (diff < \alpha) \\ cwnd + \left(-\textit{diff} + \alpha\right)/cwnd \\ \quad (\textit{diff} > 3\alpha) \\ cwnd \\ \quad (otherwise) \end{cases} \tag{37}$$

$$cwnd = reno\_cwnd \qquad (cwnd < reno\_cwnd) \tag{38}$$

α は，ネットワークの輻輳状況を判断する閾値であり，W_{inc} は，輻輳ウィンドウを急速に増加させるためのパラメータである．さらに，$reno\_cwnd$ は TCP Reno と同一の増加幅で増加するウィンドウである．

TCP Fusion は，TCP Vegas と同様に輻輳ウィンドウの増加に 3 つのフェーズがある．まず，TCP Fusion は，*diff* が閾値よりも下回る場合，ネットワークに空き帯域があると判断し，輻輳ウィンドウを急速に増加させる．一方，*diff* が閾値を上回る場合，TCP Fusion は，ネットワークが輻輳状態にあると判断し，輻輳ウィンドウを減少させる．また，*diff* が閾値の下限と上限の間に入るような，ネットワークが最適な状態であると考えられる場合，TCP Fusion は輻輳ウィンドウ増加を停止する．また，パケット棄却が発生した場合，TCP Fusion は，次式のように輻輳ウィンドウを減少させる．

$$cwnd = max\left(cwnd\frac{baseRTT}{congRTT}, \frac{cwnd}{2}\right) \tag{39}$$

TCP Fusion は，パケット棄却が発生した場合，TCP Westwood の輻輳制御に基づいて損失ウィンドウの減少を行う．これにより，TCP のようなパケット棄却発生時に大きく輻輳ウィンドウが減少しない．

2.6.9 TCP AdaptiveReno

TCP AdaptiveReno(TCP-AReno) [21] は，パケット棄却を輻輳の指標とする損失ベースの輻輳制御と，ラウンドトリップ時間を輻輳の指標とする遅延ベースの輻輳制御を組み合わせた，ハイブリッド型の輻輳制御を行う．まず，TCP-AReno は，ラウンドトリップ時間から，次式を用いてネットワークの輻輳レベル c を推定する．

$$c = min\left(\frac{sRTT - baseRTT}{congRTT - baseRTT}, 1\right) \tag{40}$$

$sRTT$ は，平滑化された現在のラウンドトリップ時間，$baseRTT$ は，これまでに観測された最小のラウンドトリップ時間，$congRTT$ は，パケット棄却が発生する直前のラウンドトリップ時間を表す．$congRTT$ は，次式を用いて求める．

$$congRTT = (1-a)\,congRTT + sRTT \cdot a \tag{41}$$

TCP-AReno は，c を用いて，次式のように輻輳ウィンドウの増減を行う．

$$cwnd = w_{base} + w_{probe} \tag{42}$$
$$w_{base} = w_{base} + 1 \tag{43}$$
$$w_{probe} = max\left(w_{probe} + W_{inc}/cwnd, 0\right) \tag{44}$$
$$W_{inc}\left(c\right) = W_{inc}^{max}/e^{\alpha_a c} + \beta_a c + \gamma_a \tag{45}$$
$$W_{inc}^{m}ax = B/M \cdot MSS \tag{46}$$
$$\beta_a = 2W_{inc}^{max}\left(1/\alpha_a - 1\left(1/a_a + 1\right)/e^{\alpha_a}\right) \tag{47}$$
$$\gamma_a = 1 - 2W_{inc}^{max}\left(1/\alpha_a - \left(1/\alpha_a + 1/2\right)/e^{\alpha_a}\right) \tag{48}$$

MSS は最大セグメントサイズ，B は，TCP Westwood と同様に ACK の到着間隔から求められる帯域の推定値，M はスケーリング関数である．

図 12 に，TCP-AReno の輻輳ウィンドウの変化を示す．図 12 に示すように，TCP-AReno では，w_{base} と w_{probe} の 2 つのウィンドウが存在し，その和が TCP-AReno の輻輳ウィンドウとなっていることがわかる．w_{base} は，損失ベースの輻輳制御である TCP Reno と同一の増加幅で増加するウィンドウであり，w_{probe} は，遅延ベースの輻輳制御で増加するウィンドウである．TCP-AReno は，c により，ネットワークに空き帯域があると考えられる場合，w_{probe} の増加幅を大きくすることによって，空き帯域を有効に活用するように制御する．また，c が 1 に近づくにつれて，w_{probe} の増加量は減少し，w_{base} による増加によってのみ輻輳ウィンドウが変化する．すなわち，TCP Reno と同一の増加幅で輻輳ウィンドウが増加する．

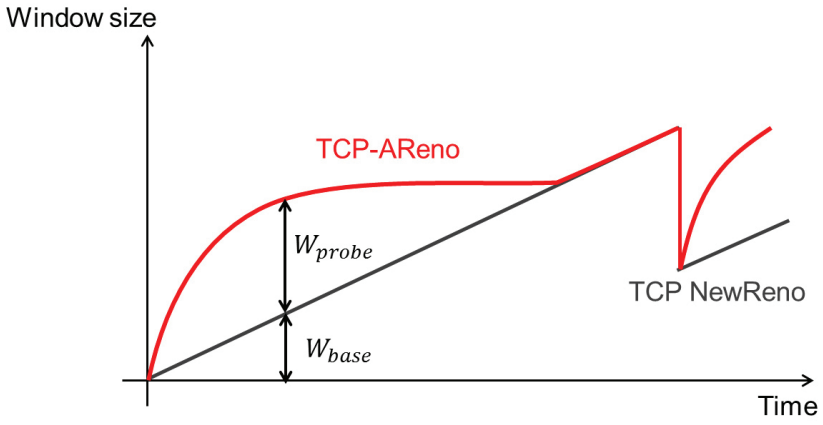


図 12: TCP AReno の輻輳ウィンドウの変化

また，パケット棄却が発生した場合，TCP Westwood は c に基づいて輻輳ウィンドウの減少幅を決定する．具体的には，次式のように輻輳ウィンドウを決定する．

$$w_{base} = cwnd/(1+c) \tag{49}$$

$$w_{probe} = 0 \tag{50}$$

TCP-AReno は，パケット棄却が発生した場合，遅延ベースの輻輳制御に相当するウィンドウである w_{probe} を 0 に設定し，損失ベースの輻輳制御に相当する w_{probe} を c に基づいて減少させる．TCP-AReno では，c が 0 に近い値である場合に発生するパケット棄却は，非輻輳のパケット棄却と判断し，輻輳ウィンドウの減少幅を小さくする．これによって，ランダムなパケット棄却発生時のスループットを向上させることが可能である．また，c が 1 に近い値である場合に発生するパケット棄却は，輻輳によるパケット棄却と判断し，輻輳ウィンドウを半減させる．これにより，TCP-AReno は，TCP と競合した場合でも，TCP の帯域を奪わず同等程度のスループット得る TCP-Friendly な動作を行う．

2.6.10 Compound TCP

Compound TCP は，パケット棄却を輻輳の指標とする損失ベースの輻輳制御と，ラウンドトリップ時間を輻輳の指標とする遅延ベースの輻輳制御を組み合わせた，ハイブリッド型の輻輳制御である．図 13 に，Compound TCP の輻輳ウィンドウの変化を示す．図 13 のように，Compound TCP は，損失ベースの輻輳制御で増減が行われる損失ウィンドウと，遅延ベースの輻輳制御で増減が行われる遅延ウィンドウの和が，輻輳ウィンドウとなる．損失ウィンドウは，TCP と同様にパケット棄却が発生するまで線形的に輻輳ウィンドウを増加させる．一方で，遅延ウィンドウは，ラウンドトリップ時間からネットワークの輻輳状況を推測し，空き帯域があれば急激に遅延ウィンドウを増加させることで，広帯域高遅延なネットワークにおいて，帯域を有効に活用する．さらに，Compound TCP は，ネットワークの輻輳状況の悪化が進むほど，遅延ウィンドウは減少し動作が停止する．その結果，Compound TCP は，損失ベースの輻輳制御によって輻輳ウィンドウが決定される．したがって Compound TCP は，TCP Friendly な通信が可能である．

このように，広帯域高遅延なネットワークにおける TCP の問題を解決する手法として，これまでに損失ベースの輻輳制御，遅延ベースの輻輳制御，損失ベースの輻輳制御と遅延ベースの輻輳制御を組み合わせたハイブ

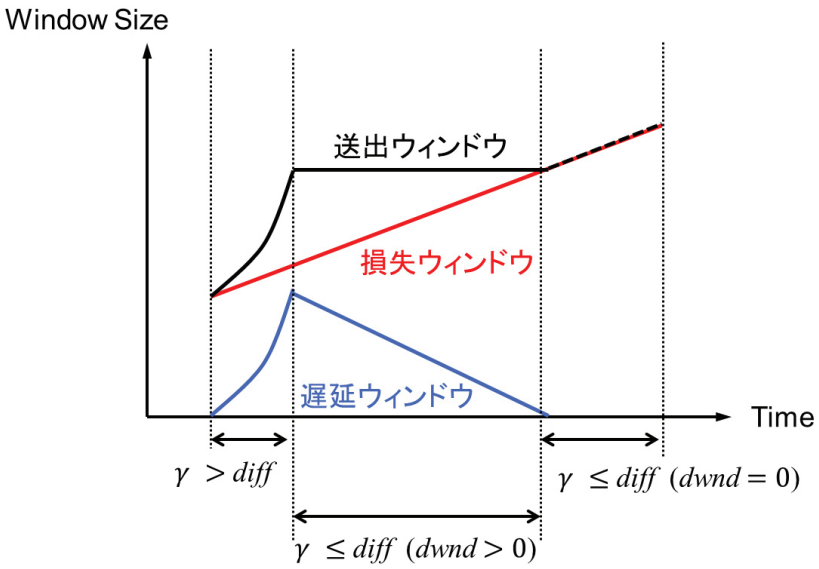


図 13: Compound TCP の輻輳ウィンドウの変化

リッド型の輻輳制御が提案されている．しかしながら，損失ベースの輻輳制御による改善手法は，TCP と競合した時その帯域を奪う公平性の問題が存在する．さらに遅延ベースの輻輳制御は，損失ベースの輻輳制御と競合した場合，帯域を奪われる公平性の問題が存在する．ハイブリッド型の輻輳制御は，損失ベースの輻輳制御と遅延ベースの輻輳制御が持つ問題を解決している．ハイブリッド型の輻輳制御の中でも，Compound TCP は Windows Vista SP1 以降に搭載され，Windows Server のような法人向けの OS ではデフォルトで有効に設定されている．そのため，Compound TCP は，今後広く用いられることが考えられる．そこで，本論文では Compound TCP を対象とする．

3　無線 LAN

本章では，まず無線 LAN 規格 IEEE 802.11 の概要を述べる．また，IEEE 802.11 において，衝突を回避するためのアクセス制御である CSMA/CA の制御を述べる．さらに，無線 LAN において，複数の TCP コネクションがデータ転送を行うとき，同一環境にも関わらず TCP コネクション間のスループット公平性が失われる問題が発生する．この問題の原因を述べた後，これまでに提案されている無線 LAN においてコネクション間のスループット公平性が失われる問題を解決する手法を述べる．

3.1　IEEE 802.11

IEEE 802.11 は，無線 LAN に関する規格であり，物理層における変調方式や MAC 層の通信制御を定めている．IEEE 802.11 は，1997 年に策定された規格である．IEEE 802.11 では，2.4 [GHz] 帯を用いた直接拡散方式，2.4 [GHz] 帯を用いた周波数ホッピング方式，赤外線通信方式 (IR) の 3 種類の方式が規定されている．2.4 [GHz] 帯は，ISM バンドと呼ばれる多目的な周波数帯である．そのため，2.4 [GHz] 帯を用いる方式では，他の電子機器や電子レンジといった多数のノイズが存在するため，干渉を受けやすい．そこで，2.4 [GHz] 帯の周波数を用いる方式では，対干渉性に優れたスペクトル拡散方式が採用されている．さらに，IEEE 802.11 では，MAC 層の制御として CSMA/CA による制御と，オプションとしてポーリングによる制御の 2 種類が規定されている．IEEE 802.11 では，伝送速度として 1 [Mbit/s] と，2 [Mbit/s] の 2 種類が規定されている．

IEEE 802.11 が策定された後，その次世代規格として IEEE 802.11b が 1999 年に策定された．IEEE 802.11b は 2.4[GHz] 帯の周波数帯域を用いる，最大伝送速度 11 [Mbit/s] の規格である．IEEE 802.11b は IEEE 802.11 との互換性を持つ．IEEE 802.11b は，IEEE 802.11 の直接拡散方式を改良した Complementary Code Keying(CCK) を採用する．IEEE 802.11 と同様に，2.4 [GHz] 帯の周波数を用いるため，IEEE 802.11b は他の電子機器と干渉を受けやすい．

また，IEEE 802.11b と同時期に，IEEE 802.11a が策定された．IEEE 802.11a は，5 [GHz] 帯の周波数帯域を用いる最大伝送速度 54 [Mbit/s] の規格である．IEEE 802.11a は，物理層において変調方式に，Orthogonal Frequency Division Multiplexing(ODFM) 方式を用いることで，IEEE 802.11b と比較して大幅に広帯域化した．IEEE 802.11a は，IEEE 802.11 や IEEE 802.11b との互換性は持たない．しかし，5 [GHz] 帯の周波数帯域を用いる IEEE 802.11a は，IEEE 802.11 や IEEE 802.11b と比較して他の電子機器による干渉は受けにくい．

IEEE 802.11b の策定後，2.4 [GHz] 帯の周波数帯域を用いる無線 LAN 規格をさらに高速化した IEEE 802.11g が策定された．IEEE 802.11g は，2003 年に策定された 2.4 [GHz] 帯の周波数帯域を用いる最大伝送速度 54 [Mbit/s] の規格である．IEEE 802.11g は，IEEE 802.11b との互換性を維持しつつ，IEEE 802.11a と同等の伝送速度を得ることができる．IEEE 802.11g の物理層では，IEEE 802.11b の CCK 方式と，IEEE 802.11a の ODFM 方式の 2 つの方式を必須とするように規定されている．また，IEEE 802.11b と IEEE 802.11a が混在する場合における互換性を維持するために，DSSS-OFDM 方式と，PBCC 方式をオプションとして規定されている．IEEE 802.11g は，2.4 [GHz] 帯の周波数帯域を用いるため，IEEE 802.11a と比較して，伝播の減衰が小さく，障害物に対して強いという利点を持つ．

IEEE 802.11 では，インフラストラクチャーモードとアドホックモードと呼ばれる 2 種類のネットワークを構成することができる．図 14 に，インフラストラクチャーモードを用いた時のネットワークの例を示す．インフラストラクチャーモードでは，アクセスポイントとアクセスポイントの通信範囲内に存在する無線端末に

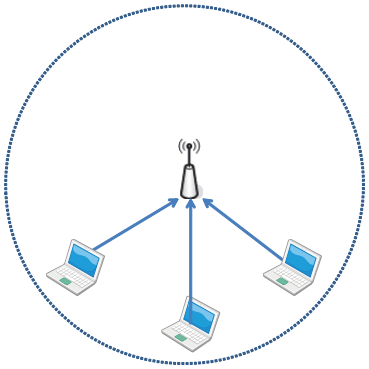


図 14: インフラストラクチャーモードによるネットワーク

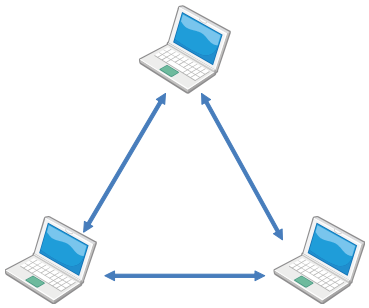


図 15: アドホックモードによるネットワーク

よってネットワークが構成される．無線端末が行う通信は，全てアクセスポイントを経由して行われる．

図 15 に，アドホックモードを用いた時のネットワークの例を示す．アドホックモードでは，インフラストラクチャーモードのようにアクセスポイントを必要としない．アドホックモードにおいて端末間で通信を行う場合は，端末同士が直接通信する．

3.2 Carrier Sense Multiple Access with Collision Avoidance

有線 LAN では，同一の回線を使う複数のユーザがアクセスする際の競合を避けるため，Carrier Sense Multiple Access with Collision Detection (CSMA/CD) と呼ばれるアクセス制御を行う．CSMA/CD では送信された信号を監視し，衝突による信号の変化によって衝突を検出する．しかし，無線 LAN においては送信信号がノイズの影響により大きく変化するため，CSMA/CD では信号の衝突を検出することができない．

そこで，無線 LAN 規格である IEEE 802.11 では，アクセス制御方式として Distributed Coordination Function (DCF) と，Point Coordination Function (PCF) という 2 つの方式が規定されている．DCF では，CSMA/CA と呼ばれるアクセス制御を使用することで，パケットの衝突を避ける．一方，PCF では，アクセスポイントが各無線ノードに対して，送信権を与える制御を行う．IEEE 802.11 では，DCF は必須である一方，PCF はオプションとなっている．そのため，本論文では，DCF をアクセス制御方式に用いる場合を対象とする．

IEEE 802.11 において，DCF を用いる場合，CSMA/CA によるアクセス制御が行われる．CSMA/CA によるアクセス制御では，まず，送信するデータを持つ無線端末は，他の無線端末が通信を行っているかを確認す

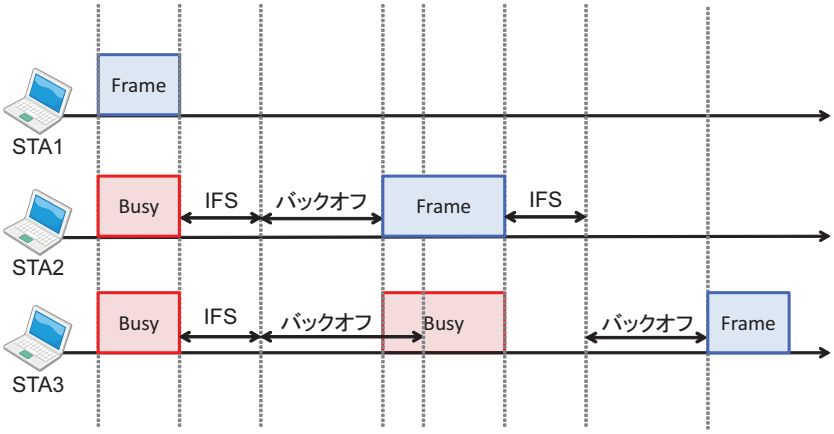


図 16: CSMA/CA での通信例

る．他の無線端末が通信を行っていないかを確認することをキャリアセンスと呼ぶ．一定時間キャリアセンスを行った結果，他の無線端末が通信を行っていない状態であると判断した場合は，無線端末はデータの送信を開始する．一方，キャリアセンスを行った際，他の無線端末の通信を検出した場合は，他の無線端末が行う通信が終了するまで，データの送信は行わず待機する．

図 16 に，CSMA/CA を用いた場合における各無線端末の通信の例を示す．STA2 および STA3 がデータを送信する時，既に STA1 が通信を行っている場合，STA2 および STA3 は，STA1 の通信が終わるまで通信を行うことができない．STA1 の通信が終了した時，Inter Frame Spacing(IFS) 時間だけ，STA2 および STA3 は待機する．STA2 および STA3 は，IFS 時間経過した後，ランダムなバックオフ時間だけキャリアセンスを行う．CSMA/CA では，キャリアセンスに加えて，バックオフ制御を行うことでパケットの衝突を回避する．このバックオフ時間が一番最初に終了した STA2 が STA1 の次にデータを送信することができる．STA3 はバックオフ時間の間に STA2 の通信が始まったため，STA2 の通信が終了するまでデータを送信することができない．

IFS には，Short IFS(SIFS)，PCF IFS(PIFS)，Distributed IFS(DIFS) の 3 種類がある．これらの IFS 時間は固定長であり，送信するフレームによって用いる IFS を使い分けることで，優先制御を行うことができる．図 17 に，各 IFS のフレーム間隔を表す．各 IFS は優先度によって各フレームの長さが異なり，フレームの長さが小さいほど優先度が高い．例えば，SIFS は，ACK フレームや CTS フレームの送信時に使用され，優先度は最高に設定されている．一方，DIFS は，データフレームの送信前に使用され，待ち時間は IFS 中で一番長く優先度は最低である．

既に述べたように，CSMA/CA では，パケットの衝突を回避するために，キャリアセンスに加えて，バックオフ制御が行われる．バックオフ制御は，IFS 時間待機した端末が Contention Window(CW) の範囲内で乱数を発生させ，その乱数値をもとにランダムなバックオフ時間が決定し，その時間だけキャリアセンスを行う．キャリアセンスをランダムな時間だけ行うことにより，各無線端末には送信する機会が公平に与えられる．バックオフ時間は，次式のように求められる．

$$backoff\_time = r \cdot slot\_time \tag{51}$$

r は，0 から CW までのランダムな整数値である．CW は，$CW_{min} \leq CW \leq CW_{max}$ の範囲内の整数である．このように，バックオフ時間は CW までのランダムな数値と slot time と呼ばれる規定の時間の倍数となる．しかし，他の無線端末とバックオフ時間が同一となった場合などにより衝突が発生した場合，再送が発生

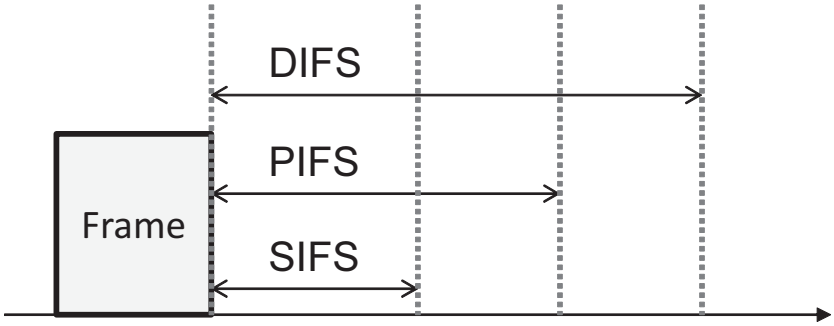


図 17: IFS による優先制御

するごとに次式のように CW の値を大きくする．

$$CW = (CW_{min} + 1) \cdot 2^n - 1 \tag{52}$$

n は 0 以上の再送回数を表す．再送が発生するごとに CW を指数関数的に増加させる．ただし，CW の上限値は CW_{max} である．CW が上限値である CW_{max} に達した場合において再送が発生した場合，最大再送回数 M に達するまで CW の値は CW_{max} とする．CW が CW_{max} となり，かつ，再送回数が最大再送回数である M に達した場合は，そのフレームを棄却する．

CSMA/CA では，キャリアセンスによって通信を行っている端末が存在するかを確認する．しかしながら，端末同士が障害物や電波の届かない位置にいる場合，キャリアセンスによって正しく他の端末の存在を確認することができない．この場合，データを送信する端末は，通信を行っている端末がいないと判断してデータの送信を行ってしまうため，衝突が発生する恐れがある．これを隠れ端末問題と呼ぶ．

隠れ端末問題を解決するために，IEEE 802.11 では RTS/CTS という制御がある．RTS/CTS は，Request To Send(RTS) と Clear To Send(CTS) の 2 つのフレームの送受信をアクセスポイントと行う．端末はまず RTS フレームをアクセスポイントに送信する．アクセスポイントが RTS を受信した時に，通信可能であれば，CTS を送信する．CTS は，RTS を受け取ったアクセスポイント配下のすべての端末に対して送信される．RTS を送信した端末は送信を開始し，それ以外の端末は待機する．アクセスポイント配下のすべての端末に CTS を送信することで，キャリアセンスが困難な端末でも無線回線の使用状況を知ることができるため，フレームの衝突を防ぐことが可能である．

3.3 無線 LAN における TCP の問題

無線 LAN 環境において，複数の TCP コネクションがデータ転送を行う時，同一環境下で通信を行っているにも関わらず TCP コネクション間のスループット公平性が失われる問題が発生する [32–34]．この問題は，無線 LAN で用いられるアクセス制御である CSMA/CA と TCP の輻輳制御に起因する．

CSMA/CA によるアクセス制御では，アクセスポイントを含む無線端末は，公平に送信する機会がある．ここで，多数の無線端末がアクセスポイントを経由して受信ホスト TCP を用いてデータ転送する場合を考える．1 つのアクセスポイントに対して接続される無線端末数を n とした時，パケットを送信する機会はすべての端末で平等であるため，各無線端末およびアクセスポイントが，送信機会を得る確率は $1/(n+1)$ となる．しかし，上下フロー単位で考えた場合，上りフローには n 台の無線端末が存在するので，上りフローが送信

機会を得る確率は $n/(n+1)$ となる．一方，下りフローは，無線端末はアクセスポイント 1 台のみであるため $1/(n+1)$ となる．このように，無線端末数の増加に伴って，下りフローが送信機会を得る確率が減少する．そのため，アクセスポイントは，無線端末からデータパケットを受信する回数に比べて，受信ホストからの ACK パケットを送信する回数が少なくなる．その結果，アクセスポイントにおいて送信できない ACK パケットがバッファリングされ続ける．この状態は，1 つのアクセスポイントに対して接続される無線端末数が減少しない限り続く．したがって，バッファに ACK パケットがバッファリングされ続けた結果，バッファが溢れが発生しバッファに入りきらない ACK パケットは破棄される．

TCP は，3 つの重複する ACK パケットを受信した場合，パケット棄却が発生したと判断し，高速再送を行う．高速再送は，再送タイマーのタイムアウトまで待たずに，棄却されたパケットを再送する制御である．アクセスポイントにおいてバッファ溢れが発生している場合，輻輳ウィンドウが小さいコネクションは，高速再送に必要な重複 ACK を受信することが困難になる．パケット棄却が発生し，かつ，十分な重複 ACK を受信することができなかった場合，そのコネクションはタイムアウトとなる．TCP は，タイムアウトが発生した場合，輻輳ウィンドウを 1 にまで減少させる．輻輳ウィンドウが非常に小さい値になることで，パケット棄却が発生した時，そのコネクションは，再びタイムアウトとなる可能性が高い．よって，1 度タイムアウトが発生したコネクションは，輻輳ウィンドウを増加させることができず，スループットは非常に低くなる．

一方，輻輳ウィンドウが大きいコネクションは，輻輳ウィンドウが小さいコネクションに比べて，パケット棄却が発生した場合でも高速再送となる確率が高い．その結果，ウィンドウサイズの差が大きくなり，スループットの差が大きくなる．このようにして，無線 LAN において TCP コネクション間のスループット公平性が失われる．

3.4 既存の無線 LAN におけるスループット公平性改善手法

無線 LAN において TCP を用いた場合，コネクション間のスループット公平性が失われる問題が発生する．その問題を改善する手法がこれまでに数多く提案されている．本章では，既存の無線 LAN におけるコネクション間のスループット公平性が失われる問題を解決する手法を説明する．これまでに提案されている無線 LAN におけるスループット公平性改善手法は，アクセスポイントに変更を加える手法と送信側ホストに変更を加える手法の 2 種類に分類される．以下では，それぞれについて説明する．

3.4.1 アクセスポイントに対して改良を加える手法

[35] では，アクセスポイントに対して改良を加えることで，無線 LAN におけるコネクション間のスループット公平性が失われる問題を解決する手法を提案している．具体的には，アクセスポイントの CW の最小値である CW_{min} を動的に変化させることで，無線 LAN におけるスループット公平性を維持する．IEEE 802.11 の CSMA/CA では，各無線端末には公平に送信する機会が与えられる．しかし，上下フロー間において与えられる送信機会は平等ではない．そのため，アクセスポイントのバッファが溢れるほど ACK パケットが蓄積し，ACK パケットが棄却されることが原因でコネクション間のスループット公平性が失われる．

IEEE 802.11 では，他の無線端末の通信終了した後 IFS 時間だけ待機する．その後，各無線端末はバックオフ時間の間キャリアセンスを行う．バックオフ時間は CW_{min} から CW_{max} までのランダムな値と slot time との倍数時間である．バックオフ時間経過後，他の無線端末が通信してない場合はパケットを送信することができる．したがって，バックオフ時間が短い端末ほど，他の無線端末よりも早くパケットを送信することができる．よって，アクセスポイントの CW_{min} を，他の無線端末の CW_{min} よりも小さくすることで，アクセス

ポイントのバックオフ時間が短くなり，アクセスポイントは他の無線端末よりも優先的にパケットを送信することができる．[35] では，M 本のフローが存在する時，CW_{min} を次式のように求める．

$$CW_{min} = \lfloor \frac{3}{2} + \frac{B}{M} + \sqrt{(1 + \frac{B}{M})^2 + \frac{2B}{M}} \rfloor \tag{53}$$

$\lfloor x \rfloor$ は，x を超えない最大の整数を表す．また，B は次式のように求める．

$$B = \frac{CW_{min}(CW_{min} - 2)}{2(CW_{min} + 1)} \tag{54}$$

[35] の手法はアクセスポイントにおいて下りフローの数を把握する必要がある．したがって，[35] の手法を導入するためにはアクセスポイントに対して，新たに改良を加える必要がある．これは，ネットワークの中で複雑な制御を行わない，End-to-End の原則に反する．さらに，アクセスポイントの制御機構は，ハードウェア上に実装されているため，既存のアクセスポイントに対して新たに修正を加えることは困難である．

次に，[36] では，アクセスポイントに対して改良を加えることで，無線 LAN におけるコネクション間のスループット公平性が失われる問題を解決する手法を提案している．具体的には，アクセスポイントに優先的に送信する機会を与えることで，スループット公平性が失われる問題を解決する．アクセスポイントに優先的に送信する機会を与えるために，アクセスポイントが用いる IFS に PIFS を用いるように変更する．アクセスポイントと通信を行う各無線端末の IFS は変更しない．PIFS は，DIFS よりも短いフレーム間隔である．したがって，アクセスポイントの IFS として，PIFS を用いることで，無線端末よりも早く送信する機会が得られる．これにより，無線 LAN 環境においてスループット公平性が失われる問題を解決する．

しかし，[36] の手法を導入するためにはアクセスポイントに対して変更を加える必要がある．上述したように，アクセスポイントにおいて新たな制御を組み込むことは困難である．従って，[36] の手法は現実的ではない．

また，[37] では，アクセスポイントに対して改良を加えることで，無線 LAN におけるコネクション間のスループット公平性が失われる問題を解決する手法を提案している．具体的には，アクセスポイントにおいて確率的にデータパケット棄却することでスループットの公平性を維持する．まずアクセスポイントにおいて，バッファ内の ACK パケットの占有率 (b) を計測する．b は次式のように求められる．

$$b = n/n_{max} \tag{55}$$

n はアクセスポイントにおいてバッファリングされているパケット数，n_{max} はアクセスポイントの最大バッファサイズを表す．アクセスポイントのバッファに n_{max} 個のパケットがバッファリングされている場合，次にキューイングされるパケットはオーバーフローとなる．

ACK パケットの占有率 b が，閾値 b_{th} を超えた場合，[37] の手法は上り TCP フローのデータパケットを任意の確率 p で棄却する．TCP は，データパケット棄却を検出した場合，輻輳ウィンドウを半減させる．そのため，アクセスポイントにおいてデータパケットを棄却することで，送信レートが減少する．それにより，アクセスポイントにおいてバッファ溢れを回避し，無線 LAN において TCP を用いた場合にスループットの公平性が失われる問題を改善する．

[37] の手法は，非常に実装が容易であり，さらに，IPSec のような暗号化されたパケットへの対応が可能である利点を持つ．しかしながら，データパケットの棄却する確率 p の適切な値は，ネットワークの状況によって変化すると考えられる．したがって p の適切に設定することは困難であると考えられる．

さらに，[38] では，アクセスポイントに対して改良を加えることで，無線 LAN におけるコネクション間のスループット公平性が失われる問題を解決する手法を提案している．具体的には IEEE 802.11e において，ア

クセスポイントの CW を動的に変化させることで，コネクション間のスループット公平性が失われる問題を解決する．IEEE 802.11e は，音声や動画などのリアルタイム型トラヒックや，ファイル転送などのベストエフォート型のトラヒックといった，トラヒックの種類に応じて優先制御を行うことにより，QoS 保証を実現する．しかし，IEEE 802.11e においても，コネクション間のスループット公平性が失われる問題が発生する．上述したように，アクセスポイントの CW を動的に変化させることで公平性を維持する手法は既に提案されている．しかし，IEEE 802.11e では，トラヒックの種類に応じて優先制御が行われていることが原因で，その手法を適用することができない．

そこで [38] では，[35] において提案された手法を IEEE 802.11e に適用する．まず，アクセスポイントにおいて優先度の低く設定されているベストエフォート型のフローの CW_{min} を [35] と同様の方法で求める．ベストエフォート型のフローの CW_{min} をもとに，優先度が高く設定されているリアルタイム型トラヒックの CW_{min} および CW_{max} を次式のように決定する．

$$CW_{min\_0} = \frac{CW_{min\_d} + 1}{4} - 1 \tag{56}$$

$$CW_{max\_0} = \frac{CW_{min\_d} + 1}{2} - 1 \tag{57}$$

$$CW_{min\_1} = \frac{CW_{min} + 1}{2} - 1 \tag{58}$$

$$CW_{max\_1} = CW_{min\_d} \tag{59}$$

CW_{min_d} は，ベストエフォート型のフローの CW_{min} である．さらに CW_{min_0} は，VoIP のトラヒックである場合の CW_{min}，CW_{max_0} は，VoIP のトラヒックである場合の CW_{max} である．また CW_{min_1} は，動画のトラヒックである場合の CW_{min}，CW_{max_1} は，動画のトラヒックである場合の CW_{max} である．

[38] の手法はアクセスポイントに優先権を与えることで，スループットの公平性が失われる問題を解決している．しかし，[38] の手法を導入するためにはアクセスポイントに対して変更を加える必要がある．既に述べたように，アクセスポイントにおいて新たな制御を組み込むことは困難である．従って，[38] の手法は現実的ではない．

これまでに多種多様な無線 LAN アクセスポイントが数多く販売されているにも関わらず，上記のように，無線 LAN においてコネクション間のスループット公平性が失われる問題を解決する手法が提案され続けている．そのため，現在販売されているアクセスポイントには，スループットの公平性を改善する手法が導入されていないと考えられる．実際に，BUFFALO 社製 WAPS-HP-AM54G54，NEC 社製 Aterm WR8500N，Corega 製 CG-WLR300NNH をアクセスポイントに用いた性能評価が行われており，いずれの製品においてもコネクション間のスループット公平性が失われる問題が確認されている [39]．また，本論文においては，後述するように BUFFALO 社製 WAPM-APG300N において性能評価を行っており，その結果スループット公平性が失われることを明らかにしている．このように，無線 LAN において公平性を改善する手法が採用されていない理由は，End-to-End の法則に反すること，また，適切なパラメータ設定が難しいことが考えられる．

3.4.2 送信側ホストに対して改良を加える手法

[39] では，送信ホストの制御を変更することにより，無線 LAN におけるコネクション間のスループット公平性が失われる問題を解決する手法が提案されている．具体的には，ACK パケットの棄却を輻輳の指標とする輻輳制御手法を提案している．無線 LAN においてスループット公平性が失われる問題が発生する時，アク

セスポイントにおいてバッファ溢れが発生し，大量の ACK パケットが棄却されており，ネットワークが輻輳状態となっている．TCP はデータパケットの棄却を検出した場合，ネットワークが輻輳状態であると判断して輻輳ウィンドウを半減させる．しかし，TCP は ACK パケットが棄却された場合でも，輻輳が発生していると判断しない．

[39] では，ACK パケットの棄却に対しても輻輳制御を行うことで，無線 LAN において，スループット公平性が失われる問題を解決する手法が提案されている．[39] の手法では，受信した ACK パケットのシーケンス番号を監視することで，ACK パケットの棄却を検出する．具体的には，受信した ACK パケットのシーケンス番号が 1 パケット分ずつ増加している場合は，ACK パケットの棄却は発生していないと判断する．受信した ACK パケットのシーケンス番号が 2 パケット分以上増加している場合は，ACK パケットが棄却されたと判断する．1 ラウンドトリップ時間以内に，ACK パケットの棄却を閾値である thresh ack losses 以上であれば，ネットワークが輻輳状態であると判断し，輻輳ウィンドウを半減させる．このように，[39] の手法は，送信ホストの制御を変更するのみで導入することが可能である．したがって，アクセスポイントに変更を加える手法とは異なり，End-to-End の原則に反しない．[39] の手法では，ACK パケットの棄却をデータパケットの棄却と同様に扱う．そのため，ACK パケットの棄却した場合，輻輳ウィンドウが大きく減少し，その結果スループットが大きく低下する．

既に述べたように，無線 LAN においてスループットの公平性を維持するために，アクセスポイントにおいて新たな制御を組み込むことは困難である．また，アクセスポイントに公平性改善手法を導入することは，End-to-End の原則に反するため，望ましくない．そこで，本論文では，[39] と同様に，エンド端末間の制御を変更することにより，無線 LAN におけるコネクション間のスループット公平性が失われる問題を解決する手法を提案する．

4　Compound TCP の輻輳制御と無線 LAN における性能評価

本章では，まず，広帯域高遅延なネットワーク向けに提案された Compound TCP の輻輳制御を説明する．その後，無線 LAN における Compound TCP の性能評価を行う．Compound TCP は TCP と同一の輻輳制御を行う損失ベースの輻輳制御と，ラウンドトリップ時間を輻輳の指標とする遅延ベースの輻輳制御を組み合わせた輻輳制御を行う．この 2 つの輻輳制御を組み合わせることで，広帯域高遅延なネットワークにおいて，Compound TCP は帯域を有効に活用することができる．しかし，Compound TCP は，TCP と同一の輻輳制御を含むため，無線 LAN において TCP と同様の問題が発生する恐れがある．そこで，無線 LAN における Compound TCP の性能評価を行いシミュレーションにより行う．その結果，Compound TCP を無線 LAN において用いた場合，TCP と同様の問題が発生することを示す．

4.1　Compound TCP の輻輳制御

Compound TCP は，パケット棄却を輻輳の指標とする，損失ベースの輻輳制御と，ラウンドトリップ時間を輻輳の指標とする，遅延ベースの輻輳制御を組み合わせた，輻輳制御を行う．本節では，Compound TCP の輻輳制御を説明する．

1 ラウンドトリップ時間にネットワークに送出するパケット数を表す送出ウィンドウ $swnd$ を，Compound TCP は次式から求める．

$$swnd = cwnd + dwnd \tag{60}$$

ここで，$cwnd$ は，損失ベースの輻輳制御で用いられる損失ウィンドウ，そして $dwnd$ は，遅延ベースの輻輳制御で用いられる遅延ウィンドウである．

Compound TCP は，損失ベースの輻輳制御と，遅延ベースの輻輳制御をそれぞれ独立に同時に動作させ，それぞれの制御においてウィンドウを決定する．そして，Compound TCP は，損失ウィンドウと遅延ウィンドウの和をもって，パケットを実際に送出する際に用いるウィンドウを決定する．以下では，損失ベースの輻輳制御と，遅延ベースの輻輳制御をそれぞれ説明する．

まず，損失ベースの輻輳制御は，TCP と同様にスロースタートフェーズと，輻輳回避フェーズの 2 つのフェーズから構成されている．まず，通信開始時，Compound TCP の損失ベースの輻輳制御はスロースタートフェーズとなる．スロースタートフェーズでは，損失ウィンドウを急速に増加させ帯域遅延積を早期に推定する．スロースタートフェーズにおいてパケット棄却を検出した時，Compound TCP は輻輳回避フェーズに移行する．輻輳回避フェーズでは，輻輳が発生を回避するように損失ウィンドウを増加させる．具体的には，損失ベースの輻輳制御は各動作フェーズにおいて，ACK パケットを受信するごとに損失ウィンドウを次式に従い決定する．

$$cwnd = \begin{cases} cwnd + 1 \\ \quad (slow\ start\ phase) \\ cwnd + \dfrac{1}{swnd} \\ \quad (congestion\ avoidance\ phase) \end{cases} \tag{61}$$

Compound TCP の損失ベースの輻輳制御は，スロースタートフェーズでは，損失ウィンドウを指数的に増加させる．また，輻輳回避フェーズでは，損失ウィンドウを線形的に増加させる．

また，パケット棄却を検出した場合，Compound TCP は，次式のように損失ウィンドウを減少させる．

$$cwnd = \begin{cases} \dfrac{cwnd}{2} & (duplicateACKpackets) \\ 1. & (timeout) \end{cases} \tag{62}$$

Compound TCP は，重複 ACK の受信によりパケット棄却を検出した時，ネットワークに軽度の輻輳が発生したと判断して，損失ウィンドウを半減する．また，タイムアウトによりパケット棄却を検出した時，ネットワークに重度の輻輳が発生したと判断し，Compound TCP は，損失ウィンドウを 1 に減少させる．

遅延ベースの輻輳制御は，損失ベースの輻輳制御と同様に，スロースタートフェーズと輻輳回避フェーズのそれぞれのフェーズによって動作が異なる．遅延ベースの輻輳制御は，スロースタートフェーズ時において，遅延ウィンドウの増減を行わない．輻輳回避フェーズ時における，遅延ウィンドウの増減方法は，以下の手順で計算される．まず，Compound TCP は，ルータのバッファ内パケット数 *Diff* を，次式から求める．

$$\textit{Diff} = \left(\frac{swnd}{baseRTT} - \frac{swnd}{RTT} \right) \cdot baseRTT \tag{63}$$

ただし，$baseRTT$ は，ラウンドトリップ時間の最小値，RTT は，現在のラウンドトリップ時間である．$swnd/baseRTT$ から，ネットワークが輻輳していない理想状態であると考えられる時のスループットが求められる．また，$swnd/RTT$ から，現在のスループットが求められる．理想状態である時のスループットから現在のスループットの差に $baseRTT$ を掛けることで，ルータのバッファ内パケット数を求められる．

Diff を用いて，Compound TCP は 1 ラウンドトリップ時間毎に遅延ウィンドウを次式から求める．

$$dwnd = \begin{cases} dwnd + (\alpha \cdot swnd^k - 1)^+ & (\textit{Diff} < \gamma) \quad (64) \\ (dwnd - \zeta \cdot \textit{Diff})^+ & (\textit{Diff} \geq \gamma) \quad (65) \end{cases}$$

ただし，α，β，k, ζ は，Compound TCP の制御パラメータであり，γ は，ネットワークの輻輳状態を判断する閾値である．*Diff* が γ より小さい場合，ルータのバッファに空きがあり，さらに多くのパケットをネットワークに対して送出することができるような，空き帯域がある状態であると考えられる．そこで，遅延ベースの輻輳制御は，遅延ウィンドウを増加させる．一方，*Diff* が γ を超える場合，パケットがルータのバッファ内に多数存在する，ネットワークが軽度の輻輳状態であると考えられる．そこで，遅延ベースの輻輳制御は，遅延ウィンドウを減少させる．

パケット棄却を検出した時，遅延ウィンドウは次式で決定される．

$$dwnd = \begin{cases} (swnd \times (1 - \beta) - cwnd/2)^+ & (duplicateACKpackets) \\ 0 & (timeout) \end{cases} \tag{66}$$

重複 ACK の受信によりパケット棄却を検出した時，ネットワークに軽度の輻輳が発生したと判断する．この時，Compound TCP は，遅延ウィンドウを約 $\beta dwnd$ (ただし $0 < \beta < 1$) だけ減少させる．また，タイムアウトによりパケット棄却を検出した時，ネットワークに重度の輻輳が発生したと判断し，Compound TCP の遅延ベースの輻輳制御は，遅延ウィンドウを 0 とする．

このように，Compound TCP は，損失ベースの輻輳制御と遅延ベースの輻輳制御が独立に動作し，組み合わせることで広帯域なネットワークにおいて帯域を有効に活用することができる．しかし，Compound TCP に

表 1　Compound TCP の性能評価に用いるシミュレーションモデルの設定

ネットワーク環境	
無線端末数	2-20
往復伝搬遅延	10[ms], 100[ms], 220 [ms]
有線帯域幅	10 [Gbit/s]
パケットサイズ	1500 [byte]
無線 LAN パラメータ	
無線 LAN 規格	IEEE 802.11g
Slot time	9 [μs]
SIFS	16 [μs]
DIFS	34 [μs]
CW_{min}	15
CW_{max}	1023
Data rate	54 [Mbit/s]

含まれる損失ベースの輻輳制御は，TCP と同一の制御を行う．そのため，無線 LAN において Compound TCP を用いた場合，TCP と同様の問題が発生する恐れがある．Compound TCP は，Windows Vista SP1 以降に搭載されており，今後広く用いられることが考えられる．また，近年，無線 LAN の普及が進んでいる．そのため，今後 Compound TCP が有線環境だけではなく，無線環境において広く用いられることが考えられる．そのため，無線 LAN における Compound TCP の評価を行い，TCP と同様の問題があるかを明らかにする必要がある．

4.2　シミュレーションによる Compound TCP の性能評価

無線 LAN における Compound TCP の性能評価をシミュレーションにより行う．シミュレーションは，ns-2 [44] を用いて，パケットレベルで行った．図 18 にシミュレーションモデル示す．無線端末側を送信側とする．無線 LAN 規格は，IEEE 802.11g とし，パラメータは Slot Time は 9[μs]，SIFS は 16 [μs]，DIFS は 34 [μs]，CW_{min} は 15，CW_{max} は 1023，Data Rate は 54 [Mbit/s] に設定した．アクセスポイントと受信端末間の帯域は 10 [Gbit/s]，遅延は 5 [ms]，50[ms]，110 [ms] とした．さらに，無線端末数は 2 から 20 端末，パケットサイズは 1500 [Byte]，アクセスポイントのバッファサイズは 100 [packet] とした．また，Compound TCP のパラメータ α, β, k, ζ, および γ は，それぞれ 1/8, 1/2, 1, 0.8, 30 に設定した．これらのパラメータは，Compound TCP を提案した論文 [45] で用いられている値である．これらの条件で 500 秒間のシミュレーションを行った．シミュレーションで用いるネットワーク環境およびパラメータ設定を表 1 にまとめ，Compound TCP のパラメータ設定を表 2 にまとめる．

図 19 に，無線端末が 20 台存在する場合における，アクセスポイントと受信端末間の遅延を 110 [ms] とした時のスループットの平均値と，全帯域を全ての無線端末で均等に分け与えた fair share を示す．なお，図はスループットが高い順にソートしている．また，無線端末が 2 台から 19 台存在する場合と，アクセスポイントと受信端末間の遅延が 5 [ms] または 50 [ms] である場合のシミュレーション結果は，付録 A に記載する．図 19 から，無線 LAN において Compound TCP を用いた場合，fair share よりもスループットが高いコネク

表 2　Compound TCP の性能評価に用いるパラメータ設定

α	1/8
β	1/2
ζ	1
k	0.8
γ	30

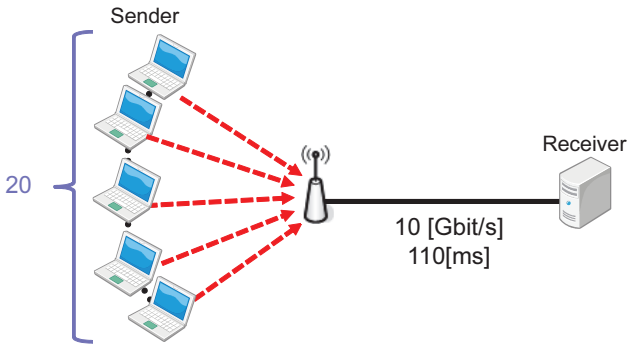


図 18: Compound TCP の性能評価に用いるシミュレーションモデル

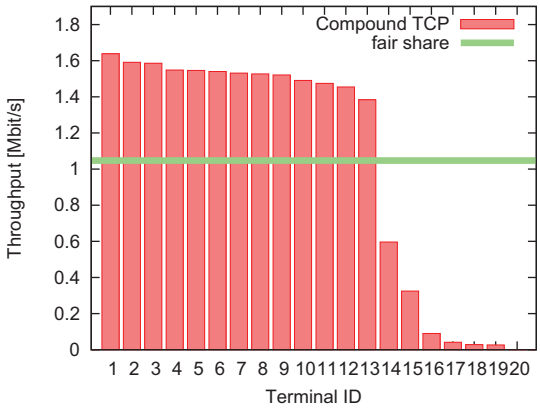


図 19: 遅延 110[ms] 時において無線端末が 20 台存在する時のスループット

ションが存在すること，その反対にスループットが非常に低いコネクションが存在することがわかる．このことから，TCP と同様に，Compound TCP においても，コネクション間のスループットが不公平になる問題が発生することが分かる．

このように，無線 LAN において Compound TCP を用いた場合，アクセスポイントと受信端末間のネットワーク環境に関わらず，無線端末の増加に伴ってスループットの公平性が失われることを示した．その原因を示すために，図 20 に，図 19 で最もスループットの高いコネクションの損失ウィンドウ，遅延ウィンドウ，送出ウィンドウサイズを示す．図 20 から，遅延ベースの輻輳制御によって増減する，遅延ウィンドウに変化はないことがわかる．また，送出ウィンドウは，損失ベースの輻輳制御で増減する，損失ウィンドウのみで決定

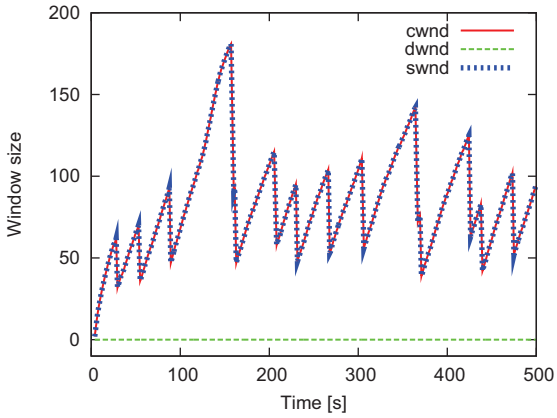


図 20　端末数 20，遅延 110 [ms] 時における Compound TCP の損失，遅延，送出ウィンドウ

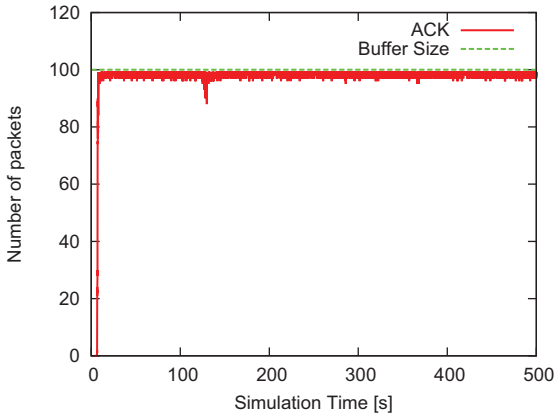


図 21　端末数 20 時におけるアクセスポイントのバッファ内パケット数

されていることがわかる．これは，無線 LAN において Compound TCP の輻輳制御は，TCP と同一の輻輳制御となることを意味している．

図 21 に，送信ホストとなる無線端末が 20 台存在する場合の，アクセスポイントのバッファ内パケット数を示す．図 21 から，アクセスポイントのバッファが常に埋められていることが分かる．アクセスポイントに大量のパケットが蓄積したことによって，RTT が増加する．遅延ベースの輻輳制御は，このラウンドトリップ時間の増加を，ネットワークの輻輳状態であると判断し，遅延ウィンドウを減少し続ける．従って，図 20 のように，遅延ウィンドウが常に 0 となる．一方，損失ベースの輻輳制御は，ラウンドトリップ時間の増加に関わらず，パケット棄却を検出するまで損失ウィンドウを増加させる．従って，無線 LAN において Compound TCP を用いた場合，TCP と同一の輻輳制御制御となる．

5　Compound TCP+ の輻輳制御と無線 LAN における性能評価

本章では，無線 LAN における Compound TCP の問題を解決する，Compound TCP+ の輻輳制御を提案する．Compound TCP+ は，軽度の輻輳状態と考えられる場合，Compound TCP とは異なり，損失ウィンドウを減少させる．これにより無線 LAN において，アクセスポイントのバッファ溢れを回避し，コネクション間のスループット公平性が失われる問題を解決する．また，シミュレーションおよび，実験ネットワークにおける Compound TCP+ の性能評価を行う．その結果，Compound TCP+ が，無線 LAN において高い公平性を得ることを示す．

5.1　Compound TCP+ の輻輳制御

Compound TCP+ は，軽度の輻輳を検出した時に，遅延ベースの輻輳制御が損失ウィンドウの動作を，Compound TCP から変更することによって，無線 LAN において公平性が失われる問題を解決する．ここでは，遅延ベースの輻輳制御で用いられる遅延ウィンドウが 0 である場合を軽度の輻輳とする．ルータのバッファ内にパケットが閾値以上存在する場合，遅延ベースの輻輳制御はネットワークが輻輳状態であると判断し，遅延ウィンドウを減少させる．この輻輳状態が継続した場合，遅延ウィンドウは最終的に 0 となる．また，パケット棄却が発生する場合は，ルータのバッファがあふれる深刻な輻輳状態である．よって，遅延ウィンドウが 0 となる時，遅延ベースの輻輳制御はネットワークが輻輳状態であると判断し続けているが，パケット棄却が発生するような深刻な輻輳状態ではない．よって，このような輻輳状態を軽度の輻輳状態であるとした．しかし，Compound TCP では，バッファがあふれパケットが棄却されない限り，損失ウィンドウを増加させ続ける．その結果，輻輳の悪化が進み，無線 LAN においてはコネクション間のスループット公平性が失われる．

そこで，Compound TCP+ では，遅延ウィンドウが 0 となるような，ネットワークが軽度の輻輳状態である場合において，たとえ，適切に ACK パケットを受信していようとも，損失ウィンドウを増加させない．これにより，Compound TCP+ は，アクセスポイントのバッファ溢れを防ぎ，コネクション間のスループット公平性を維持する．本論文では，遅延ウィンドウが 0 となる場合，損失ウィンドウを以下の 3 つの式で与えられる場合を検討する．

$$cwnd = \lceil cwnd * a \rceil \tag{67}$$
$$cwnd = \lceil cwnd - b \rceil \tag{68}$$
$$cwnd = cwnd \tag{69}$$

ここで，$\lceil x \rceil$ は，x 以上の最小の整数である．$dwnd = 0$ である場合，ネットワークは軽度の輻輳状態であると考えられる．そこで，まず損失ウィンドウを乗算的に減少 (式 (67)) する，損失ウィンドウを大きく減少させる場合を検討する．軽度の輻輳状態において，損失ウィンドウを大きく減少させることにより，無線 LAN においてはアクセスポイントのバッファ溢れを防ぎ，スループット公平性を維持することができると考えられる．しかし，損失ウィンドウを乗算的に減少した場合，スループットが大きく減少することが考えられる．そこで，乗算的な減少よりも軽度な減少として，損失ウィンドウを線形的に減少 (式 (68)) させる場合を検討する．さらに，軽度な輻輳状態において，損失ウィンドウを減少させる必要性を検討するため，損失ウィンドウを一定値 (式 (69)) とする場合を検討する．

一方，ネットワークに空き帯域があると考えられる場合や，パケット棄却が発生する重度の輻輳状態では，

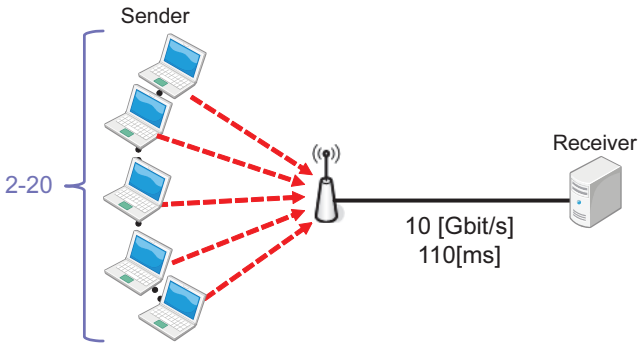


図 22　Compound TCP+ の性能評価に用いるシミュレーションモデル

Compound TCP+ は Compound TCP と同様の輻輳制御を行う．まず．Compound TCP+ の損失ベースの輻輳制御は，ACK パケットを受信するごとに，損失ウィンドウを式 (61) に従い決定する．また，パケット棄却を検出した場合は，式 (62) に従い，損失ウィンドウを減少させる．また，Compound TCP+ の遅延ベースの輻輳制御は，スロースタートフェーズでは遅延ウィンドウの増減を行わない．輻輳回避フェーズでは，ルータのバッファ内パケット数の推測値 *Diff* を，式 (63) から求める．その後，遅延ウィンドウを式 (64)，式 (65) から求める．また，パケット棄却を検出した場合は，式 (66) のように遅延ウィンドウを減少させる．

5.2　シミュレーションによる Compound TCP+ の性能評価

ネットワークが軽度の輻輳状態における Compound TCP+ の輻輳制御方式の検討を，シミュレーションにより行う．シミュレーションは ns-2 [44] を用いた．シミュレーションに用いたネットワークモデルを，図 22 に示す．図 22 のネットワークにおいて，500 秒間のシミュレーションを 10 回行い，最初の 100 秒を除いた，後半 400 秒のシミュレーション結果の平均値を用いて評価する．

本論文では，公平性の指標として，Fairness Index [46] を用いる．n をサンプル数，$x_i(1 \leq i \leq n)$ を n 個のサンプル値とするとき，n 個のサンプル値の公平性を表す Fairness Index f は，以下の式で与えられる

$$f = \frac{(\sum_{n=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2} \qquad (1 \leq i \leq n)$$

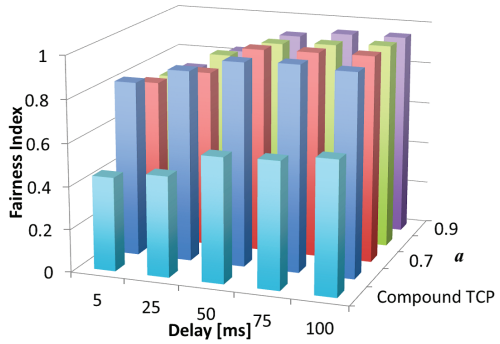
Fairness Index f は，0 以上 1 以下の値を取る．また，Fairness Index f は，1 に近い値ほど公平性が高い．

以下では，まず，Compound TCP+ のパラメータ検討を行う．その後，無線 LAN における Compound TCP+ と既存のスループット公平性改善手法 [39] を適用した Compound TCP，および Compound TCP との比較評価を行う．

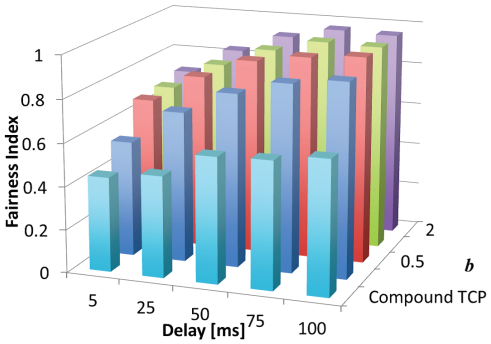
5.2.1　Compound TCP+ のパラメータ検討

Compound TCP+ のパラメータを検討する．Compound TCP+ は，遅延ウィンドウが 0 であるような，ネットワークが軽度の輻輳状態である場合において，損失ウィンドウを増加させない輻輳制御を行う．ここでは，ネットワークが軽度の輻輳状態である場合に，式 (67) で損失ウィンドウを乗算的に減少させる場合のパラメータ a について検討する．さらに，式 (68) で損失ウィンドウを線形的に減少させる場合のパラメータ b について検討する．

図 23 に，アクセスポイントと受信端末間の遅延，および a または b が変化したときの Compound TCP お

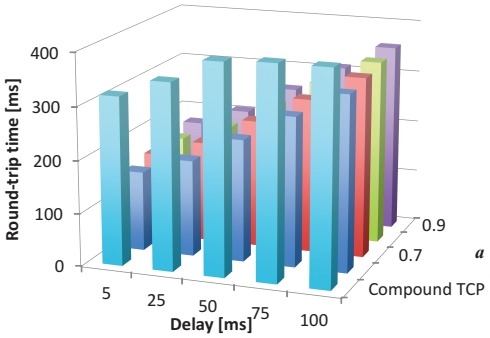


(a) 乗算的な減少

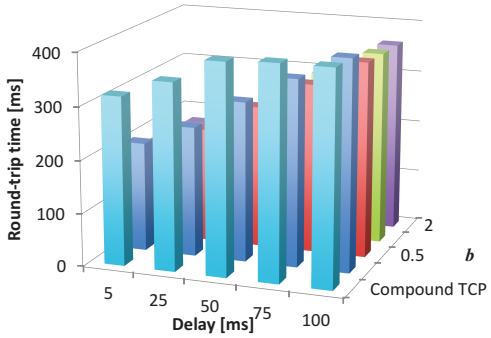


(b) 線形的な減少

図 23　Compound TCP+ のパラメータ変化に対する Fairness index



(a) 乗算的な減少



(b) 線形的な減少

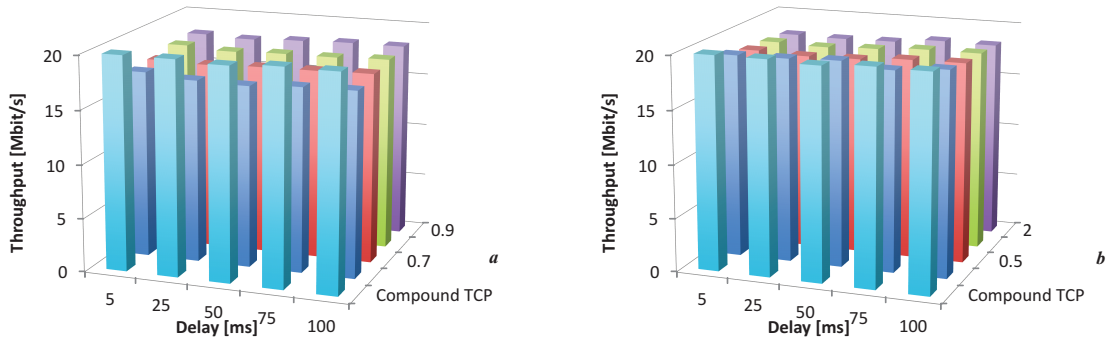
図 24　Compound TCP+ のパラメータ変化に対するラウンドトリップ時間

よび，Compound TCP+ の Fairness Index を示す．図 23 から，a が小さくなるほど，または，b が大きくなるほど，僅かに公平性が高まることがわかった．また，乗算的に減少する輻輳制御方式の方が，線形的に減少または一定に維持する輻輳制御方式より公平性が高まることがわかった．これは，ネットワークが軽度の輻輳状態である場合における，損失ウィンドウの減少幅が大きくなるほど，アクセスポイントのバッファに空きが生まれ，その結果，公平性が高くなる．

さらに，アクセスポイントと受信端末間の遅延が小さくなるほど，公平性が低下していることがわかる．これは，遅延の小さいネットワークは，遅延の大きいネットワークに比べて，帯域遅延積が小さく，送出ウィンドウが小さくなる．そのため，タイムアウトが発生する確率が高まり，その結果，公平性が低下する．

図 24 に，アクセスポイントと受信端末間の遅延，および a または b が変化したときの Compound TCP および，Compound TCP+ のラウンドトリップ時間を示す．図 24 から，a が大きくなるほど，または，b が小さくなるほど，僅かに，ラウンドトリップ時間が小さくなることがわかった．また，乗算的に損失ウィンドウを減少する場合の方が，線形的に損失ウィンドウを減少させる場合に比べて，ラウンドトリップ時間が小さくなることがわかった．これは，損失ウィンドウの減少幅が大きいほど，ルータのバッファ内パケット数が少ないため，平均ラウンドトリップ時間は小さくなる．

図 25 に，アクセスポイントと受信端末間の遅延，および a または b が変化したときの Compound TCP および，Compound TCP+ の合計スループットを示す．図 25 から，a が大きくなるほど，または，b が小さくな



(a) 乗算的な減少　　(b) 線形的な減少

図 25　Compound TCP+ のパラメータ変化に対する合計スループット

るほど，僅かに，合計スループットが大きくなることがわかった．また，損失ウィンドウを線型的に減少させる場合，損失ウィンドウを乗算的に減少させる場合と比較して，合計スループットが高いことがわかった．これは，線形的に損失ウィンドウを減少させる場合の方が，損失ウィンドウの減少幅は小さいためである．さらに，アクセスポイントと受信端末間の遅延が変化によって，合計スループットに大きな変化がないことがわかる．これは，Compound TCP+ は，ルータのバッファ内パケット数が γ となるように，送信レートを調節するためである．

これらの結果から，Compound TCP+ は，損失ウィンドウを乗算的に減少する輻輳制御を行うとき，パラメータ a に設定する値を小さくするほど，高い公平性を得ること，また，ラウンドトリップ時間が小さくなることがわかった．しかし，パラメータ a 設定する値に関わらず，Compound TCP に比べて，合計スループットが低下する．また，損失ウィンドウを一定に維持する輻輳制御を行うことで，合計スループットの低下は改善するが，ラウンドトリップ時間が大きく，公平性が低下することがわかった．さらに，損失ウィンドウを線形的に減少する輻輳制御を行うとき，パラメータ b に設定する値を大きくするほど，Compound TCP に比べて公平性を改善すること，さらに，パラメータ b に設定する値が小さくなるほど，合計スループットの低下を抑えることがわかった．そのため，総合的に見ると，軽度の輻輳状態となる時，損失ウィンドウを線形的に減少させ，そのパラメータには，スループットの低下を最も抑える 0.5 が最適であると考えられる．

5.2.2　Compound TCP+ と従来手法との比較評価

次に，Compound TCP+ と従来手法との性能評価を行う．図 26 に，端末数が変化した場合の Compound TCP，および，[39] の手法を適用した Compound TCP，Compound TCP+ の (a) Fairness Index，(b) 平均ラウンドトリップ時間，そして，(c) 合計スループットを，それぞれ示す．ここでは，[39] の手法における閾値 thresh_ack_losses は，[39] で用いられている値である 1 に設定している．

図 26(a) から，Compound TCP は，端末数の増加に伴って，公平性が低下することがわかる．それに対して，[39] の手法を適用した Compound TCP は，端末数に関わらず高い公平性を持つことがわかる．[39] の手法は，1RTT 中に閾値以上の ACK パケットの棄却を検出した場合，送出ウィンドウを半減する．その結果，アクセスポイントのバッファ溢れを防ぎ，高い公平性を持つことができたと考えられる．また，Compound TCP+ は，高い公平性を得ることがわかる．これは，ネットワークが軽度の輻輳状態である時に，損失ウィンドウが減少したことでアクセスポイントにおけるバッファを防ぐ．その結果，Compound TCP+ は高い公平性

を達成する．

図 26(b) から，Compound TCP の平均ラウンドトリップ時間は，非常に高いことがわかる．これは，Compound TCP が，パケット棄却を検出するまで，損失ウィンドウを増加させ続ける輻輳制御を行うためである．送出ウィンドウが増加するほど，アクセスポイントのバッファには，多くのパケットが蓄積する．その結果，Compound TCP の平均ラウンドトリップ時間は，非常に大きくなる．それに対して，Compound TCP+，および，[39] の手法を適用した Compound TCP は，Compound TCP と比較して，ラウンドトリップ時間は小さいことがわかる．これは，Compound TCP+ や [39] の手法は，アクセスポイントのバッファ溢れを回避する輻輳制御を行う．そのため，バッファを埋め尽くす輻輳制御を行う Compound TCP と比較してラウンドトリップ時間は小さくなる．ラウンドトリップ時間が非常に大きいことは，単なるファイル転送のアプリケーションでは，問題とならないが，インタラクティブなアプリケーションでは，非常に問題となる．

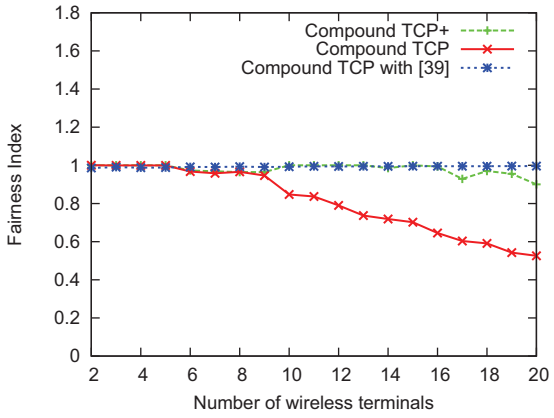
図 26(c) から，[39] の手法を適用した Compound TCP，および，Compound TCP+ の合計スループットは，Compound TCP の合計スループットと比較して，低いことがわかる．Compound TCP は，パケット棄却が発生するまで，損失ウィンドウを増加させる輻輳制御を行う．それに対して，[39] の手法を適用した Compound TCP+ は，軽度の輻輳状態である場合，損失ウィンドウを減少させる．そのため，Compound TCP+ の合計スループットが，Compound TCP の合計スループットと比較して，低くなったと考えられる．しかし，[39] の手法を適用した Compound TCP と比較して Compound TCP+ のスループットは高いことがわかる．また，無線端末数が少ない時，CompoundTCP+ の合計スループットが，一定となっていることがわかる．これは，Compound TCP+ は，ルータのバッファ内パケット数が，γ 以下となるように，送信レートを制御するためであると考えられる．

以上の結果から, 無線 LAN において Compound TCP+ は, 高い公平性を得ることを示した．また, Compound TCP+ は，既存の無線 LAN におけるスループット公平性改善手法と比較して高いスループットを得ることを示し，その有効性を示した．

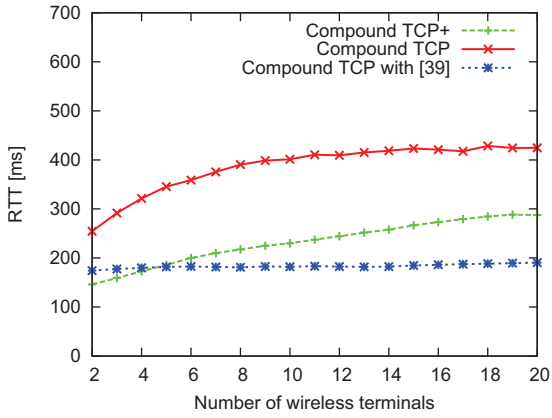
5.3 実験ネットワークにおける Compound TCP+ の性能評価

Compound TCP+ を Linux カーネル 3.0.0 上に実装し，実験ネットワークにおける Compound TCP+ を性能評価する．図 27 に，実験ネットワークを示す．実験ネットワークは，送信側ホストとなる 2 から 10 台の無線端末，1 台のアクセスポイント，Dummynet がインストールされた計算機，1 台の受信ホストから構成される．アクセスポイントと受信ホスト間は，100 [Mbit/s] の Ethernet で接続されている．また，アクセスポイントと受信ホスト間は，Dummynet を用いて 60 [ms] の遅延を発生させている．無線 LAN 規格は IEEE 802.11a を用いた．アクセスポイントには，BUFFALO 社製 WAPM-APG300N，また，無線端末の無線 LAN カードには，BUFFALO 社製 WLI-CB-HGHP を使用した．実験では，各無線端末が受信ホストに対して，同時にトラヒックを発生させる．トラヒック生成には `Iperf` を用いた．本論文では，1 回 150 秒間の計測を 20 回行い，実験開始 60 秒後から 60 秒間の結果を用いる．Compound TCP+ は，γ を 30 とし，遅延ウィンドウが 0 である場合，損失ウィンドウを式 (68) で決定し，そのパラメータ b は，$1/2$ とする．また，比較のために Compound TCP を用いた場合の実験も行う．表 3 に，実験ネットワークの設定をまとめる．

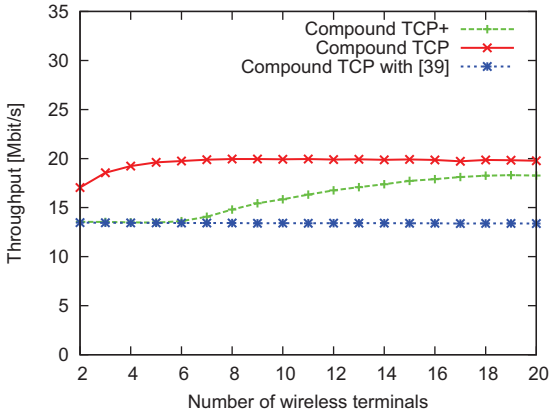
図 28 に端末数が変化に対する Fairness Index を示す．図 28 から，Compound TCP の Fairness Index は，無線端末数の増加に伴い徐々に低下しており，公平性が悪化することがわかる．それに対して，Compound TCP+ は，無線端末が増加した場合でも，Fairness Index は低下せず，公平性を保った状態で通信することが可能であることがわかる．Compound TCP は，アクセスポイントにおいて，バッファリングされているパケッ



(a) Compound TCP+ および他手法の Fairness Index



(b) Compound TCP+ および他手法のラウンドトリップ時間



(c) Compound TCP+ および他手法の合計スループット

図 26: Compound TCP+ および他手法とのシミュレーション結果の比較

表 3　Compound TCP+ の性能評価に用いる実験ネットワークの設定

無線端末数	2–10
有線帯域	100 [Mbit/s]
往復伝搬遅延	120 [ms]
無線 LAN 規格	IEEE 802.11a
無線アクセスポイント	BUFFALO WAPM-APG300N
無線 LAN カード	BUFFALO WLI-CB-HGHP
γ	30
b	1/2

ト数に関わらず，損失ウィンドウを増加させる．そのため，アクセスポイントのバッファを埋め尽くし，公平性が低下する．一方，Compound TCP+ は，$dwnd = 0$ となるネットワークが軽度の輻輳状態である時，損失ウィンドウを減少させる．これにより，アクセスポイントにおいてバッファ溢れを防ぎ，Compound TCP は高い公平性を得ることができる．

また，図 29 に，無線端末が 10 台存在する場合の Compound TCP および Compound TCP+ を用いた場合に

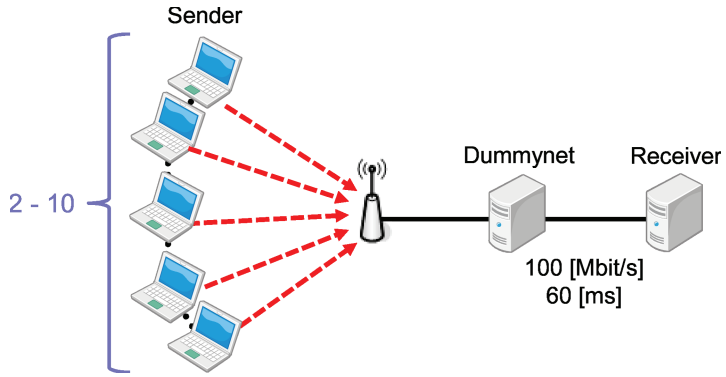


図 27: Compound TCP+ の性能評価に用いる実験ネットワークモデル

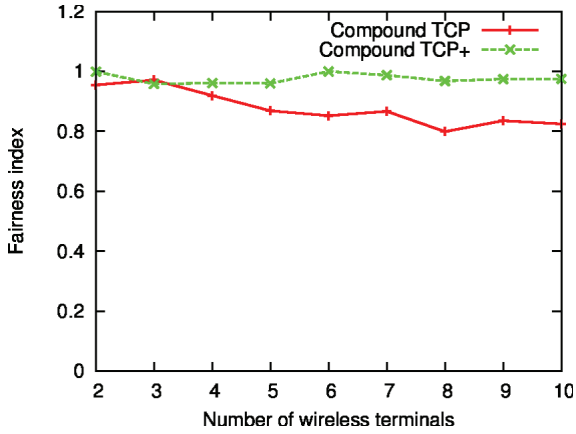
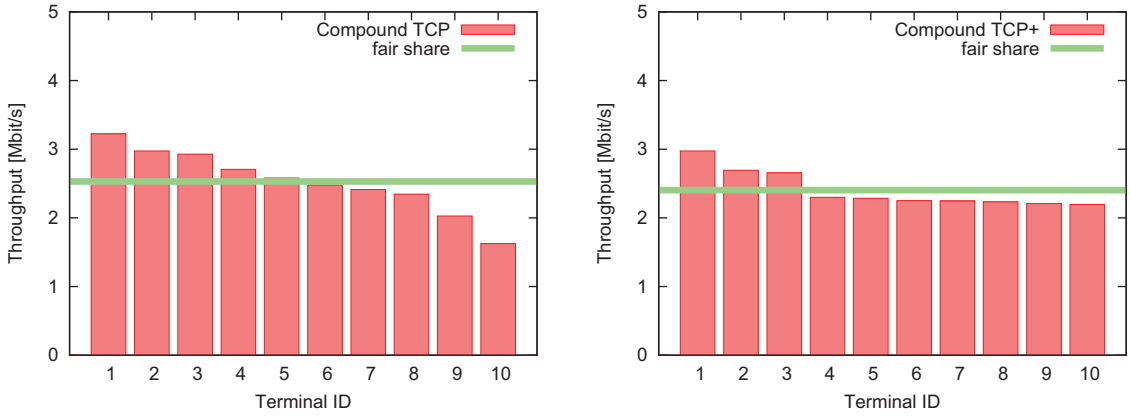


図 28: 実験ネットワークにおける Compound TCP，Compound TCP+ の Fairness Index

おける各端末のスループットと，fair share を示す．なお，無線端末が 2 台から 9 台存在する場合の結果は，付録 B に記載する．図 29 より，Compound TCP よりも Compound TCP+ は，公平性が高いことがわかる．



(a) Compound TCP　　(b) Compound TCP+

図 29　無線端末が 10 台存在する時のスループット

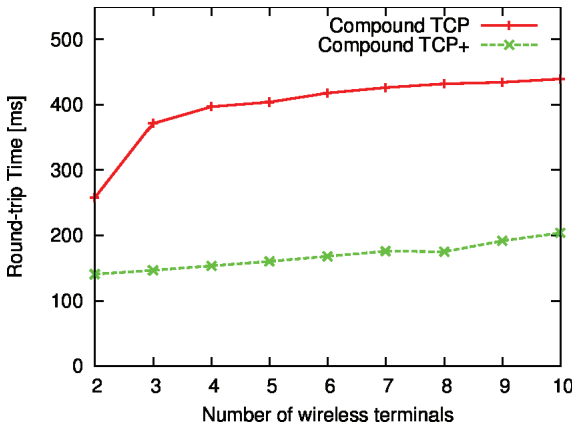


図 30: 実験ネットワークにおける Compound TCP，Compound TCP+ のラウンドトリップ時間

次に，図 30 に端末数の変化に対する平均ラウンドトリップ時間を示す．図 30 から，Compound TCP+ の平均ラウンドトリップ時間は，シミュレーションによる評価と同様に，Compound TCP と比較して非常に低いことがわかる．これは，Compound TCP がバッファを埋め尽くす輻輳制御を行うのに対して，Compound TCP+ はバッファを埋め尽くさない輻輳制御を行うためである．Compound TCP はパケット棄却を検出するまで，損失ウィンドウを増加させ続ける．そのため，アクセスポイントのバッファには，多くのパケットがバッファリングされ，その結果平均ラウンドトリップ時間は，非常に高くなる．それに対して，Compound TCP+ は，アクセスポイントのバッファを埋め尽くさないように制御する．バッファを埋め尽くしていないように制御することは，Compound TCP+ のラウンドトリップ時間が，Compound TCP と比較して低いことからわかる．

さらに，図 31 に，端末数の変化に対する平均合計スループットを示す．図 31 から，シミュレーション結果と異なり，Compound TCP+ の合計スループットは，Compound TCP の合計スループットと，ほぼ等しいことがわかる．これは次のように説明できる．実験ネットワークでは，シミュレーションよりも，平均ラウンドトリップ時間が小さい．すなわち，ネットワークの滞留パケット数が少ない．これは，実験ネットワークでは，シミュレーションに比べて，アクセスポイントでのパケットの処理時間が，短いためと考えられる．ネットワーク中の滞留パケットが少ない時，損失ベースの輻輳制御は，ネットワークのリンク，ルータのバッファをすべて埋め尽くそうとする，どん欲な制御を行う．その結果，損失ウィンドウは大きくなり，Compound TCP+ の合計スループットは，シミュレーション結果よりも高くなったと考えれる．以上の結果より，無線 LAN において，Compound TCP+ は，バッファを埋め尽くさない制御を行うことによって，スループットが低下することなく，高い公平性を維持することを示した．

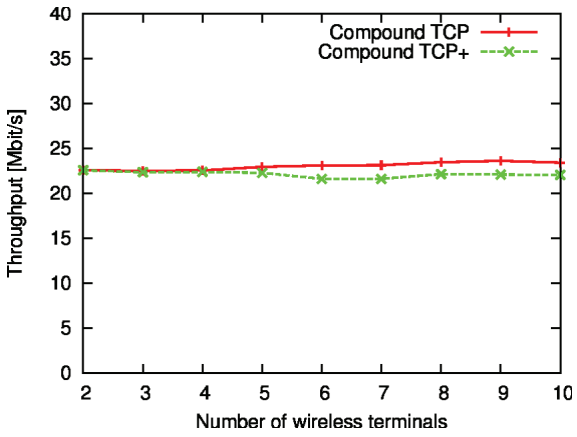


図 31: 実験ネットワークにおける Compound TCP，Compound TCP+ の合計スループット

6　Compound TCP+ の広帯域ネットワークにおける性能評価

Compound TCP+ は，軽度の輻輳状態である時，損失ウィンドウを増加させない．従って，広帯域ネットワークにおいて帯域を使い切ることができない恐れがある．そこで本章では，広帯域ネットワークにおける Compound TCP+ の性能評価をシミュレーションにより行う．シミュレーションによる性能評価から，広帯域ネットワークにおいて Compound TCP+ が十分なスループットを得ることを示す．

図 32 に，広帯域ネットワークを想定したシミュレーションのネットワークモデルを示す．送信端末からルータまでの帯域は 40 [Gbit/s]，遅延は 5 [ms] に設定した．また，ルータ間の帯域は 10 [Gbit/s]，遅延は 100 [ms] に設定した．さらにパケットサイズは 1500 [Byte] に設定し，ルータのバッファサイズは 500 [packet] に設定している．このようなネットワーク環境において，Compound TCP+ コネクションが 1 本存在する場合の 500 秒間のシミュレーションを行う．表 4 に，シミュレーション環境をまとめる．

図 33 に，Compound TCP と Compound TCP+ のスループットの時間変化を表す．図 33 から，Compound TCP と Compound TCP+ それぞれのコネクションのスループットは，設定した帯域上限まで使い切っていることがわかる．Compound TCP+ は，軽度の輻輳状態である場合，損失ウィンドウの増加を行わない．しかし，ネットワークに空き帯域がある場合，Compound TCP+ は遅延ベースの輻輳制御によってその帯域を活用するように制御する．以上より，Compound TCP+ は広帯域ネットワークにおいても，Compound TCP と同様に高いスループット達成可能であることが分かる．

表 4　広帯域ネットワークを想定したシミュレーション環境

コネクション数	1
往復伝搬遅延時間	220 [ms]
ルータ間の帯域	10 [Gbit/s]
パケットサイズ	1500 [byte]

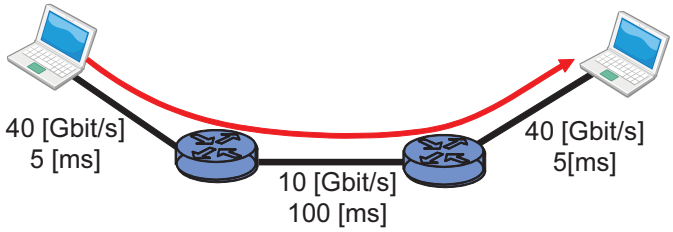


図 32　広帯域ネットワークを想定したシミュレーションモデル

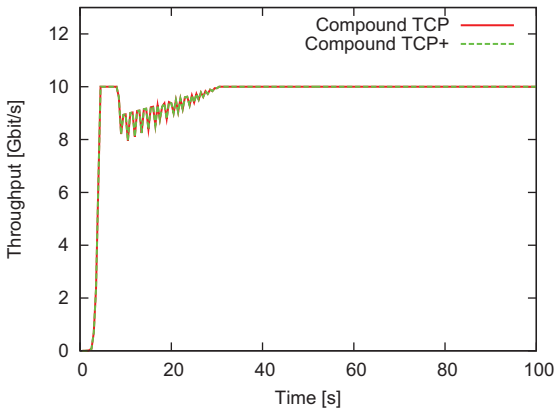


図 33　広帯域ネットワークにおける Compound TCP と Compound TCP+ のスループット

7　まとめと今後の課題

本論文では，広帯域なネットワーク向けに提案された Compound TCP の無線 LAN における問題点を，シミュレーションにより明らかにし，さらに，Compound TCP を改善した Compound TCP+ を提案した．加えて，シミュレーションおよび実験ネットワークによる実験により，Compound TCP+ の有効性を示した．

本論文では，まず，2 章において，インターネットの歴史に始まり，現在のインターネットで広く用いられているトランスポート層プロトコルである TCP の制御を述べた．さらに，広帯域ネットワークにおける TCP の問題，および，その問題を解決する既存の手法を述べた．

次に，3 章では，まず現在の無線 LAN の普及状況や，無線 LAN におけるアクセス制御を述べた．また，無線 LAN において TCP を用いた場合にスループット公平性が失われる原因と，その問題に対する既存の改善手法を述べた．

4 章では，広帯域高遅延なネットワーク向けに提案された Compound TCP の輻輳制御を述べ，その後，無線 LAN における Compound TCP の性能評価を行った．その結果，無線 LAN において Compound TCP を用いた場合，同一環境下で通信を行っているにも関わらず，スループットが非常に高い端末とスループットが非常に低い端末が混在する，スループットの公平性が失われることを明らかにした．

5 章では，無線 LAN における Compound TCP の問題を解決する Compound TCP+ の輻輳制御を提案した．Compound TCP+ は，ネットワークが軽度の輻輳状態である場合の Compound TCP の制御を変更する．具体的には，Compound TCP+ は，アクセスポイントのバッファが溢れる重度の輻輳状態となる前に，損失ウィンドウを減少させる．これにより，Compound TCP+ は，アクセスポイントのバッファ溢れを回避し，無線 LAN においてスループット公平性が失われる Compound TCP の問題を解決する．

さらに，無線 LAN における Compound TCP+ をシミュレーションおよび実験ネットワークにおける実験によって性能評価を行った．シミュレーションから，Compound TCP+ の公平性は Compound TCP よりも高いことを示した．さらに，Compound TCP+ を用いた場合の合計スループットは，Compound TCP を用いた場合の合計スループットと，同等程度のスループットを獲得しており，スループットの劣化が少ないことを示した．さらに，実験ネットワークにおける実験から，Compound TCP+ の公平性は Compound TCP より高いことを示した．

6 章では，広帯域なネットワークにおける Compound TCP+ の性能評価を，シミュレーションにより行った．その結果，ボトルネックリンクの帯域が 10 [Gbit/s] の回線において，Compound TCP+ は，10 [Gbit/s] のスループットを得ることを示し，広帯域なネットワークにおいて帯域を有効に活用することを示した．

以下では今後の課題を述べる．まず，実ネットワークにおいて無線端末の台数がさらに増加した場合における，Compound TCP+ の性能評価を行うことが挙げられる．本論文では実験ネットワークにおいて無線端末が 10 台存在する場合の性能評価を行った．しかし，無線 LAN を搭載した機器の普及により，1 台のアクセスポイントに 10 台以上の無線端末が接続されることは少なくない．そこで，さらに多くの無線端末が存在する場合の性能評価を行う必要があると考えられる．

また，本論文ではシミュレーションおよび実験ネットワークにおける Compound TCP+ の性能評価を行った．しかしながら，シミュレーションや実機を用いた実験では，ネットワークのあらゆる環境において，その有効性が示されるわけではない．そこで，Compound TCP+ の数学的解析手法による評価を行うことが重要であると考えられる．具体的には，[47] の手法を拡張し，Compound TCP+ の性能評価を行う．数学的解析手法による性能評価により，Compound TCP+ の有効性をさらに保証し，また，Compound TCP+ の正当性を示す

ことができる．

さらに，現在，最大伝送速度 6.93 [Gbit/s] である，無線 LAN 規格 IEEE 802.11ac が策定され，その規格に準拠した製品が販売され始めている．本論文では無線 LAN 規格に IEEE 802.11g または IEEE 802.11a を用いて Compound TCP および Compound TCP+ の性能評価を行っている．そのため，本論文での性能評価では，ネットワークのボトルネックが無線 LAN の区間に存在する．しかし，IEEE 802.11ac を用いた場合，ネットワークのボトルネックとなる区間が無線 LAN 区間ではなく，有線 LAN 区間となる．ネットワークのボトルネックとなる区間が変わることで，Compound TCP+ の性能に何らかの影響を与えるおそれがある．そこで，IEEE 802.11ac を用いた場合の Compound TCP+ 性能評価を行うことが重要であると考えられる．

また，本論文ではファイルサイズが無限大のファイルを転送した場合の Compound TCP および Compound TCP+ の性能評価を行っている．しかし，実際のインターネットにおける Web のトラヒックは，ファイルサイズが大きなファイルでも 1 [MByte] 程度であると考えられる．そこで，実際のインターネットにおける Web のトラヒックを考慮したファイル転送を行う場合の Compound TCP+ の性能評価が挙げられる．

謝辞

本研究と本論文を終えるにあたり，御指導，御教授を頂きました，大阪電気通信大学大学院総合情報学研究科の登尾啓史教授に深く感謝致します．また，本論文をご精読頂きました，渡邊郁教授，上善恒雄教授に深謝致します．

さらに，本研究に関する議論やアドバイス，実験設備の提供，その他，研究を遂行するうえで必要な作業などをいろいろと御支援していただきました，久松潤之准教授に深く感謝致します．また，これまでの学生生活を通じて，基礎的な学問などを御教授頂いた，大阪電気通信大学総合情報学部情報学科の升谷保博教授，鴻巣敏之教授，北嶋暁教授，南角茂樹准教授，大西克彦准教授，小枝正直准教授に深く感謝致します．

また，畑中貴弘さんをはじめとする大阪電気通信大学大学院総合情報学研究科コンピュータサイエンス専攻の卒業生を含む皆様や，高速ネットワーク研究室の卒業生を含む，皆様に心から御礼申し上げます．

Dayz 株式会社の玉城朝様には，経済的にサポートいただきました．そのおかげで，学会発表や論文投稿ができました．深く感謝致します．

最後に，博士後期課程まで，日々支えてくれた家族に感謝します．これまでの大学生活は，経済的にも精神的にも家族の理解と支え無しでは続けられませんでした．ありがとうございました．

参考文献

[1] I. T. Union, "Global numbers of individuals using the internet, total and per 100 inhabitants, 2001-2013." available at `http://www.itu.int/ITU-D/ict/statistics/material/excel/2011/Internet_users_01-11_2.xls`.

[2] I. I. J. Inc., "Internet infrastructure review." available at `http://www.iij.ad.jp/company/development/report/iir/pdf/iir_vol16_report.pdf`.

[3] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," in *Proceedings of ACM SIGCOMM '98*, pp. 303–314, Sept. 1998.

[4] J. Padhye, V. Firoiu, D. F. Towsley, and J. F.Kurose, "Modeling TCP Reno performance: A simple model and its empirical validation," *IEEE/ACM Transactions on Networking*, vol. 8, pp. 133–145, Apr. 2000.

[5] K. Takagaki, H. Ohsaki, and M. Murata, "Stability analysis of a window-based flow control mechanism for TCP connections with different propagation delays," in *Proceedings of INET 2000*, July 2000.

[6] H. Hisamatsu, H. Ohsaki, and M. Murata, "Steady state and transient state behaviors analyses of TCP connections considering interactions between TCP connections and network," *International Journal of Communication Systems*, vol. 18, pp. 619–637, Sept. 2005.

[7] T. Tsugawa, N. Fujita, T. Hama, H. Shimonishi, and T. Murase, "TCP-AFEC: An adaptive FEC code control for end-to-end bandwidth guarantee," in *Proceedings of PV2007*, pp. 294–301, July 2007.

[8] G. Hasegawa, K. Yamanegi, and M. Murata, "TCP congestion control mechanisms for achieving predictable throughput using inline network measurement," *IEICE Transactions on Communications*, vol. E91-B, pp. 3945–3955, Dec. 2008.

[9] H. Obata, K. Ishida, S. Takeuchi, and S. Hanasaki, "TCP-STAR: TCP congestion control method for satellite internet," *IEICE Transactions on Communications*, vol. E89-B, pp. 1766–1773, June 2006.

[10] 平和弘, 小畑博靖, 石田賢治, "高速衛星インターネットに適したコネクション分割機構 TCP-gSTAR," **電子情報通信学会論文誌**, vol. J91-B, pp. 1587–1599, Dec. 2008.

[11] K. Yamanegi, T. Hama, G. Hasegawa, M. Murata, H. Shimonishi, and T. Murase, "Implementation experiments of the TCP proxy mechanism," in *Proceedings of APSITT2005*, pp. 17–22, Nov. 2005.

[12] I. Maki, G. Hasegawa, M. Murata, and T. Murase, "Performance analysis and improvement of TCP proxy mechanism in TCP overlay networks," in *Proceedings of ICC2005*, pp. 184–190, May 2005.

[13] S. Floyd, "Highspeed TCP for large congestion windows," *RFC 3649*, Dec. 2003.

[14] T. Kelly, "Scalable TCP: Improving performance in highspeed wide area networks," in *Proceedings of PFLDnet 2003*, pp. 82–91, Feb. 2003.

[15] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control(BIC) for fast long-distance networks," in *Proceedings of IEEE INFOCOM 2004*, pp. 2514–2524, Mar. 2004.

[16] S. Ha, I. Rhee, and L. Xu, "CUBIC: A New tcp-friendly High-Speed TCP Variant," *ACM SIGOPS Operating ASystems Review*, vol. 42, pp. 64–74, July 2008.

[17] C. Jin, D. Wei, S. H. Low, J. Bunn, H. D. Choe, J. C. Doyle, H. Newman, S. Ravot, S. Singh, F. Paganini, G. Buhrmaster, L. Cottrell, O. Martin, and W. chun Feng, "FAST TCP: From theory to experiments," *IEEE Networking*, vol. 19, pp. 4–11, Jan. 2005.

[18] S. Mascolo, C. Casetti, M. Gela, M.Y.Sanadidi, and R. Wang, "TCP Westwood: Bandwidth estimation for enhanced transport over wireless links," in *Proceedings of ACM MobiCom 2001*, pp. 287–297, 2001.

[19] L. A. Grieco and S. Mascolo, "Performance evaluation and comparison of Westwood+, New Reno, and Vegas TCP congestion control," *ACM Computer Communication Review*, vol. 3, pp. 25–38, 2004.

[20] K. Kaneko, T. Fujikawa, Z. Su, and J. Katto, "TCP-Fusion: A hybrid congestion control algorithm for high-speed networks," in *Proceedings of PFLDnet 2007*, pp. 31–36, 2007.

[21] H. Shimonishi, T. Hama, and T. Murase, "TCP-Adaptive Reno for improving efficiency-friendliness tradeoffs of TCP congestion control algorithm," in *Proceedings of PFLDnet 2006*, pp. 87–91, 2006.

[22] K. Tan, J. Song, Q. Zhang, and M. Sridharan, "Compound TCP: A scalable and TCP-friendly congestion control for high-speed networks," in *Proceedings of PFLDnet 2006*, Feb. 2006.

[23] ICT 総研, "公衆無線 LAN サービス利用者予測調査." available at `http://www.ictr.co.jp/report/20121022000015.html`.

[24] C. E. Perkins and E. M. Royer, "Ad hoc on-demand distance vector routing," in *Proceedings of MCSA1999*, pp. 90–100, Feb. 1999.

[25] P. Jaquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol for ad hoc networks," in *Proceedings of INMIC2001*, pp. 62–68, Dec. 2001.

[26] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance vector routing(DSDV) for mobile computers," in *Proceedings of ACM SIGCOMM'94*, pp. 234–244, Sept. 1994.

[27] D. B. Johnson, D. A. Maltz, and J. Broch, "DSR: the dynamic source routing protocol for multihop wireless ad hoc networks," *Ad Hoc Networking*, pp. 139–172, 2001.

[28] A. P. Jardosh, K. Papagiannaki, E. M. Belding, K. C. Almeroth, G. Iannaccone, and B. Vinnakota, "Green WLANs: On-demand WLAN infrastructures," *Journal of Mobile Networks and Applications*, vol. 14, pp. 798–814, Dec. 2009.

[29] T. Hiraguri, M. Ogawa, M. Umeuchi, and T. Sakata, "Power saving scheme suitable for wireless LAN in multimedia communications," *IEICE Transactions on Communications*, vol. E92-A, pp. 2184–2190, Sept. 2008.

[30] T.-H. Lee and J.-R. Hsieh, "An efficient scheduling algorithm for scheduled automatic power save delivery for wireless LANs," in *Proceedings of VTC 2012*, pp. 1–5, May 2010.

[31] M. Hashimoto, G. Hasegawa, and M. Murata, "Exploiting SCTP multi-streaming to reduce energy consumption of multiple TCP flows over a WLAN," in *Proceedings of GreeNETs 2012*, Oct. 2012.

[32] S. Pilosof, R. Ramjee, D. Raz, Y. Shavitt, and P. Sinha, "Understanding TCP Fairness over wireless LAN," in *Proceedings of IEEE INFOCOM 2003*, pp. 863–872, Apr. 2003.

[33] C. Wang and W. Tang, "A probability-Based algorithm to adjust contention window in IEEE 802.11 DCF," in *Proceedings of ICCCAS 2004*, pp. 418–422, June 2004.

[34] Y. Yang, J. Wang, and R. Kravets, "Distributed Optimal Contention Window Control for Elastic Traffic in Wireless LANs," in *Proceedings of IEEE INFOCOM 2005*, pp. 35–46, Apr. 2005.

[35] B. A. H. S. Abeysekera, T. Matsuda, and T. Takine, "Dynamic contention window control to achieve fairness between uplink and downlink flows in IEEE 802.11 WLANs," in *Proceedings of IEEE WCNC'07*, pp. 2109–2114, Mar. 2007.

[36] Y. Fukuda and Y. Oie, "Unfair and inefficient share of wireless lan resource among uplink and downlink data

traffic and its solution," *IEICE Transactions on Communications*, vol. E88-B, pp. 1577–1585, Apr. 2005.

[37] Y. Hirano and T. Murase, "Uplink TCP traffic Control with Monitoring downlink buffer for throughput fairness over wireless LANs," in *Proceedings of PIMRC 2009*, pp. 737–741, Sept. 2009.

[38] B. S. Abeysekera, T. Matsuda, and T. Takine, "Dynamic contention window control scheme in IEEE 802.11e EDCA-based wireless LANs," *IEICE Transactions on Communications*, vol. E93-B, pp. 56–64, Jan. 2010.

[39] M. Hashimoto, G. Hasegawa, and M. Murata, "A transport-layer solution for alleviating TCP unfairness in a wireless LAN environment," *IEICE Transactions on Communications*, vol. E94-B, pp. 765–776, Mar. 2011.

[40] J. Postel, "Transmission control protocol," *Request for Comments (RFC) 793*, Sept. 1981.

[41] D. Katabi, M. Handley, and C. Rohrs, "Congestion Control for High Bandwidth-Delay Product Networks," in *Proceedings of ACM SIGCOMM 2002*, pp. 89–102, Aug. 2002.

[42] K. Tokuda, G. Hasegawa, and M. Murata, "Performance analysis of highspeed tcp and its improvement for high throughput and fairness against tcp reno connections," in *Proceedings of High Speed Network Workshop (HSN 2003)*, Mar. 2003.

[43] R. Mbarek, M. T. B. Othman, and S. Nasri, "Performance evaluation of competing high-speed TCP protocols," *International Journal of Computer Science and Network Security*, vol. 8, pp. 99–105, June 2008.

[44] SAMAN, CONSER and ACIRI, "The network simulator – ns2." available at `http://www.isi.edu/nsnam/ns/`.

[45] K. Tan, J. Song, and Q. Zhang, "A compound TCP approach for high-speed and long distance networks," in *Proceedings of IEEE INFOCOM 2006*, pp. 1–12, July 2006.

[46] R. Jain, "Throughput Fairness Index: An Explanation," in *ATM Forum Contribution 99-0045*, Feb. 1999.

[47] S. H. Low, "A duality model of TCP and queue management algorithms," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 525–536, Aug. 2003.

付録 A　無線 LAN における Compound TCP のスループット

A.1　アクセスポイントと受信端末間の遅延が 5 [ms] 時

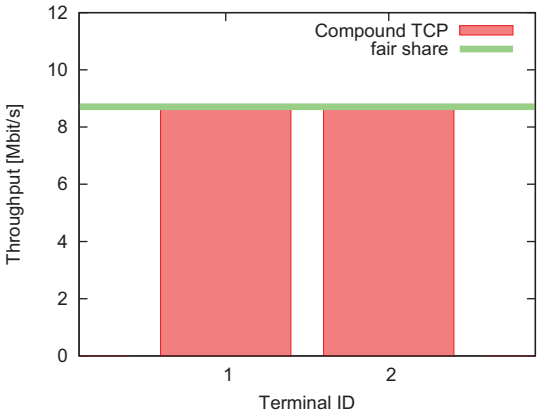


図 34　遅延 5[ms] 時において無線端末が 2 台存在する時のスループット

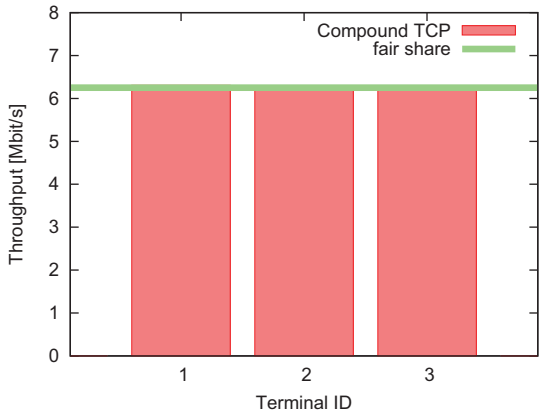


図 35　遅延 5[ms] 時において無線端末が 3 台存在する時のスループット

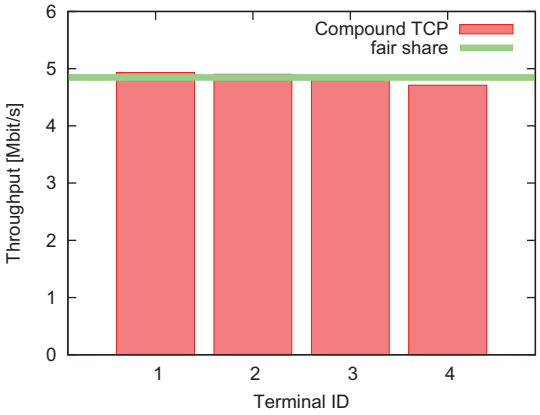


図 36　遅延 5[ms] 時において無線端末が 4 台存在する時のスループット

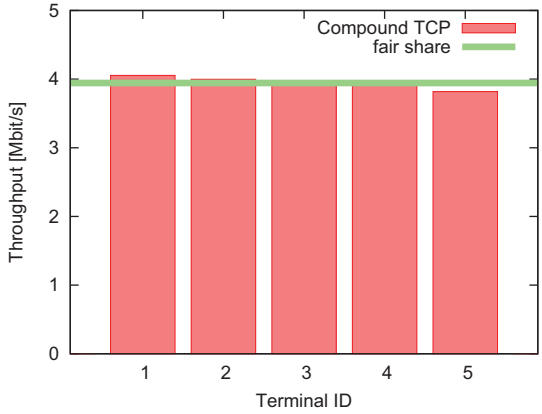


図 37　遅延 5[ms] 時において無線端末が 5 台存在する時のスループット

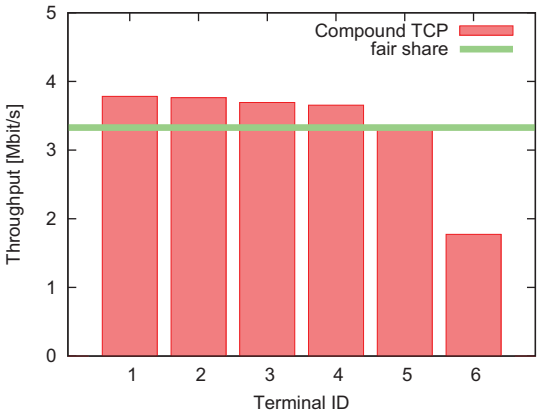


図 38　遅延 5[ms] 時において無線端末が 6 台存在する時のスループット

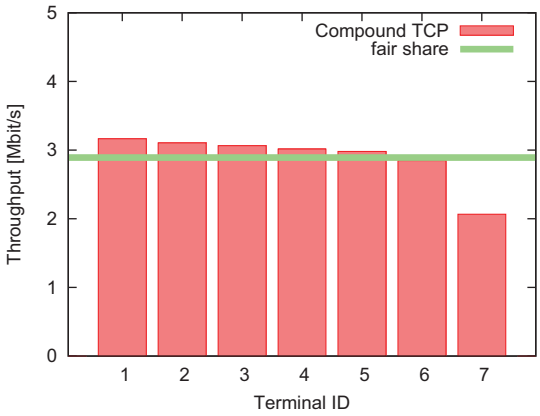


図 39　遅延 5[ms] 時において無線端末が 7 台存在する時のスループット

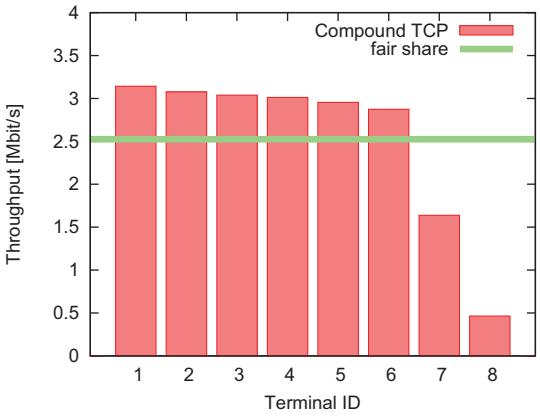


図 40　遅延 5[ms] 時において無線端末が 8 台存在する時のスループット

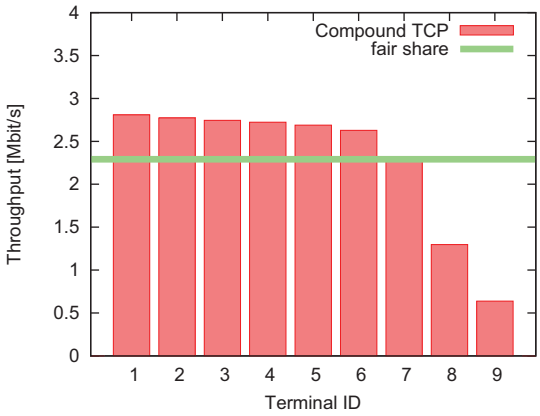


図 41　遅延 5[ms] 時において無線端末が 9 台存在する時のスループット

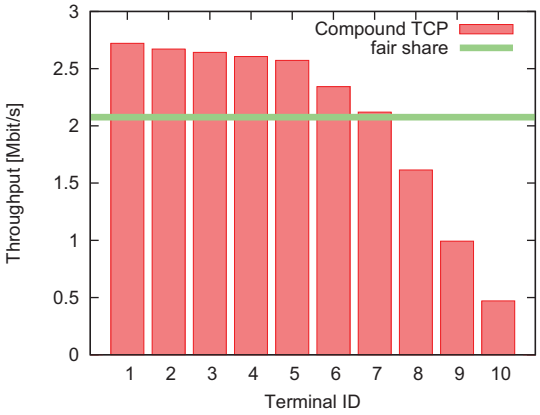


図 42　遅延 5[ms] 時において無線端末が 10 台存在する時のスループット

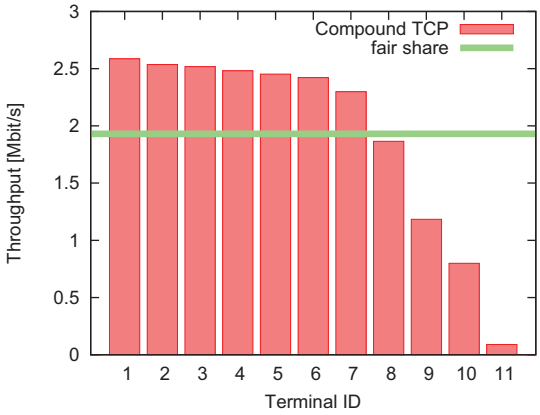


図 43　遅延 5[ms] 時において無線端末が 11 台存在する時のスループット

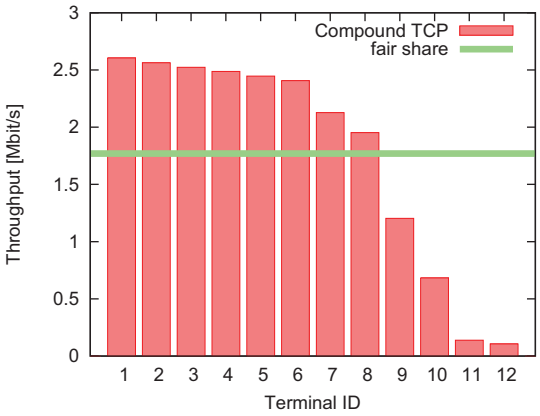


図 44　遅延 5[ms] 時において無線端末が 12 台存在する時のスループット

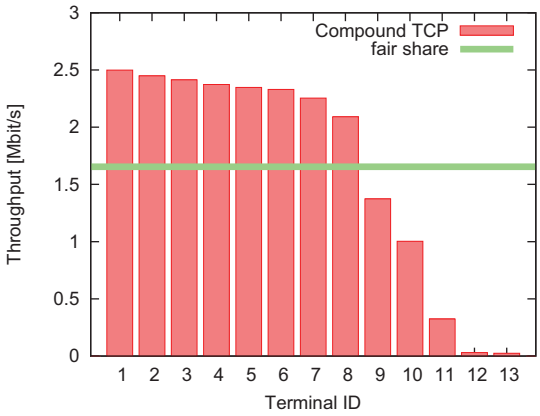


図 45　遅延 5[ms] 時において無線端末が 13 台存在する時のスループット

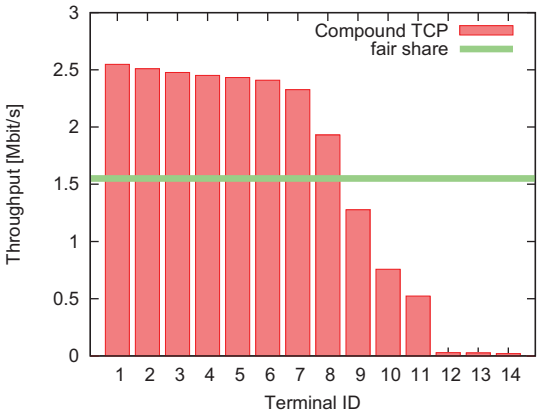


図 46　遅延 5[ms] 時において無線端末が 14 台存在する時のスループット

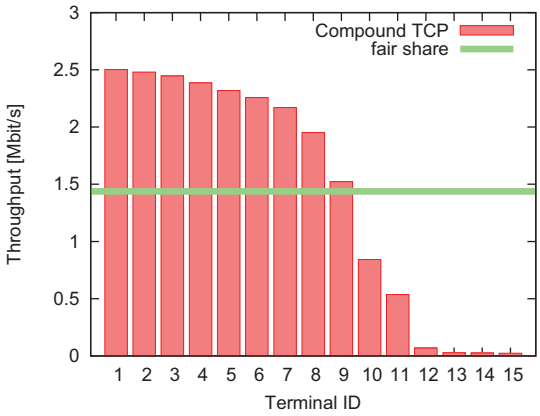


図 47　遅延 5[ms] 時において無線端末が 15 台存在する時のスループット

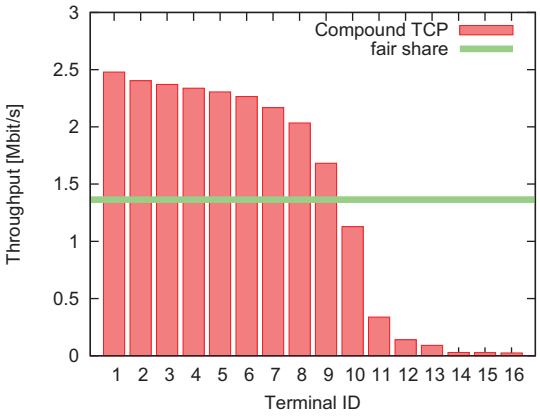


図 48　遅延 5[ms] 時において無線端末が 16 台存在する時のスループット

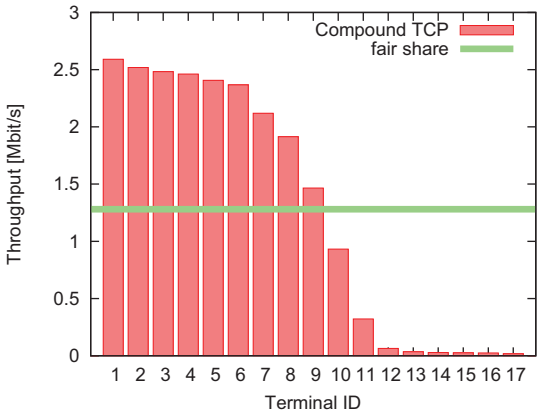


図 49　遅延 5[ms] 時において無線端末が 17 台存在する時のスループット

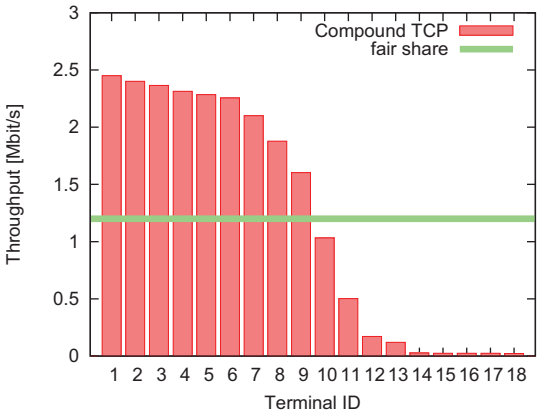


図 50　遅延 5[ms] 時において無線端末が 18 台存在する時のスループット

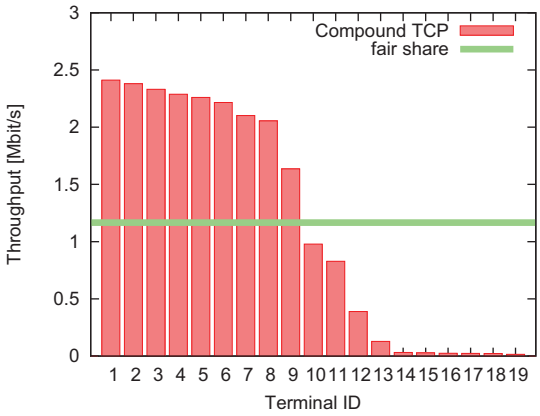


図 51　遅延 5[ms] 時において無線端末が 19 台存在する時のスループット

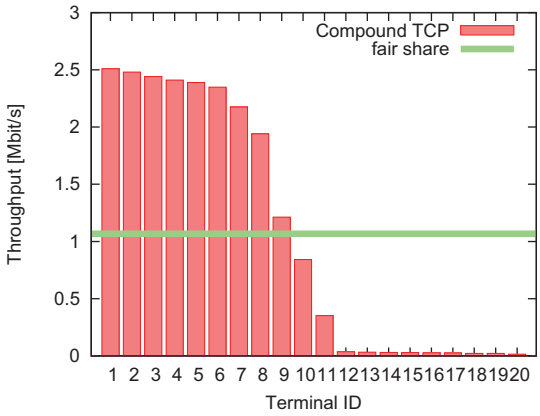


図 52: 遅延 5[ms] 時において無線端末が 20 台存在する時のスループット

A.2　アクセスポイントと受信端末間の遅延が 50 [ms] 時

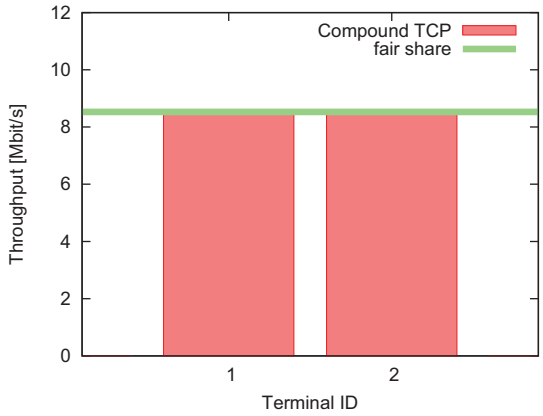


図 53　遅延 50[ms] 時において無線端末が 2 台存在する時のスループット

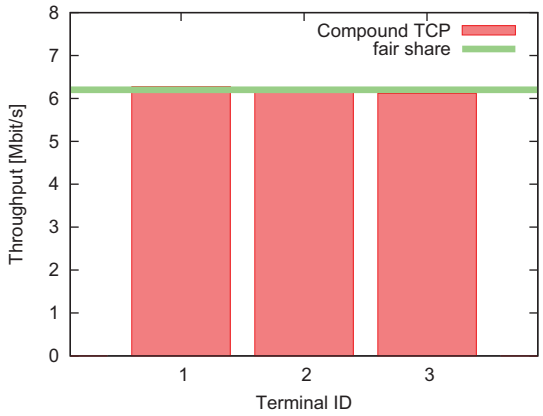


図 54　遅延 50[ms] 時において無線端末が 3 台存在する時のスループット

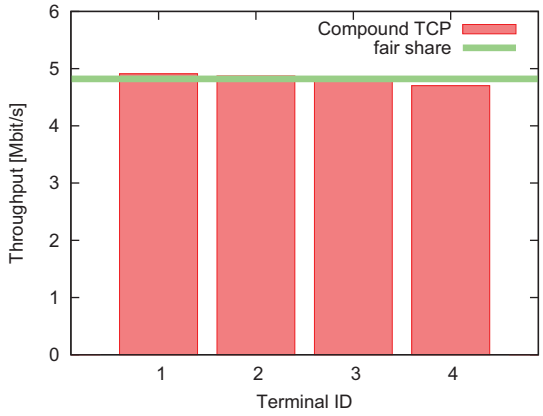


図 55　遅延 50[ms] 時において無線端末が 4 台存在する時のスループット

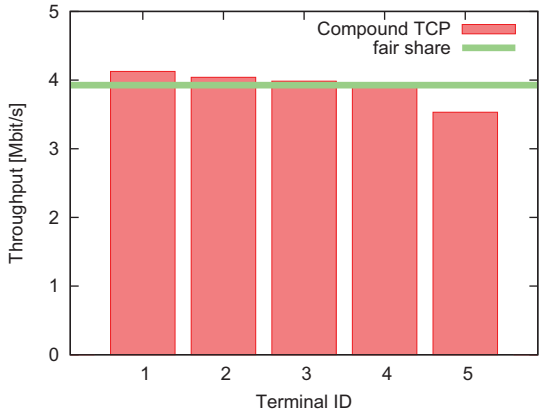


図 56　遅延 50[ms] 時において無線端末が 5 台存在する時のスループット

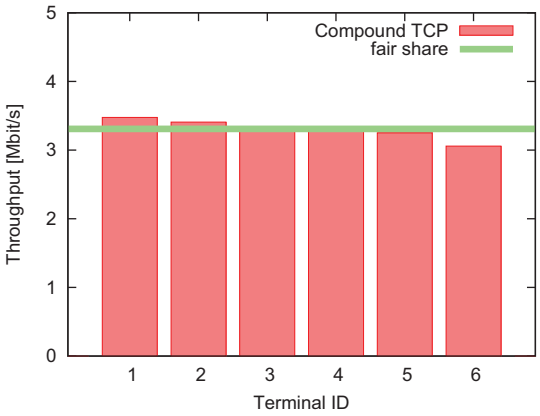


図 57　遅延 50[ms] 時において無線端末が 6 台存在する時のスループット

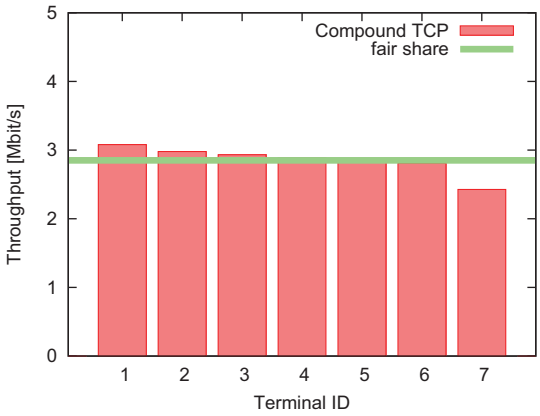


図 58　遅延 50[ms] 時において無線端末が 7 台存在する時のスループット

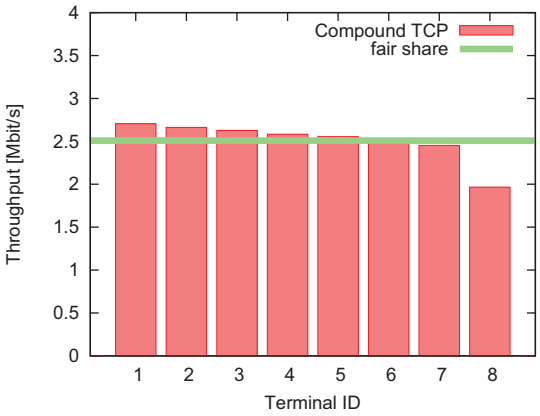


図 59　遅延 50[ms] 時において無線端末が 8 台存在する時のスループット

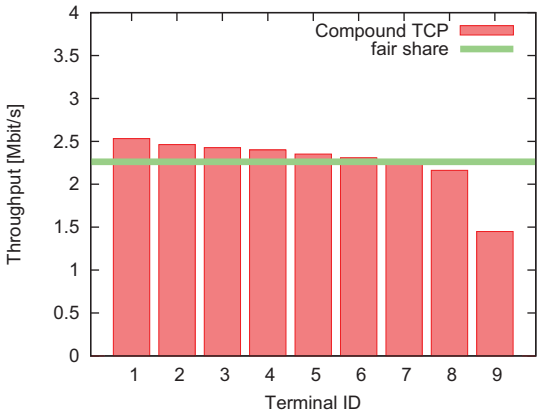


図 60　遅延 50[ms] 時において無線端末が 9 台存在する時のスループット

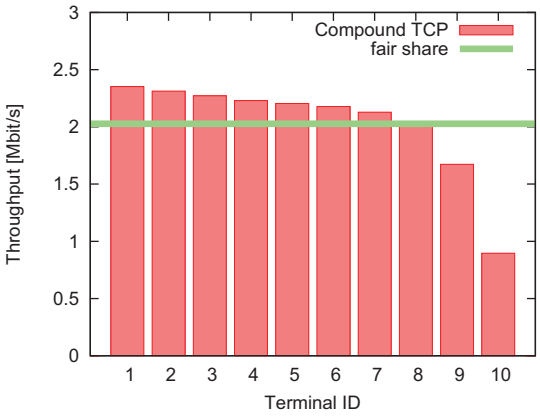


図 61　遅延 50[ms] 時において無線端末が 10 台存在する時のスループット

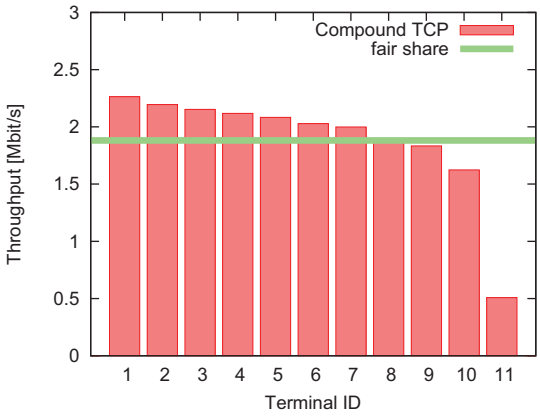


図 62　遅延 50[ms] 時において無線端末が 11 台存在する時のスループット

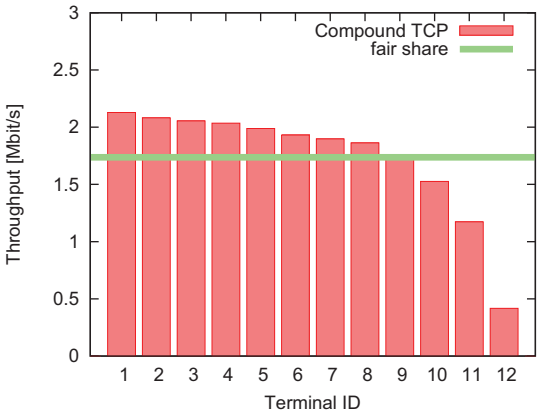


図 63　遅延 50[ms] 時において無線端末が 12 台存在する時のスループット

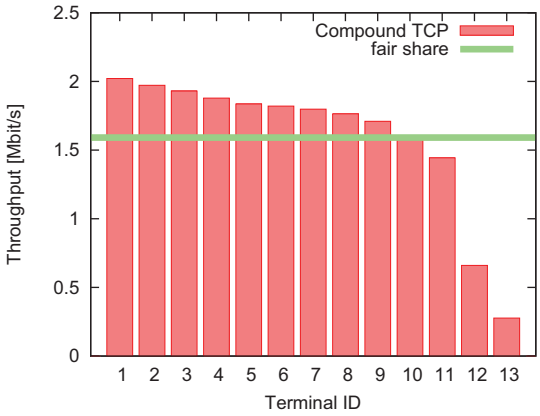


図 64　遅延 50[ms] 時において無線端末が 13 台存在する時のスループット

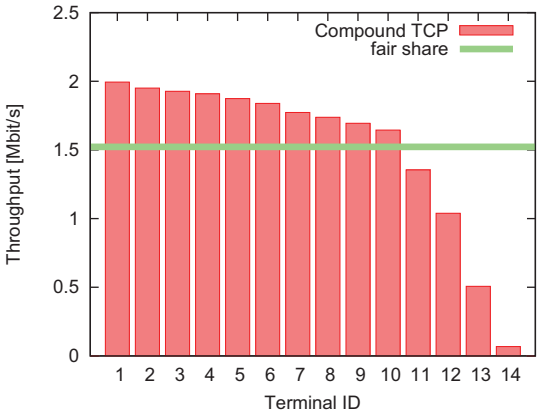


図 65　遅延 50[ms] 時において無線端末が 14 台存在する時のスループット

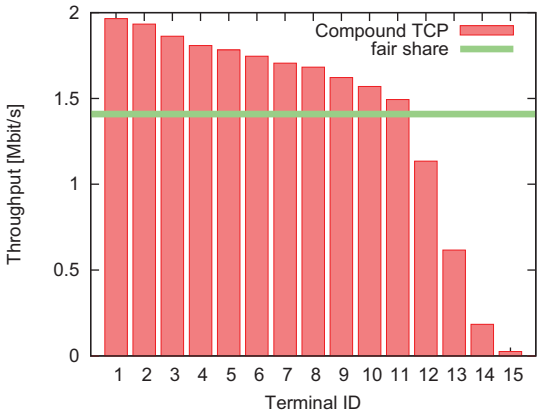


図 66　遅延 50[ms] 時において無線端末が 15 台存在する時のスループット

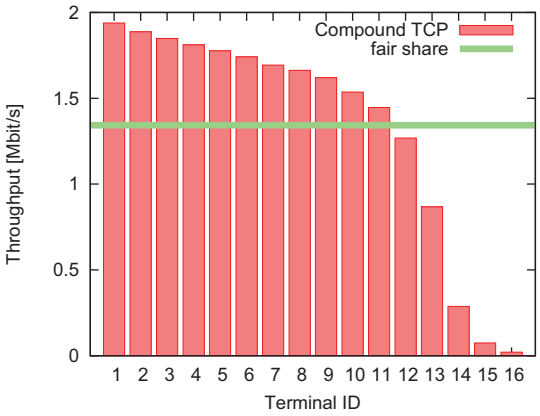


図 67　遅延 50[ms] 時において無線端末が 16 台存在する時のスループット

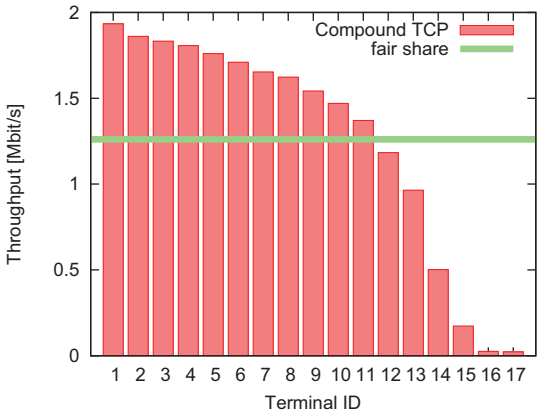


図 68　遅延 50[ms] 時において無線端末が 17 台存在する時のスループット

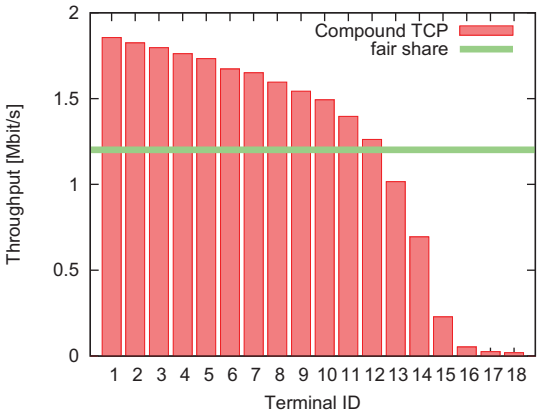


図 69　遅延 50[ms] 時において無線端末が 18 台存在する時のスループット

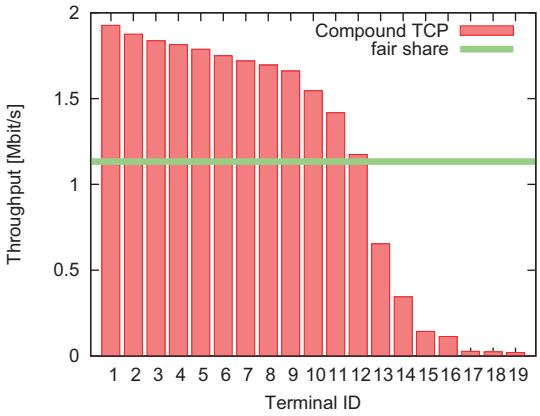


図 70　遅延 50[ms] 時において無線端末が 19 台存在する時のスループット

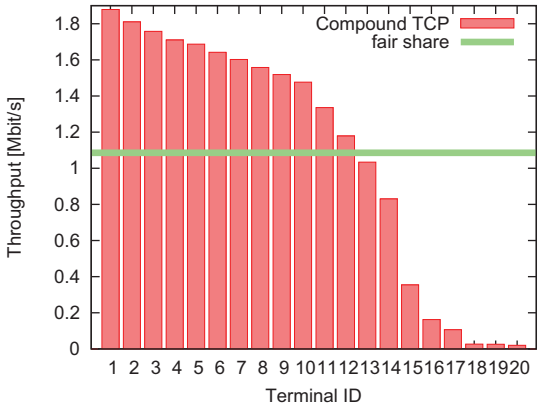


図 71: 遅延 50[ms] 時において無線端末が 20 台存在する時のスループット

A.3　アクセスポイントと受信端末間の遅延が 110 [ms] 時

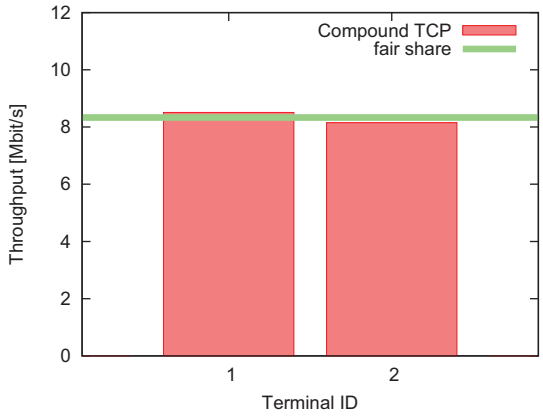


図 72　遅延 110[ms] 時において無線端末が 2 台存在する時のスループット

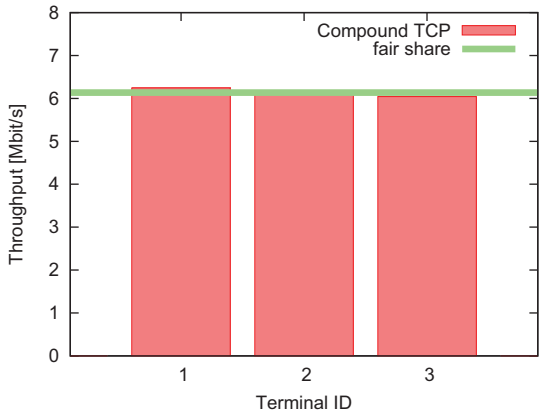


図 73　遅延 110[ms] 時において無線端末が 3 台存在する時のスループット

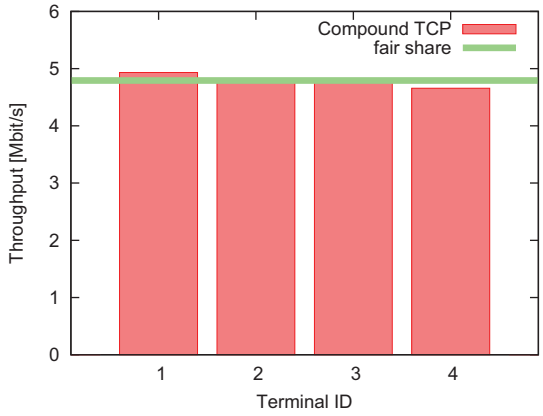


図 74　遅延 110[ms] 時において無線端末が 4 台存在する時のスループット

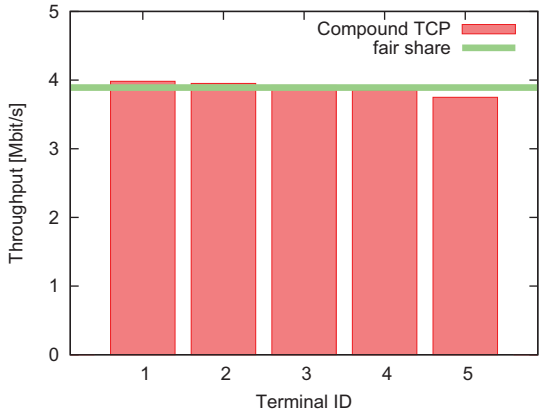


図 75　遅延 110[ms] 時において無線端末が 5 台存在する時のスループット

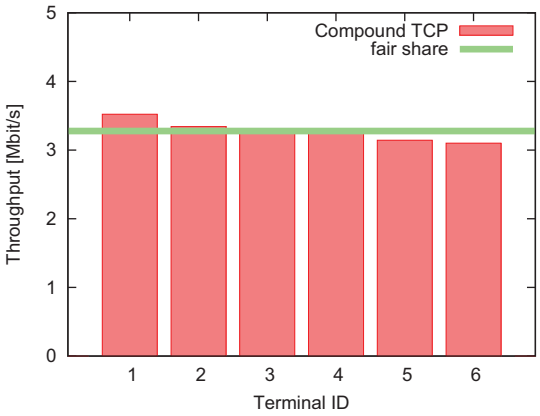


図 76　遅延 110[ms] 時において無線端末が 6 台存在する時のスループット

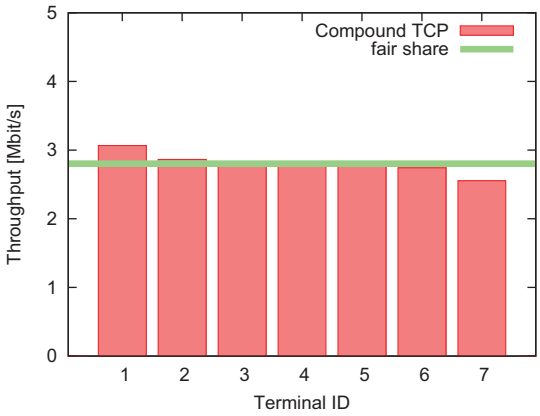


図 77　遅延 110[ms] 時において無線端末が 7 台存在する時のスループット

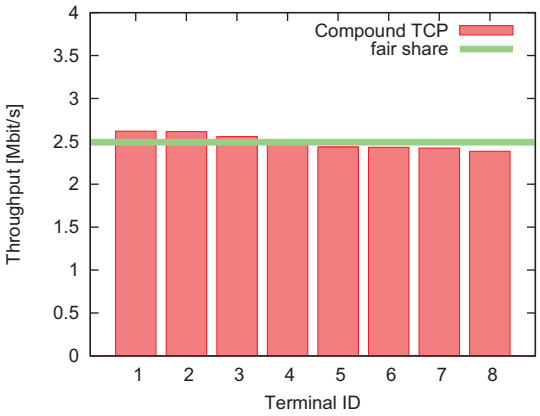


図 78　遅延 110[ms] 時において無線端末が 8 台存在する時のスループット

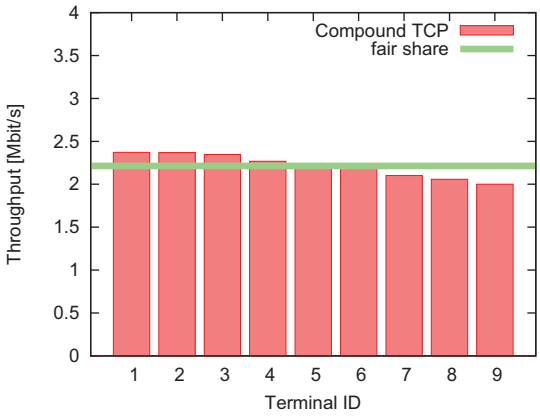


図 79　遅延 110[ms] 時において無線端末が 9 台存在する時のスループット

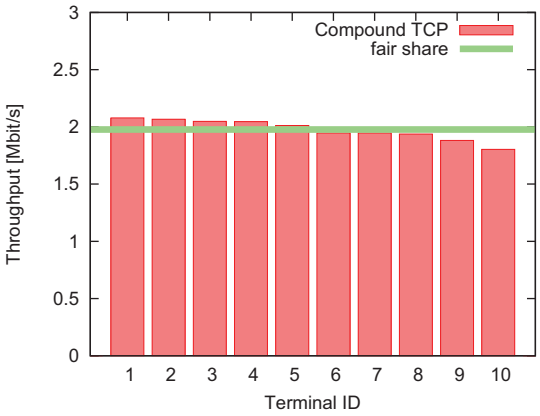


図 80　遅延 110[ms] 時において無線端末が 10 台存在する時のスループット

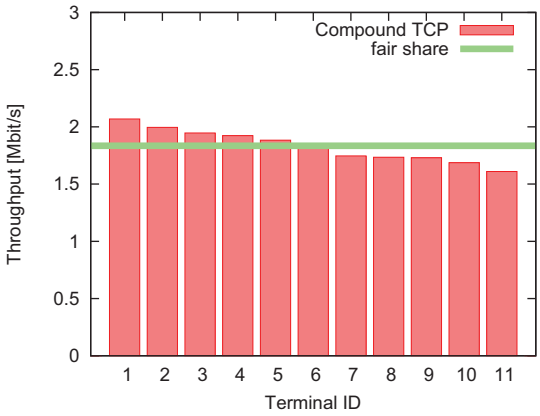


図 81　遅延 110[ms] 時において無線端末が 11 台存在する時のスループット

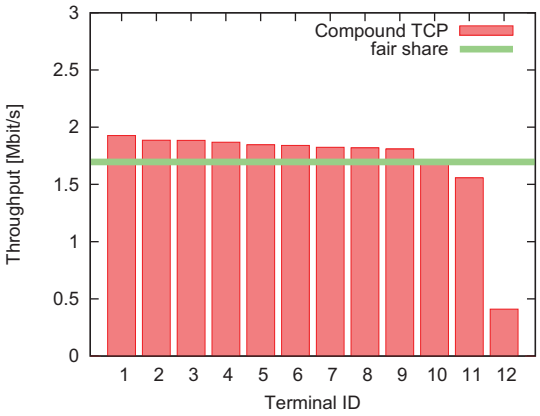


図 82　遅延 110[ms] 時において無線端末が 12 台存在する時のスループット

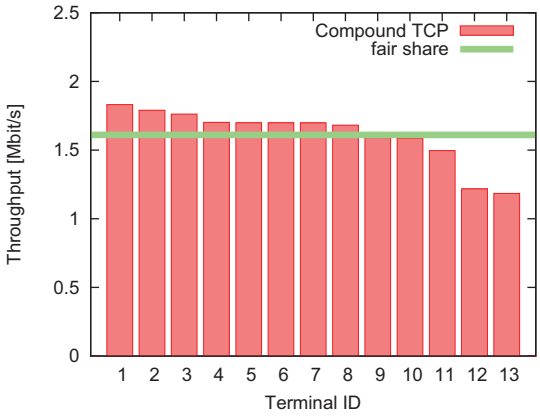


図 83　遅延 110[ms] 時において無線端末が 13 台存在する時のスループット

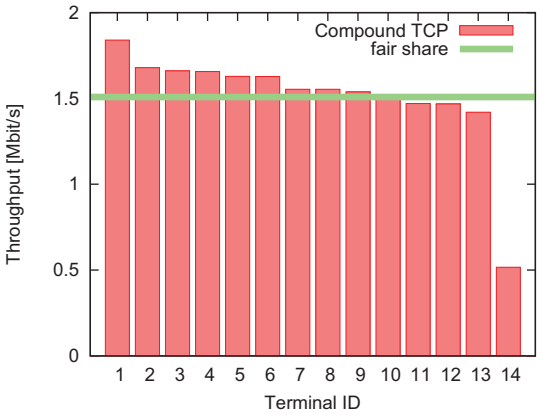


図 84　遅延 110[ms] 時において無線端末が 14 台存在する時のスループット

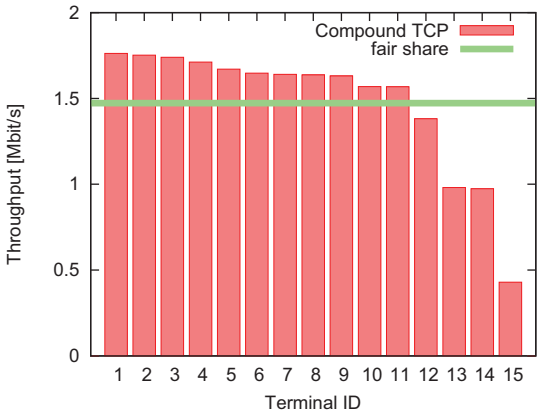


図 85　遅延 110[ms] 時において無線端末が 15 台存在する時のスループット

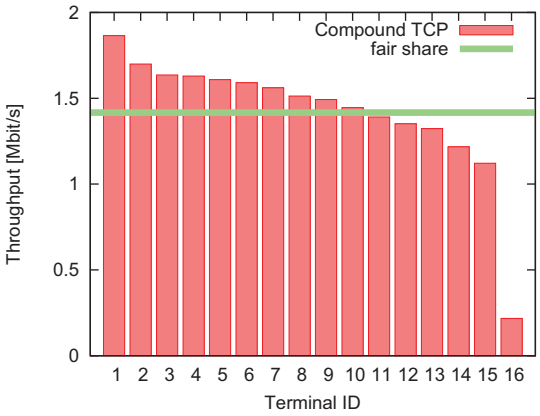


図 86　遅延 110[ms] 時において無線端末が 16 台存在する時のスループット

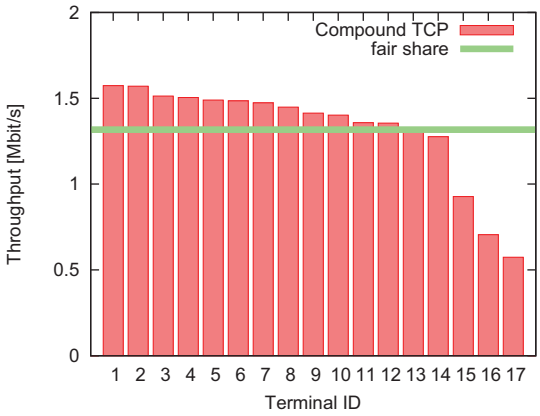


図 87　遅延 110[ms] 時において無線端末が 17 台存在する時のスループット

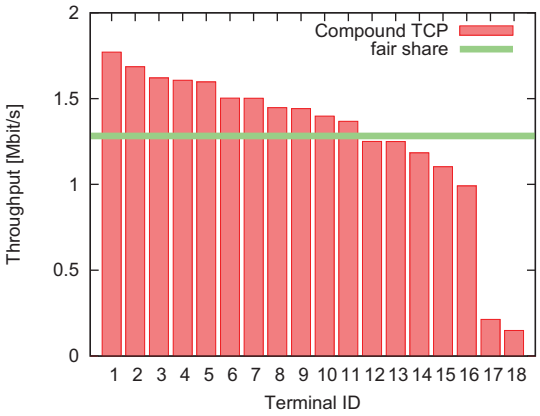


図 88　遅延 110[ms] 時において無線端末が 18 台存在する時のスループット

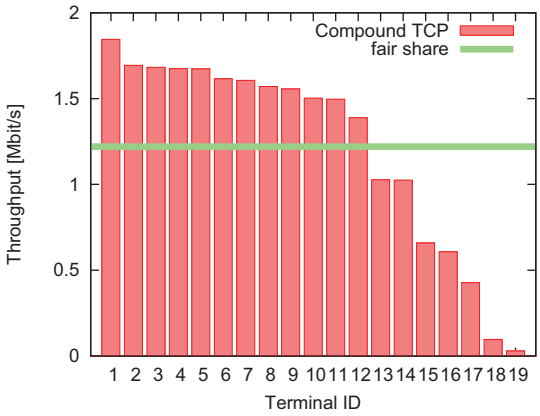
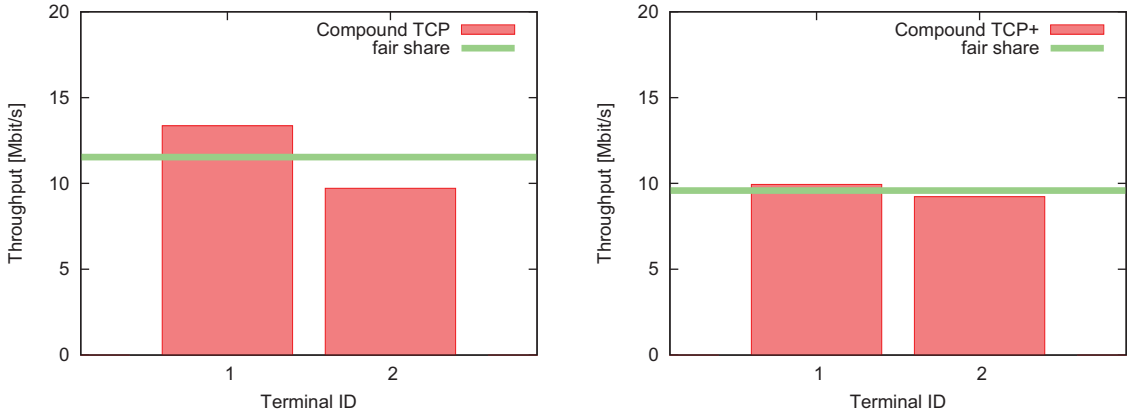


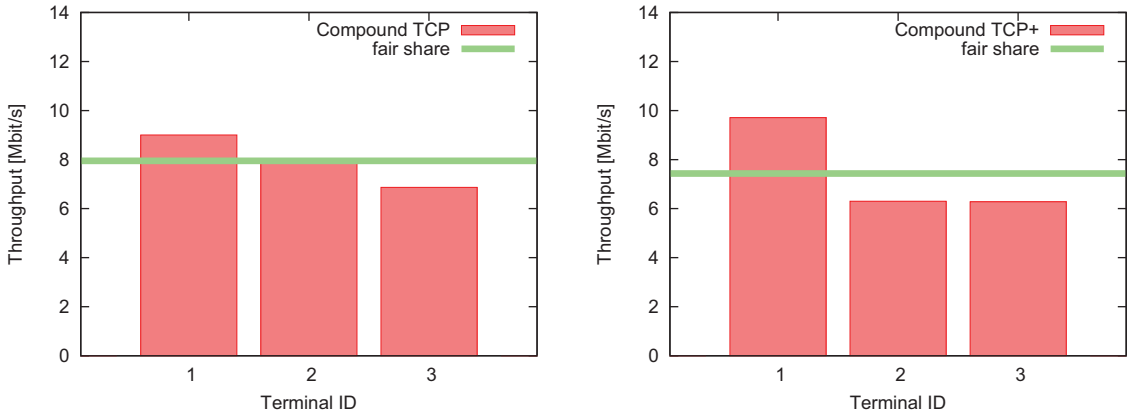
図 89　遅延 110[ms] 時において無線端末が 19 台存在する時のスループット

付録 B　実験ネットワークにおける Compound TCP および Compound TCP+ のスループット



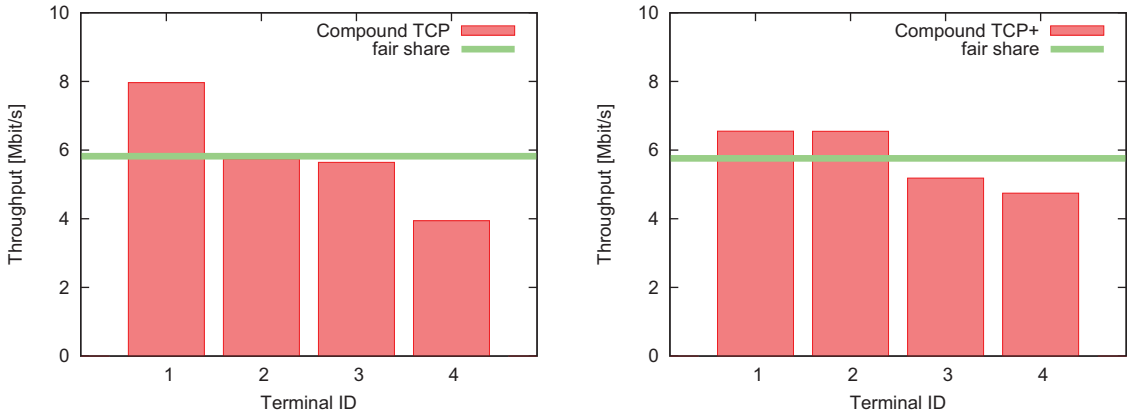
(a) Compound TCP　　(b) Compound TCP+

図 90　無線端末が 2 台存在する時のスループット



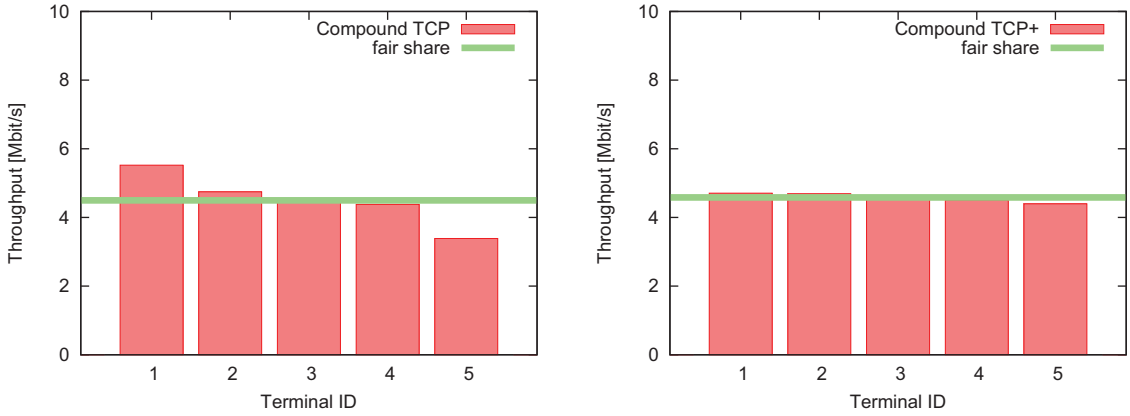
(a) Compound TCP　　(b) Compound TCP+

図 91　無線端末が 3 台存在する時のスループット



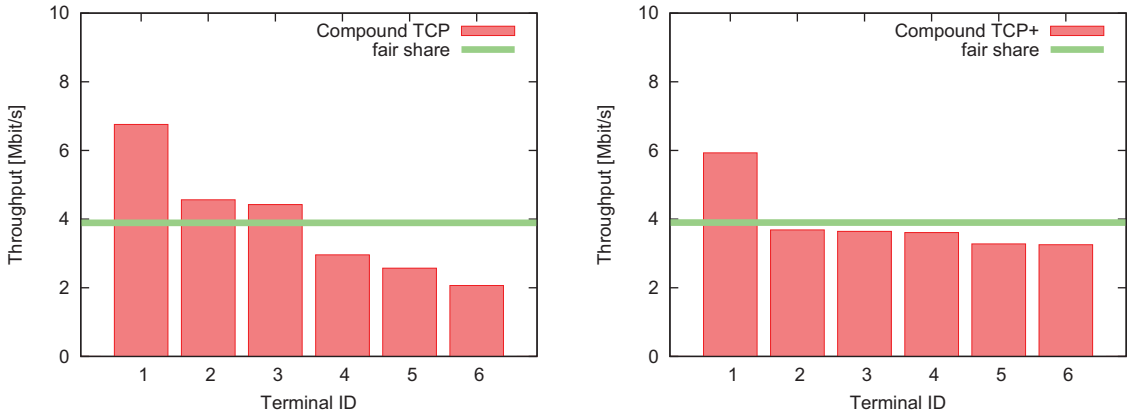
(a) Compound TCP (b) Compound TCP+

図 92　無線端末が 4 台存在する時のスループット



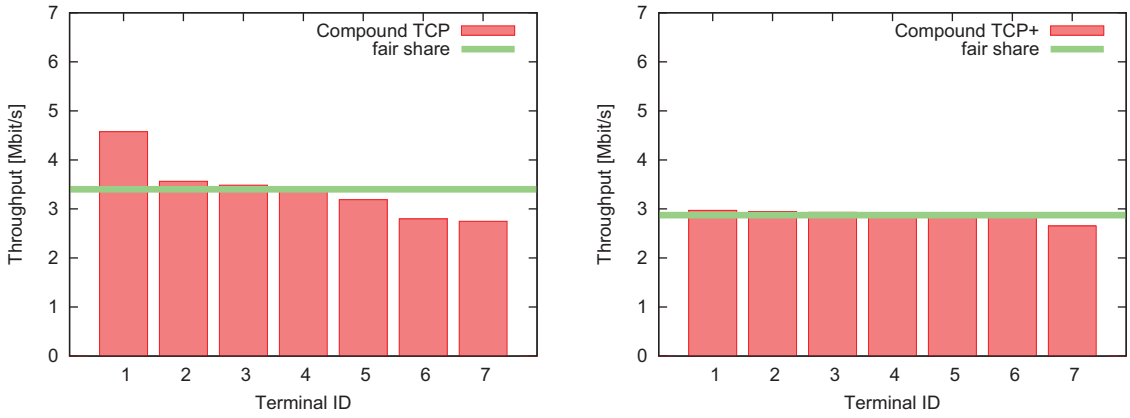
(a) Compound TCP (b) Compound TCP+

図 93　無線端末が 5 台存在する時のスループット



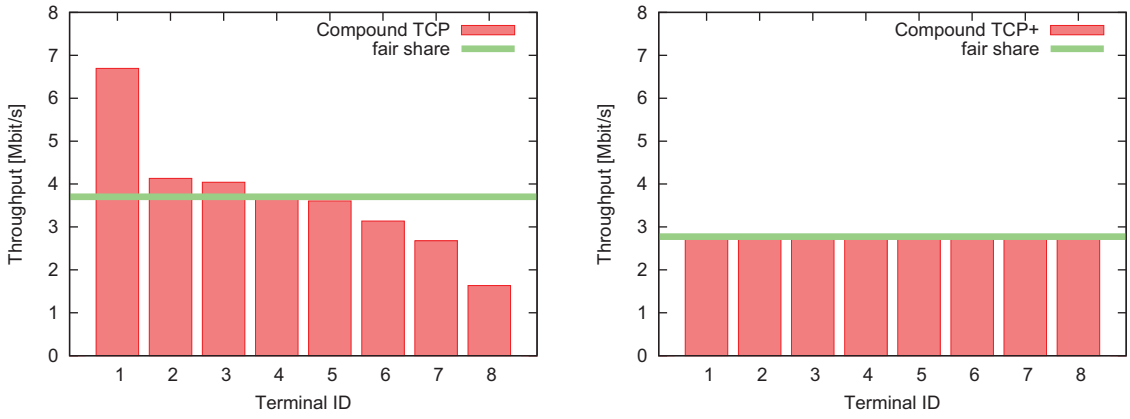
(a) Compound TCP (b) Compound TCP+

図 94　無線端末が 6 台存在する時のスループット



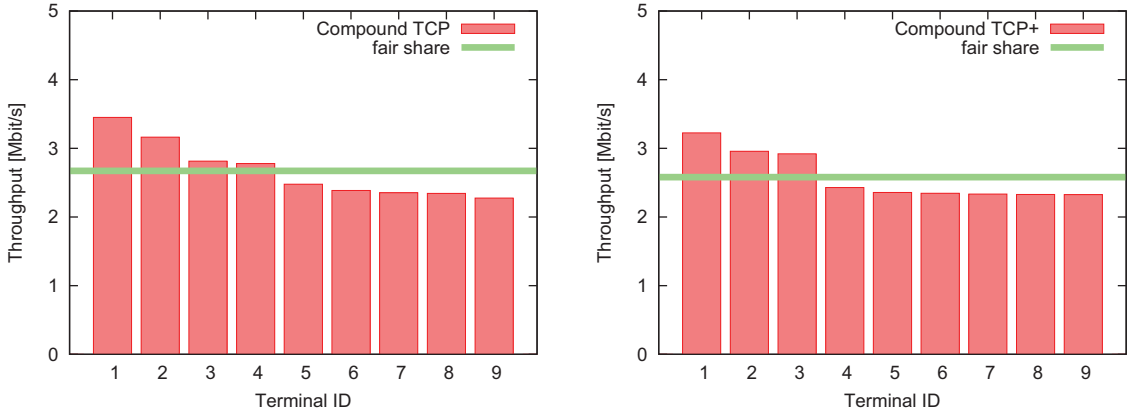
(a) Compound TCP　　(b) Compound TCP+

図 95　無線端末が 7 台存在する時のスループット



(a) Compound TCP　　(b) Compound TCP+

図 96　無線端末が 8 台存在する時のスループット



(a) Compound TCP　　(b) Compound TCP+

図 97　無線端末が 9 台存在する時のスループット