情報処理学会論文誌 Vol. 51 No. 2 466–477 (Feb. 2010)

パッシブRTT推定法を使用したAQMアルゴリズム

星原隼人[†1] 古賀久志[†1] 渡辺俊典[†1]

AQMは輻輳制御の技術であり，輻輳状態にあるルータ内のキュー溢れを防ぐため，TCPホストへどの程度の割合で輻輳通知が必要かを輻輳通知確率として計算する．この過程で，TCPコネクションのRTTは輻輳制御の効果に影響を与えるパラメータであるが，これを考慮したAQMはほとんど存在しない．よって，本論文ではルータで利用できるRTT推定法を導入し，得られたRTT値を明示的にアルゴリズムに組み込んだAQMを提案する．さらに，シミュレーションにより提案AQMが既存AQMよりも高い安定性を持つことを示す．

A New AQM Algorithm Exploiting Passive RTT Estimation

Hayato Hoshihara,[†1] Hisashi Koga[†1]
and Toshinori Watanabe[†1]

AQM is a technique for congestion control such that a router notifies congestion to a TCP sender when congestion occurs. Almost no AQM algorithms ever take the RTT values of TCP connections into account in congestion control, despite they are essential parameters. This paper proposes a new AQM algorithm that exploits them explicitly by introducing a passive RTT estimation technique in a router. The simulation results shows that our AQM stabilizes the queue length better than previous AQM algorithms.

1. まえがき

近年，インターネットにおける通信量増大にともない，輻輳制御はより重要性を増してきている．ルータ内での代表的な輻輳制御技術の1つにAQM（Active Queue Management）がある．AQMは，ルータにおいて輻輳の兆候を検出し，早期に輻輳通知を行ってエンドホストにトラヒック量を抑えるよう促す技術である．AQMのこの処理によって，ルータ内のキュー溢れと，キュー溢れにともない発生する大量のパケット廃棄が未然に防止される．代表的なAQMアルゴリズムには，RED（Random Early Detection）[3]やARED（Adaptive RED）[4]，REM（Random Exponential Marking）[5]などがあげられる．これらのAQMではTCPコネクションのRTT（Round Trip Time）を考慮していないが，以下の理由から，RTTはAQMの輻輳制御を左右する重要なパラメータである．

(I) AQMの輻輳通知がエンドホストのスループットに及ぼす影響は，RTTの大きさに依存している．たとえば，TCPエンドホストが輻輳制御後に減少したウィンドウサイズを増加させる際，RTTが小さいほどトラヒックレートの回復が早く，RTTが大きいほどトラヒックレートの回復が遅い．

(II) ルータがTCPコネクションへ輻輳通知を行う際，この効果としてルータでトラヒックレートの減少が観測されるまでにはRTTの時間経過が必要である[6]．この性質を，ルータが輻輳通知にECN（Explicit Congestion Notification）パケットマーキングを使用しているとして，以下の例で説明する．

(1) 時刻T_1に，TCPコネクションへの輻輳通知のため，ルータrがパケットPにマーキングする．
(2) Pが受信者側に到着した後，受信者はACKパケットP_{ack}にマークを付与して送信者に送出する．
(3) P_{ack}が送信者側に到着し，送信者はrから輻輳通知を受けたことを認識する．
(4) 輻輳通知を受けた送信者は送信レートを下げる．送信者からrへトラヒックが到着するまでの時間を経て，時刻T_2にrがレートの減少を検知する．

以上の過程から，$T_2 - T_1$はTCPコネクションのRTTと同程度の値になる．

以上をふまえ，本論文では，近年活発に研究が行われているパッシブなRTT推定法を利用して，RTT値を明示的にアルゴリズムに取り入れたAQMを提案する．

RTT値の取得には，セルフクロッキングベースのRTT推定法[7]を基に観測時間と観測データ量を削減させた独自の手法を使用し，省リソース化とリアルタイム性を実現する．NS-2シミュレータ上で提案AQMを既存AQMと比較し，提案AQMのキュー長やスループットの安定性と，トラヒックの変化に対するロバスト性を示す．

パッシブRTT推定は，有効な応用先としてAQMがあげられていることが多い．しか

†1 電気通信大学大学院情報システム学研究科
Graduate School of Information Systems, University of Electro-Communications

し，実際にこれらを AQM へ導入した研究例は少なく，我々が調査した限りでは Shyu らによる文献 8) が唯一の研究例である．しかし，Shyu らの方式ではフローごとの状態保持が必要となる．これに対し，本論文では RTT 代表値のみを記憶することでフローステートレスな AQM を実現している．なお，以降では，この Shyu らの方式のことを「Shyu-AQM」と呼ぶ．

本論文は以下のように構成される．2 章で関連する先行研究を紹介する．3 章で，RTT 推定法を含む，提案 AQM で必要なパラメータの取得法について説明し，これらを使用した提案 AQM のアルゴリズムを説明する．4 章で導入した RTT 推定法の精度検証を行う．5 章で提案 AQM の性能評価を行う．最後に，6 章で本論文のまとめを行う．

なお，本論文は同著者による文献 1)，2) の発展論文である．本論文では文献 1)，2) に対し，コネクション数の推定法を改良し，RTT 代表値に関する議論を加える．また，提案方式の評価に関して既存 AQM の比較対象を増やし，さらにマルチホップ環境などの実験項目を追加する．

2. 関 連 技 術

2.1 パッシブな RTT 推定法

RTT 推定法にはアクティブな推定法とパッシブな推定法が存在する．アクティブな推定法では RTT 推定のためのパケットをネットワークに投入する．パッシブな推定法ではパケットを投入せず，トラヒック観測のみで RTT 推定を行う．パケット投入によるトラヒックが発生しないため，AQM などの輻輳制御に適している．しかし，推定に必要な観測データの量や，RTT を推定可能なトラヒックが限定的であることなどが原因で，AQM への応用はいまだ困難とされている．これらの問題点を解消する可能性のある方式も含め，以降で既存のパッシブ RTT 推定法について紹介する．

TCP コネクション開始時の RTT を推定する方式として文献 9) がある．スリーウェイハンドシェイク，スロースタート時のトラヒックを検出する方式をそれぞれ提案しており，いずれも検出したトラヒックから RTT 値を推定する．しかしながら，本方式は TCP の通信開始時にしか利用できない．

TCP コネクションの存続中ならいつでも推定可能な方式としては，TCP コネクションの輻輳ウィンドウサイズを求めて RTT 値の推定に使用する方式がある[10]．対象コネクションのデータパケットと ACK パケットのシーケンス番号から輻輳ウィンドウサイズを推定し，これを保持して RTT 推定を行う．しかし，本方式は通信経路の対称性が条件となる．

TCP コネクションの存続中ならいつでも利用可能で，かつ非対称な通信経路でも RTT 推定が可能な方式として，セルフクロッキングベースの RTT 推定法[7] がある．セルフクロッキングとは，TCP エンドホストのスループットが経路中最も低速なリンクに従った量となって安定化する現象である．本方式は，セルフクロッキング時のトラヒックパターンが RTT の周期性を持つ性質を利用し，このパターンを検出することで RTT を推定する．以降では，短時間に連続してパケット送出が行われるときのトラヒックをバーストと定義し，TCP のトラヒックから RTT 周期のバーストが発生する現象について説明する．

TCP の送信者が通信を行い，このときのウィンドウサイズを W とする．送信者は W 個のパケットを送出し終えると，ACK パケットを受信するまでパケットの送出を行わない．よって，連続してパケット送出する期間，すなわちバースト期間の後，パケット送出のない期間が存在する．受信者はパケットの到着時に送信者へ ACK パケットを送出し，ACK にも同様にバースト期間と ACK パケットを送出しない期間が生じる．送信者は ACK パケットを受信するとウィンドウをスライドさせるが，ACK パケットがバースト的に届いた場合，送信者のウィンドウはバースト分スライドする．このため，送信者は再び以前送出したバーストと同量の W 個のパケット送出を行う．この結果，同量のバーストが RTT の周期で操り返される．経路中の輻輳などの影響を受けた場合，W 個のパケットはいっせいに到着せず，バースト w_1 の到着時刻から一定時間後に輻輳の影響を受けたバースト w_2 が到着するようなバーストの断片化が発生する（$W = w_1 + w_2$）．このような断片化後のバーストもまた RTT 周期で操り返されるため，RTT の周期性自体は損なわれない．

この現象は，エンドユーザの日常的な通信時でも発生する．たとえば，海外サーバからのファイルダウンロードのような長距離間の継続的なデータ転送時に本現象が発生する．**図 1** は，東京都調布市のクライアント PC から www.isi.edu へアクセスし，WEB ブラウザを使用して NS-2 シミュレータをダウンロードした際の到着パケット数の観測データである．1 ms のインターバルで到着したパケット数をカウントし，縦軸は到着したパケット数，横軸は時刻（JST 秒）を表す．同時刻の ping で得られた RTT は 120〜125 ms であり，図 1 では RTT の周期で似た大きさのバーストが発生している．

文献 7) では，自己相関分析によって似た大きさのバーストの周期性を検出し，この周期から RTT を推定する．しかし，厳密な自己相関分析を行うため，長期のトラヒック観測が必要である．加えて，観測時間にともなって観測データ量も増大する．このため，リアルタイム性と計算資源の確保が課題となる．

文献 11) では文献 7) の原理を基に，この課題を解決するバースト検出方式が提案されて

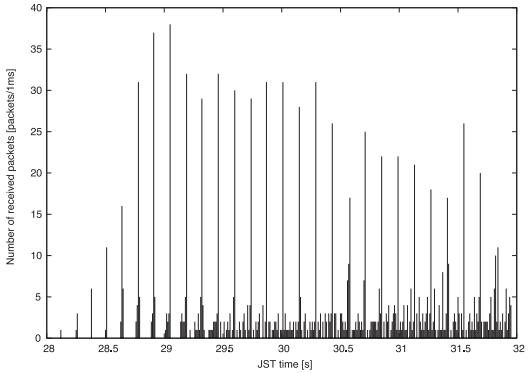


図 1　ダウンロード中の到着パケット数の推移
Fig. 1　Packet arrivals during data downloading.

いる．一定量以上の連続的な同一フローからのパケット到着をバーストと見なし，長期の観測，膨大なデータ，厳密な計算を行わずに RTT を推定する．

2.2　代表的な AQM アルゴリズム

AQM はキュー長やスループットから輻輳を見積もり，輻輳が激しくなるに従って 1 に近づく確率 p（$0 \leq p \leq 1$）を計算する．ルータは到着パケットに対し，確率 p でパケット廃棄（または ECN パケットマーキング）を行い，エンドホストへ輻輳が通知される．

最も代表的な AQM アルゴリズムとして RED があげられる．RED は平均キュー長の大きさにともない p を設定し，p が固定値 max_p より大きい場合，$p = 1$ とすることでキュー溢れを防ぐ．しかし，max_p は固定値のためトラヒックパターンによっては対応できない場合がある．

これを解消する方式として，max_p を動的に設定する ARED がある．ARED は平均キュー長の目標範囲を設定し，平均キュー長が目標範囲を超えないように max_p を増減する．つまり，max_p には輻輳の程度が反映される．

同様にキュー長の目標を利用する方式として，REM があげられる．REM は目標キュー長 b^* を設定し，キュー長を b^* に近づけるよう p を算出することで入力トラヒックを安定化し，キュー溢れを防止する．

RTT 推定を利用した AQM として，Shyu-AQM[8] がある．この手法はすべてのコネクションに対する RTT 推定値を求め，到着パケットの所属するコネクションの RTT から p を算出する．Shyu-AQM では，コネクションすべての RTT を保持する必要がある．また，RTT 推定には通信開始時にしか利用できない手法[9] を使用するため，推定後のコネクションごとの RTT 値は更新されない．

3.　提案 AQM アルゴリズム

本章で，RTT 推定を導入して TCP プロトコルの輻輳制御を明示的に考慮した AQM を提案する．提案 AQM は，輻輳通知の効果が RTT 後に得られるという性質を考慮する．また，目標キュー長を設け，目標キュー長付近でキュー長を安定化させることにより，スループットも安定化させる，という特徴を持つ．

提案 AQM では，RTT 値とコネクション数を推定して使用する．通常ルータを通過するコネクションが N 本あるとき，RTT 値はコネクションごとに異なる．しかし，N 個の RTT 値を保持して輻輳制御を行うと空間計算量が $O(N)$ となるため，これを 1 変数で扱えるように，提案 AQM ではコネクション N 本分の RTT 代表値を使用する．代表値として 1 変数で表現することで N 個の RTT 値を保持する必要がなくなる．また，どの RTT 値を使うかを選択する機構が必要ない．反面，フローごとのステートを持たないので，RTT の異なるコネクション群に対して個別の輻輳制御を行えない．たとえば，RTT の大きなコネクションを優遇するなどといった制御は行えない．

提案 AQM は，以下の機能から構成される．

- RTT 代表値の取得法
- コネクション数 N の取得法
- 輻輳制御アルゴリズム

それぞれは独立して動作し，輻輳制御アルゴリズムでは随時更新された RTT 代表値と N を使用する．

なお，提案 AQM では，UDP など TCP 以外の通信プロトコルは帯域確保によって分離されているとして，TCP コネクションのみを対象として扱う．

3.1　RTT 代表値の取得法

提案 AQM で使用する RTT 代表値の取得法について説明する．提案手法では，まず到着パケットをランダムに選択して推定対象のコネクションを決め，1 コネクションの RTT 値を推定する．そして，1 コネクションの RTT 推定値を繰り返し取得し，それらから全コネクションの RTT 代表値を計算する．

以降の項では，「1 コネクションの RTT 値推定法」と「全コネクションの RTT 代表値計

算法」について説明する．

3.1.1　1 コネクションの RTT 推定法

本論文では 2 章で述べたセルフクロッキングベースの RTT 推定法を利用し，省リソース化とリアルタイム性を実現するための拡張を行う．

最初に推定対象として選ばれたパケットを P_{beg}，P_{beg} の到着時刻を t_{beg} とする．

t_{beg} 以降の到着パケットに対し，まずバースト開始時刻 t_0 の獲得を試みる．P_{beg} はバーストの途中である可能性が高いため，現在のバースト終了を待ち，次のバーストの開始時刻を t_0 として捕捉する．t_0 決定後は，バースト終了時刻 t_1 を求める．

t_0，t_1 取得後も観測を続け，続く 2 回目のバースト開始時刻 t_2 と終了時刻 t_3 を取得する．

t_2 が得られるとバースト間の時間間隔 $t_2 - t_0$ が算出可能となる．しかし，バーストが断片化している場合は $t_2 - t_0$ が RTT 値ではなく単に断片化した 2 バースト間の間隔となる．この可能性を排除することを目的として，バースト発生に周期性があるかを判定する．このために，次のバーストの開始時刻 t_4 を取得し，連続する 2 回のバーストを比較する．比較には以下の条件判定 2 つを使用する．

条件 1：同じような時間間隔か（$t_2 - t_0 \approx t_4 - t_2$）

条件 2：同じようなバースト長か（$t_1 - t_0 \approx t_3 - t_2$）

推定 RTT 値が条件を 2 つとも満たす場合，$t_4 - t_2$ を対象コネクションの RTT 推定値に決定する．なお，バースト判定の成功/失敗にかかわらず，判定後は対象コネクションの RTT 推定を終了し，別なコネクションの RTT 推定を行う．

t_0〜t_4 とバーストの関係について，時系列に従った到着パケットの図として**図 2** に示す．色付きの正方形は観測対象のコネクションのパケット，色なしの正方形は観測対象以外のコネクションのパケットで，判定成功時には $T_b = t_4 - t_2$ が RTT 推定値となる．

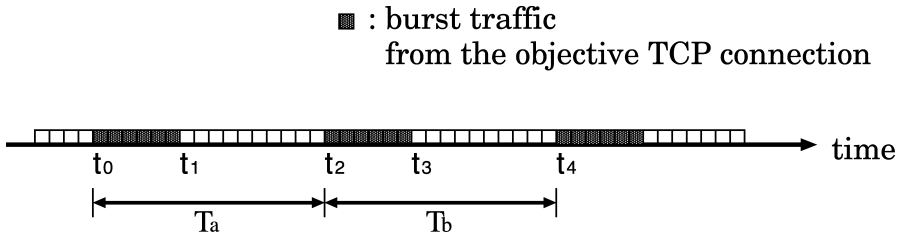


図 2　連続する 2 つのバーストにおける始点，終点の時刻
Fig. 2　The duration of two consecutive bursts.

本論文の RTT 推定法は，バースト判定において文献 7) を簡略化している．これにより推定 1 回の所要時間は RTT 値の 4 倍以下に収まり，さらに必要な記憶領域も観測時間に依存しない．反面，文献 7) の厳密な最大自己相関分析に比べ，本方式の推定精度は低下する．しかし，本推定法と似たアプローチで実験を行い，実ネットワーク上でエンドホストの RTT 値に近い推定値が得られることが文献 11) に報告されている．文献 11) では上記の条件 2 のみで判定を行うのに対し，本方式では条件 1 と条件 2 で判定を行って，より精度を向上させる．

上記の方式において，バーストの開始時刻と終了時刻は以下の手順で検出する．対象パケットがバーストの一部であるか否かを判別するために，本論文では時間間隔 ΔT を設定する．同一コネクションのパケットが ΔT 以内で観測され続ける期間をバーストと判断し，ΔT 以上経過しても次の対象パケットが観測されなければ，最後の対象パケット観測時刻でバーストが終了したと判断する．なお，本論文では $\Delta T = 10\,\mathrm{ms}$ と設定している．ΔT の妥当な値は測定対象のコネクションの持つ RTT に依存する．我々が実験により確認したところ，RTT が 60 ms，130 ms のコネクションに対しては RTT 周期のバーストが測定されたが，RTT が 10 ms のコネクションに対しては ΔT が大きすぎてバーストの終了時刻の計測に失敗した．また，RTT が 300 ms のコネクションに対しては，ΔT が小さすぎて，バーストの途中をバースト終了と誤認識し，バーストの判定に失敗した．対策として，RTT とスループットは反比例の関係にあるため，観測対象の入力レートを ΔT の決定に反映させることが有効と考えられる．この対策も含め，ΔT の決定法については現在改善を検討している．

3.1.2　全コネクションの RTT 代表値計算法

RTT 推定法によって対象コネクションの RTT 値が得られたら，N 本分の RTT 代表値に反映させる．対象コネクションの RTT 推定値 RTT_{est} の決定後，RTT 代表値 RTT_{rep} を以下の式 (1) で更新する．

$$RTT_{rep} = 0.9 \cdot RTT_{rep} + 0.1 \cdot RTT_{est} \tag{1}$$

式 (1) は取得した RTT 値の指数移動平均（Exponentially Weighted Moving Average: EWMA）を求めている．提案手法では，到着パケットをランダムに選択して推定対象を選ぶので，あるコネクション i が推定対象に選ばれる確率は，全入力レートに対する i の入力レートの割合になる．つまり，レートの大きいコネクションが測定対象になりやすい．したがって，式 (1) は式 (2) に示す入力レートの割合を考慮した RTT 値の重み付き平均をオンラインで求めようとするものになっている．

$$\widetilde{RTT_{rep}}(t) = \sum_{i=1}^{N} \frac{x_i(t)}{X(t)} \cdot RTT_i(t) \tag{2}$$

式 (2) において，t は現在の時刻，$x_i(t)$ はコネクション i のスループット，$X(t)$ は全コネクションのスループット合計値，$RTT_i(t)$ はコネクション i の RTT 値である．$\frac{x_i(t)}{X(t)}$ はコネクション i の入力レートの割合であり，重み付き平均の重みである．$\widetilde{RTT_{rep}}(t)$ はレートの大きなコネクションを重視した RTT 代表値となる．提案 AQM ではレートの大きなコネクションは輻輳を支配するという理由で，レートの大きなコネクションを重視しており，しかも到着パケットをランダムに選択するだけでこの機能を実現できている．

式 (1) で計算する RTT_{rep} はオンラインで計算するので，$\widetilde{RTT_{rep}}(t)$ 値とずれが発生する可能性がある．この精度については 4.2 節で検証を行う．

3.2　コネクション数の取得法

AQM の 1 つである SRED（Stabilized RED）では，アクティブフロー数を推定する方式が提案されている[12]．SRED では，フローごとのスループットの割合をゾンビリストと呼ばれるリストで表現し，リスト内フローのパケット検出率からフロー数を推定する．本論文ではこの方式を採用し，ゾンビリストを用いて得られるアクティブフロー数を TCP コネクション数 N とする．

3.3　輻輳制御アルゴリズム

現在時刻 t におけるスループットを X_t，キュー長を Q_t とするとき，輻輳制御アルゴリズムは以下の手順で現在の輻輳通知確率 p_t を生成する．

Step-1：時刻 t 時に，X_t と Q_t から RTT 後のキュー長 Q_{t+RTT} を見積もる．

Step-2：将来のキュー長を目標キュー長 b^* にするための目標スループット X^*_{t+RTT} を計算する．

Step-3：X^*_{t+RTT} を実現する輻輳通知確率 p_t を計算する．

以降では，各ステップの詳細について説明する．

Step-1：RTT が 3.1 節で述べた各コネクションの RTT 代表値のとき，時刻 t の輻輳通知の効果は平均的に時刻 $t + RTT$ に現れる．したがって，時刻 t の輻輳制御は時刻 $t + RTT$ の輻輳状態に基づいて考える．時刻 $t + RTT$ の輻輳状態として，RTT 後キュー長 Q_{t+RTT} を次式で算出する．

$$Q_{t+RTT} = Q_t + (X_t - C) \cdot RTT \tag{3}$$

C は出力リンク容量である．ここでは，時刻 $[t, t + RTT]$ 間のスループットが一定値 X_t であると近似する．式 (3) から計算した Q_{t+RTT} が大きければ，将来重度の輻輳が発生する可能性が高い．

Step-2：輻輳制御の目標として，キュー長を Q_{t+RTT} から時刻 $[t + RTT, t + 2RTT]$ 間で b^* にすることを目指す．ここで，Q_{t+2RTT} もまた式 (3) と同様に表現すると，X_{t+RTT} と Q_{t+RTT} との関係式 (4) が成立する．

$$Q_{t+2RTT} = Q_{t+RTT} + (X_{t+RTT} - C) \cdot RTT \tag{4}$$

ここで，目標とする $Q_{t+2RTT} = b^*$ とおくと，式 (4) より目標スループット X^*_{t+RTT} が次式で計算される．

$$X^*_{t+RTT} = C + \frac{(b^* - Q_{t+RTT})}{RTT} \tag{5}$$

式 (5) は，スループットが C と等しく，キュー長が b^* のとき定常状態となることを示している．

Step-3：Q_{t+RTT} と X^*_{t+RTT} から輻輳通知確率 p_t を計算する．

まず，$Q_{t+RTT} \leq b^*$ なら輻輳のない状態として $p_t = 0$ に設定する．以降では，$Q_{t+RTT} > b^*$ の場合について説明する．まず，現在のスループットが X_t のとき，TCP の輻輳ウィンドウ制御を考慮することで X_{t+RTT} のとりうる範囲を見積もる．N 本のコネクションがあるとき，時刻 t におけるコネクション i（$1 \leq i \leq N$）のスループットを $x_{i,t}$ とする．このとき，$x_{i,t}$ と X_t との間に以下の関係式が成立する．

$$\sum_{i=1}^{N} x_{i,t} = X_t \tag{6}$$

時刻 $t + RTT$ におけるコネクション i のスループット $x_{i,t+RTT}$ は次式 (7) となる．

$$x_{i,t+RTT} = \begin{cases} x_{i,t} + (MSS_i + IH) & \text{(輻輳通知なし)} \\ \frac{1}{2} x_{i,t} & \text{(輻輳通知あり)} \end{cases} \tag{7}$$

MSS_i はコネクション i の MSS 値，IH は IP ヘッダ長である．ここで，輻輳通知するコネクションの割合を α（$0 \leq \alpha \leq 1$）とし，すべてのコネクションの MSS が等しいと仮定すると，式 (6) と式 (7) から X_{t+RTT} が計算でき，次式で表される．

$$X_{t+RTT} = (1 - \alpha) \cdot (X_t + N \cdot (MSS + IH)) + \alpha \cdot \frac{1}{2} X_t \tag{8}$$

ルータがまったく輻輳通知を行わなければ $\alpha = 0$ となり，ルータが N 本のコネクションすべてに輻輳通知を行った場合には $\alpha = 1$ となる．このとき，X_{t+RTT} の最大値 $\max X_{t+RTT}$ と最小値 $\min X_{t+RTT}$ が式 (8) から計算され，次式 (9) となる．

$$\begin{aligned} \max X_{t+RTT} &= X_t + N \cdot (MSS + IH) \\ \min X_{t+RTT} &= \tfrac{1}{2} X_t \end{aligned} \tag{9}$$

式 (5) で算出した X^*_{t+RTT} は目標値なので，$\min X_{t+RTT}$ から $\max X_{t+RTT}$ までの範囲外の値をとることがある．$X^*_{t+RTT} \geq \max X_{t+RTT}$ の場合は，すべてのコネクションにいっさい輻輳通知をしなくても X_{t+RTT} が目標レートを下回ることを意味し，トラヒック量が少なすぎる状態である．$X^*_{t+RTT} \leq \min X_{t+RTT}$ の場合は，すべてのコネクションに輻輳通知しても X_{t+RTT} が目標レートを上回ることを意味し，トラヒック量が多すぎる状態である．両方とも，時刻 $t + RTT$ の輻輳制御では目標スループットを実現できないことを意味している．よって，この場合には p_t を式 (10) に従って決める．

$$p_t = \begin{cases} 0 & (\text{if } X^*_{t+RTT} \geq \max X_{t+RTT}) \\ 1 & (\text{if } X^*_{t+RTT} \leq \min X_{t+RTT}) \end{cases} \tag{10}$$

$\min X_{t+RTT} < X^*_{t+RTT} < \max X_{t+RTT}$ の場合は，次式 (11) で α を求める．

$$\alpha = \frac{\max X_{t+RTT} - X^*_{t+RTT}}{\max X_{t+RTT} - \min X_{t+RTT}} \tag{11}$$

前述のように，α は X^*_{t+RTT} を実現するために N 本のコネクションのうち輻輳を通知するコネクションの割合である．同時に，α は 1 つのコネクションが輻輳通知を受ける確率とも見なせるので，α と p_t との間に以下の関係が成立する．

$$\alpha = 1 - (1 - p_t)^{avgW_t} \tag{12}$$

ここで，$avgW_t$ は TCP コネクション N 本の平均ウィンドウサイズで，次式で計算する[⋆1,⋆2]．

$$avgW_t = \frac{X_t}{MSS + IH} RTT \cdot \frac{1}{N} \tag{13}$$

式 (13) は，RTT 期間で通信される総パケット数 ($\frac{X_t}{MSS+IH} RTT$) を N で割っているので，コネクション一本が RTT 期間に通信するパケット数となる．式 (12) では $avgW_t$ 個のパケットすべてが輻輳通知を受けない確率を 1 から引くことで α を算出する．以上より，$\min X_{t+RTT} < X^*_{t+RTT} < \max X_{t+RTT}$ の場合，p_t を式 (14) で算出する．

$$p_t = 1 - (1 - \alpha)^{\frac{1}{avgW_t}} \tag{14}$$

式 (8)，(9)，(13) の MSS については，ランダムに抽出した到着パケットの平均セグメントサイズとする．具体的には式 (15) で更新する．

$$MSS = 0.9 \cdot MSS + 0.1 \cdot MSS_{new} \tag{15}$$

MSS は MSS 平均値，MSS_{new} はランダム抽出された到着パケットのセグメントサイズである．

p_t を算出する流れを，**図 3** にまとめる．

```
Q_{t+RTT} = Q_t + (X_t − C) · RTT
IF  Q_{t+RTT} ≤ b*
    p_t = 0
ELSE
    X*_{t+RTT} = C + (b* − Q_{t+RTT}) / RTT
    maxX_{t+RTT} = X_t + N · (MSS + IH)
    minX_{t+RTT} = (1/2) X_t
    IF  X*_{t+RTT} ≥ maxX_{t+RTT}
        p_t = 0
    ELSEIF  X*_{t+RTT} ≤ minX_{t+RTT}
        p_t = 1
    ELSE
        α = (maxX_{t+RTT} − X*_{t+RTT}) / (maxX_{t+RTT} − minX_{t+RTT})
        avgW_t = (X_t / (MSS+IH)) RTT · (1/N)
        p_t = 1 − (1 − α)^{1/avgW_t}
```

図 3　p_t 算出の流れ
Fig. 3　Pseudo code for the p_t calculation.

4. 推定技術の精度検証

3.1 節で述べた RTT 推定の精度について，NS-2 シミュレータ上で評価を行う．ネットワークトポロジは**図 4** を使用し，伝搬遅延とコネクション数を変化させて RTT 推定法の精度検証を行う．図 4 の送信者 s_i（$1 \leq i \leq N$）はルータ r1，r2 を経由して受信者 d_i と

⋆1 本論文での $avgW_t$ の単位は「パケット」である．キュー長，スループット，MSS，IH はバイトを単位としている．
⋆2 $avgW_t$ は，RTT 推定時にバースト間の総パケット数をカウントする方法でも算出可能である．本論文では，式 (13) による値を利用した．

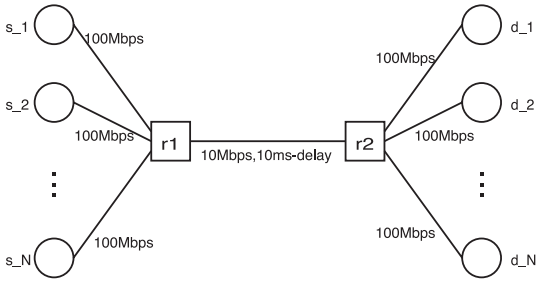


図 4　シングルホップのネットワークトポロジ
Fig. 4　Single-hop topology.

FTP 通信を行う．このため，r1，r2 間のリンクがボトルネックになる．実験の際，コネクションどうしが同時に通信を開始/終了しないように，最低でも 0.1 秒ずつ開始/終了の時刻をずらした．また，通信時のパケットサイズは 1 KB である．これらの通信に対し，ルータ r2 で RTT 推定を行う．

4.1　コネクションごとの RTT が均一な場合の RTT 代表値の精度検証

まず，コネクションごとの RTT が均一な状況において RTT 代表値の精度を検証し，本推定でエンドホストの RTT 値に近い RTT 値が得られるか評価する．

実験環境を以下に示す．

- 同じ 28 ms の往復伝搬遅延を持つ 25 本のコネクション（Group-A）と，84 ms の往復伝搬遅延を持つ 25 本のコネクション（Group-B）が存在する．
- 0〜50，50〜100 秒時に Group-A が通信を行い，50〜100 秒時に Group-B が通信を行う．

実験結果を**図 5** に示す．2 つの破線は，Group-A，B に属するコネクションの RTT 平均値，実線は RTT 代表値で，0.01 秒ごとに観測している．定常状態では RTT 代表値は通信中のグループの RTT 平均値よりやや低い．これら 2 値の誤差を以下の式で評価する．

$$error\_rate(t) = \frac{|\overline{RTT_i(t)} - RTT_{rep}(t)|}{\overline{RTT_i(t)}} \cdot 100 \quad (\%) \tag{16}$$

$error\_rate(t)$ は時間 t における誤差率である．$RTT_{rep}(t)$ は時間 t における RTT 代表値である．$\overline{RTT_i}(t)$ は時間 t における通信中のグループの RTT 平均値で，0〜50 秒時には Group-A，50〜100 秒時には Group-B，100〜150 秒時には Group-A のコネクションの RTT 平均値を表す．定常状態である 10〜50 秒時，60〜100 秒時，110〜150 秒時に式 (16) を適用した結果，誤差率は平均 2.51%となった．

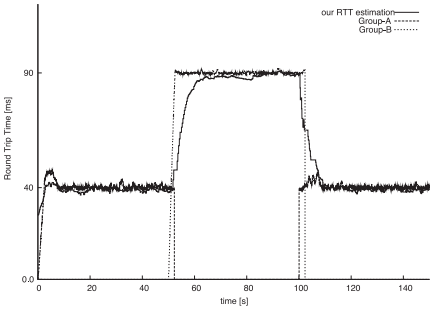


図 5　RTT が急に大きく増減する状況下の RTT 値の推移
Fig. 5　RTT transition when the RTTs of TCP connections vary drastically.

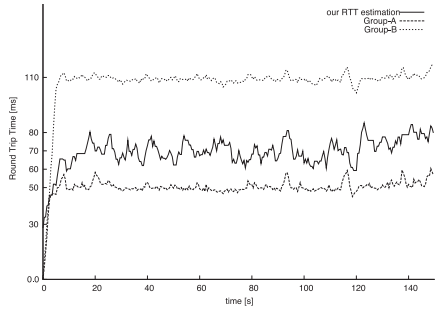


図 6　RTT が異なるコネクション混在時の RTT 値の推移
Fig. 6　RTT transition when the TCP connections have different RTTs.

また，トラヒックの変動後に RTT 代表値と通信中のグループの RTT 平均値との差が 10%未満になるまでに，50 秒時の変動後には 7.05 秒，100 秒時の変動後には 6.74 秒の時間を要した．このように，RTT 値変動時にも 10 秒未満で各グループの RTT に近い値に漸近する．

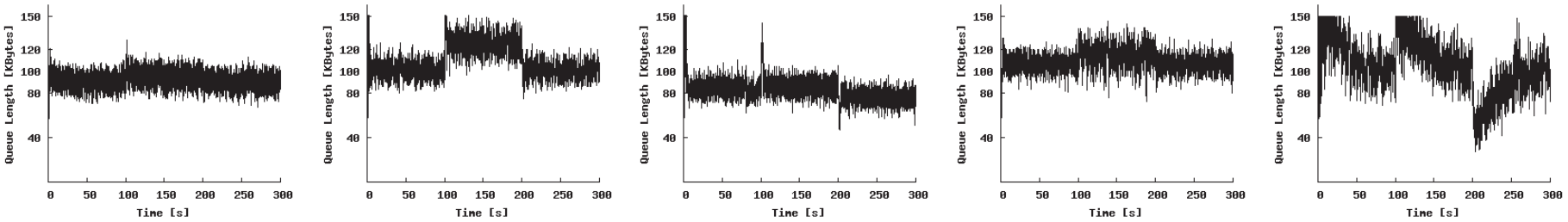
4.2　RTT が異なるコネクションが混在する場合の RTT 代表値の精度検証

さらに，異なる RTT 値を持つコネクションが混在する環境で，3.1 節で説明したように RTT 代表値がコネクションごとのスループットの割合を反映した値となるか検証する．混在の環境として，異なる伝搬遅延を持つ Group-A のコネクション 25 本と Group-B コネクション 25 本の計 50 本が，0〜150 秒間通信を行う．

実験結果を**図 6** に示す．2 つの破線は Group-A，B の RTT 平均値，実線は RTT 代表値で，0.5 秒ごとにプロットしたものである．Group-A の RTT は 50 ms 付近，Group-B の RTT は 110 ms 付近で安定している．RTT 代表値は 70 ms 付近を中心に 60〜80 ms の間で変動している．また，Group-A，B のスループットは約 0.24 Mbps，0.14 Mbps であった．これらの値を式 (2) に適用すると，

$$\widetilde{RTT_{rep}}(t) = \frac{0.24}{0.24+0.14} \cdot 50 + \frac{0.14}{0.24+0.14} \cdot 110 \tag{17}$$

となり，$\widetilde{RTT_{rep}}(t)$ は約 72 ms となる．RTT 代表値と $\widetilde{RTT_{rep}}(t)$ の誤差は平均 6.24%と小さく，コネクションごとのスループットの割合を反映した代表値が算出されていることが分



(a) 提案 AQM　(b) RED　(c) ARED　(d) Shyu-AQM　(e) REM

図 7　トラヒック量が増減する環境におけるキュー長の推移
Fig. 7　Queue length transition when traffic varies drastically.

かる．

5. 性能評価

シングルリンクとマルチホップの環境で提案 AQM を既存の AQM と比較し，性能を評価する．NS-2 シミュレータ上で実験を行い，いずれの実験でもエンドホストは FTP で通信を行い，輻輳通知の手段には ECN マーキングを使用する．また，4 章と同様に，コネクションどうしの通信開始時刻/終了時刻を最低でも 0.1 秒ずつずらしている．また，通信時のパケットサイズは 1 KB に固定する[⋆1]．

また，各実験のグラフデータでは，観測する項目によってプロット時間の粒度を以下のように設定している．

- キュー長は，NS 上で enqueue，dequeue 発生時にすべてプロットする．
- スループットは 0.5 秒ごとにプロットする．

5.1 シングルリンク時の性能評価

シングルリンクの実験には図 4 のネットワークトポロジを使用し，送信者 s_i ($1 \leq i \leq N$) はルータ r1，r2 を経由して受信者 d_i と通信を行う．実験環境の詳細を以下に示す．

- すべてのコネクションの往復伝搬遅延は 28 ms で，0〜100 秒，200〜300 秒時は 30 本のコネクションが通信を行い，100〜200 秒時には 50 本のコネクションが通信を行う．
- 提案 AQM，RED，ARED，Shyu-AQM，REM を r1，r2 に適用し，5 方式についてそれぞれ実験を行う．キューの容量は 150 KB で，各 AQM のパラメータとして，提案 AQM と REM の目標キュー長 b^* は 100 KB[⋆2]，RED と ARED の min_{th}/max_{th} はそれぞれ 40 KB/120 KB，Shyu-AQM は min_{th} を 40 KB に設定する．

表 1　変動時におけるキュー長の最小値と最大値
Table 1　Minimum and maximum queue length in changing states.

		提案 AQM	Shyu-AQM
100-120 s	最大値（KB）	127.9	139.4
	最小値（KB）	78.0	83.2
	最大値 − 最小値（KB）	50.0	56.2
200-220 s	最大値（KB）	106.1	131.0
	最小値（KB）	71.8	81.1
	最大値 − 最小値（KB）	34.3	49.9

この実験によって，100 秒，200 秒直後には，トラヒック変動の AQM の振舞いが見られる．それ以外の箇所では，トラヒックが変化しない定常状態での AQM の振舞いが見られる．

実験結果として，それぞれのキュー長の推移を**図 7** に示す．まず，トラヒック量変動時において，ARED，RED，REM ではキューが溢れたが，提案 AQM と Shyu-AQM はキュー溢れがなく，再送が発生しない．再送が発生しなかった提案 AQM と Shyu-AQM について，さらに変動時におけるキュー長の最小値と最大値を調べた（**表 1**）．キュー長最小値と最大値の差は提案 AQM のほうが Shyu-AQM より小さく，よりキュー長安定性に優れている．

⋆1 異なるパケットサイズを混在させて別途実験を行った際も，パケットサイズが輻輳に影響することなく本論文の実験と同様の結果が得られた．

⋆2 提案 AQM で b^* のとるべき値の範囲について求めるため，b^* が 0 に近い場合やキューの容量に近い場合で実験を行った結果，いずれもキュー長は b^* 付近で安定し，スループットにも影響が見られなかった．このため b^* 値は任意の値でよいが，本論文ではキューの容量の 2/3 以下の値を使用する．

表 2　定常状態におけるキュー長の COV
Table 2　Coefficient of variance of queue length in stable states.

	提案 AQM	RED	ARED	Shyu-AQM	REM
20-100 s	0.062	0.064	0.074	0.057	0.143
120-200 s	0.063	0.065	0.068	0.083	0.130
220-300 s	0.063	0.063	0.073	0.058	0.165

次に，定常状態においては，RED では 100〜200 秒時にキュー長が増加するが，提案 AQM，ARED，Shyu-AQM はトラヒック量が増減しても 100 KB 前後で安定している．さらに，キュー長の振動に関する評価として，定常状態でのキュー長の変動係数 COV（coefficient of variance）を計算する．COV は統計量 a の標準偏差 σ_a を算術平均 μ_a で割ったもので，次式 (18) で表される．COV_a が低いほど a の振動幅は小さい．

$$COV_a = \frac{\sigma_a}{\mu_a} \tag{18}$$

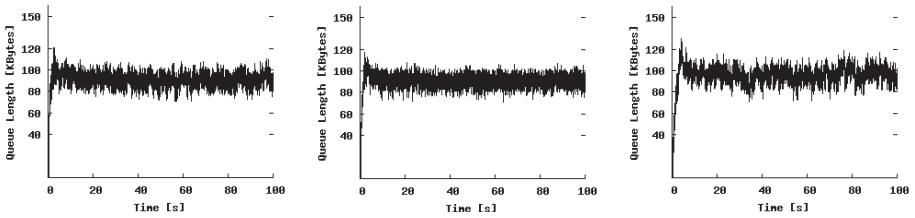
各 AQM のキュー長に対する COV を**表 2** に示す．Shyu-AQM では 120〜200 秒時に COV が高く，キュー長の振動幅が大きくなる．図 7 からもこれが読み取れ，トラヒック増加時（100〜200 s）のキュー長安定性が悪化する．対して，提案 AQM はすべての時間で一貫して COV が低く，キュー長の振動幅が小さい．

本実験ではグッドプット（再送を除いたスループット），再送量についても検証したが，グッドプットはいずれの AQM も 9.6 Mbps 付近で安定し，再送量はそれぞれのキュー溢れの回数に比例した．以上より，提案 AQM の高いキュー長安定性と，トラヒックの変化に対するロバスト性が確認できる．

5.1.1　RTT 推定精度が提案 AQM へ及ぼす影響

RTT 代表値は提案 AQM アルゴリズム内で重要なパラメータであるが，パッシブ RTT 推定により得られる値であるため，RTT 推定の精度によって影響を受ける．そこで，RTT 代表値の精度が提案 AQM へ及ぼす影響を検証した．5.1 節の実験のうち 0〜100 秒時を使用し，RTT 代表値を作為的に実際の値とかけ離れた固定値に設定した場合のキュー長への影響を検証した．実際の RTT 代表値は 98〜104 ms 付近で安定していたため，固定値には約半分の 50 ms と約 2 倍の 200 ms を設定して実験を行う．

実験の結果として，キュー長の推移を**図 8** に示す．図 8 より，固定値使用時も代表値使用時と同じ安定位置でキュー長を安定化する．平均キュー長と平均グッドプットの数値を**表 3** に示すが，大きな差は見られなかった．キュー長平均値はそれぞれ 6%未満，平均グッドプットはそれぞれ 0.01 Mbps の差にとどまった．



(a) RTT 代表値採用　(b) 50 ms 固定値　(c) 200 ms 固定値

図 8　提案 AQM の使用 RTT 値にともなうキュー長の推移
Fig. 8　Queue length transition of our AQM with different RTT value.

表 3　定常状態におけるキュー長平均値，平均グッドプット
Table 3　Average queue length and average goodput in stable state.

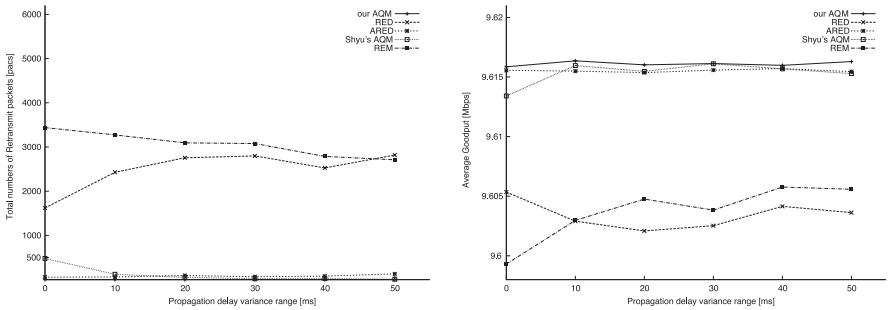
	平均キュー長（KB）	平均グッドプット（Mbps）
代表値使用	90.24	9.62
50 ms 固定値	90.30	9.61
200 ms 固定値	95.16	9.61

5.1.2　ランダム遅延リンク環境における性能評価

コネクションごとの伝搬遅延が均一でない環境において，提案 AQM と既存 AQM を比較する．図 4 のトポロジで，コネクションごとに s_i-r1 間（$1 \leq i \leq N$）の伝搬遅延を 0 ms〜Lms の範囲でランダムに設定する（L はパラメータ）．実験環境を以下に示す．

- 50 本のコネクションそれぞれがランダムな時刻に通信開始/停止を繰り返す．シミュレーション時間は 400 秒である．
- L が 0，10，20，30，40，50 ms の場合についてそれぞれ実験を行う．
- AQM に関するパラメータは 5.1 節と同様の値を用いる．

実験結果として，L と再送パケット数，平均グッドプットとの関係を**図 9** に示す．図 9 の横軸はそれぞれの実験の L 値である．図 9 (a) より，提案 AQM，ARED，Shyu-AQM は再送量が低く，中でも提案 AQM は全実験で再送が 0 である（このため，図 9 (a) には提案 AQM のプロットが表示されていない）．また，図 9 (b) より，提案 AQM が一番高い数値を出している．このように，コネクションごとの伝搬遅延が分散する環境でも提案 AQM は安定して高い性能を示す．



(a) 再送パケット数　　(b) 平均グッドプット

図 9　伝搬遅延分散と再送パケット数，平均グッドプットの関係
Fig. 9　Retransmit packets and average goodput for various propagation delay.

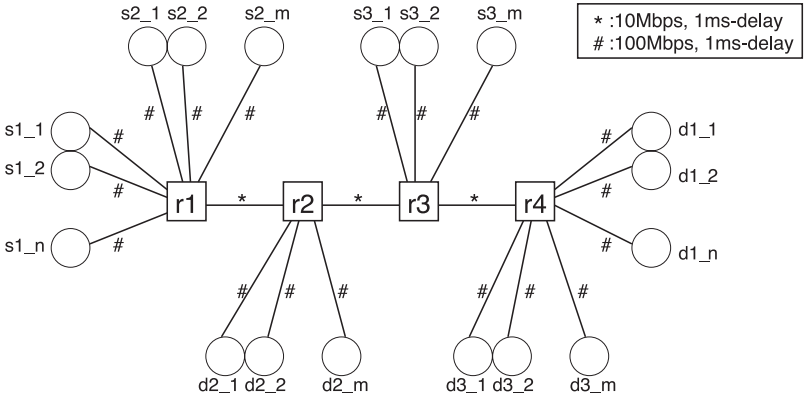


図 10　マルチホップのネットワークトポロジ
Fig. 10　A network topology for multi-hop experiment.

5.2　マルチホップ環境下における性能評価

本実験では，マルチホップ環境下における提案 AQM のキュー長安定性と，スループットについて検証する．実験には**図 10** のネットワークトポロジを使用する．複数のノードを経由するコネクションとして，s1_i（$1 \leq i \leq n_1$）は d1_i と通信を行い，通信時にルータ r1，r2，r3，r4 を経由する．クロストラヒックとして，s2_j（$1 \leq j \leq n_2$）がルータ r1，r2 を経由して d2_j と通信を行い，同様に s3_k（$1 \leq k \leq n_3$）がルータ r3，r4 を経由して d3_k と通信を行う．その他の実験環境の詳細を以下に示す．

- s1–d1 間，s2–d2 間，s3–d3 間のコネクション数はいずれも 10 本である（$n_1 = n_2 = n_3 = 10$）．すべてのコネクションが 0〜50 秒通信を行う．
- キューの容量は 100 KB，提案 AQM と REM の目標キュー長 b^* は 50 KB，RED の min_{th}/max_{th} はそれぞれ 20 KB/70 KB に設定する
- 各 AQM を r1，r2，r3，r4 すべてに適用し，観測は r3–r4 間で行う．

各 AQM のキュー長の推移を**図 11** に示す．定常状態において，いずれの AQM も 50 KB 付近に安定し，提案 AQM は他方式と比べ 10 KB 程度低めのキュー長で安定している．また，提案 AQM 以外の AQM では実験開始時にキュー溢れが発生し，再送が発生した．キュー長について COV を**表 4** に示す．表 4 より，提案 AQM の COV は最も小さいため，より少ない振動でキュー長を安定化させることが分かる．

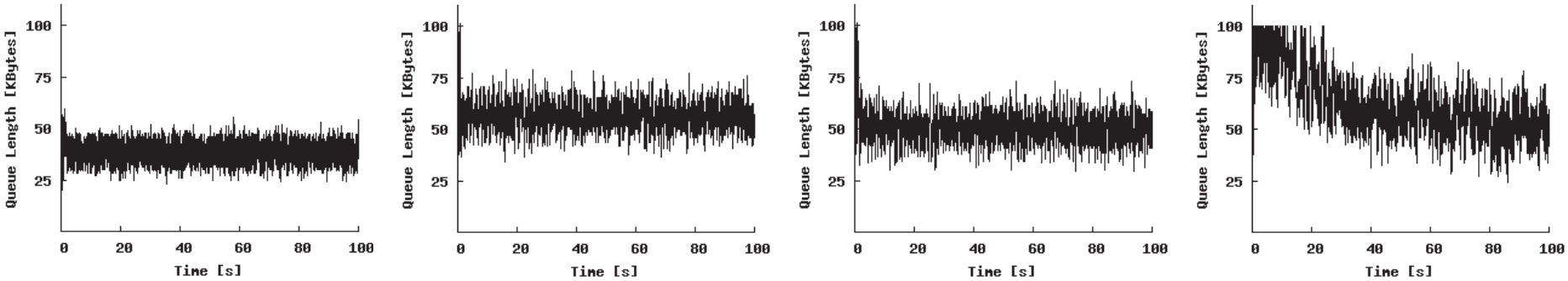
提案 AQM と ARED について，グッドプットの推移を**図 12** に，平均グッドプット，COV を**表 5** に示す．提案 AQM は ARED と同程度のグッドプット値（約 9.61 Mbps）を保つ．また，COV の低さから，提案 AQM が ARED よりもグッドプットの振動が少ないことが分かる．紙面の都合上本論文には記載しないが，RED と REM のグッドプット値は提案 AQM や ARED と同程度の値になるが，より激しく振動した．

以上より，提案 AQM がマルチホップ環境でもグッドプットを下げずにキュー長を安定させ，既存方式と同等以上の性能を示すことが考察される．

6. むすび

本論文では，TCP コネクションの RTT 値を明示的に輻輳制御アルゴリズムに組み込んだ AQM を提案した．AQM で RTT 値を使用するために，コネクション N 本分の RTT 値を RTT 代表値として 1 変数で扱うことで，オンラインかつフローステートレスな RTT 推定法を実現した．また，シミュレーションにより提案 AQM のキュー長，スループットの安定性と，トラヒック変化に対するロバスト性を示した．リアルタイム，省スペースな RTT 推定法を導入した AQM が初めて実例として示されたことにより，今後 RTT 推定と AQM 両分野の発展が期待される．

今後，RTT 推定法においてコネクションのバースト判定に用いた時間間隔 ΔT を，推定対象に応じて適応的に決定するように改良し，バースト判定のさらなる精度向上を目指す．また，Reno 以外の TCP プロトコルに対する評価や，UDP 混在時の公平性の評価を検討している．

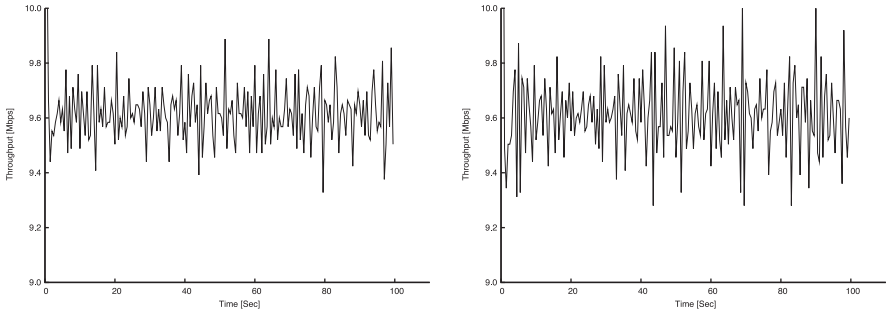


(a) 提案 AQM　(b) RED　(c) ARED　(d) REM

図 11　マルチホップ環境下におけるキュー長の推移
Fig. 11　Queue length transition in multi-hop environment.

表 4　マルチホップ環境におけるキュー長の COV
Table 4　Numeral analysis results of goodput in multi-hop environment.

	提案 AQM	RED	ARED	REM
COV	0.101	0.106	0.123	0.181



(a) 提案 AQM　(b) ARED

図 12　マルチホップ環境下におけるグッドプットの推移
Fig. 12　Goodput in multi-hop environment.

謝辞　本研究は科学研究費 若手研究（B）課題番号 17700054 の支援を受けて実施された．

表 5　マルチホップ環境における提案 AQM と ARED のグッドプット解析結果
Table 5　Numeral analysis results of goodput in multi-hop environment.

	提案 AQM	ARED
平均値（Mbps）	9.6155	9.6141
COV	0.011	0.014

参　考　文　献

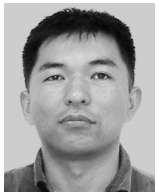
1) 星原隼人，古賀久志，北村　浩，渡辺俊典：将来の輻輳状態の予測に基づくアクティブキュー管理手法の提案，電子情報通信学会技術研究報告，情報ネットワーク，Vol.105, No.279, pp.25–30 (2005).
2) Hoshihara, H., Koga, H. and Watanabe, T.: A New Stable AQM Algorithm Exploiting RTT Estimation., *Proc. IEEE LCN'06*, pp.143–150 (2006).
3) Floyd, S. and Jacobson, V.: Random Early Detection Gateways for Congestion Avoidance, *IEEE/ACM Trans. Networking*, Vol.1, No.4, pp.397–413 (1993).
4) Floyd, S., Gummadi, R. and Shenker, S.: Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management, Technical report, ICSI (2001).
5) Athuraliya, S., Li, V.H., Low, S.H. and Yin, Q.: REM: Active Queue Management, *IEEE Network*, Vol.15, pp.48–53 (2001).
6) Feng, W., Shin, K., Kandlur, D. and Saha, D.: The BLUE Active Queue Management Algorithms, *IEEE/ACM Trans. Networking*, Vol.10, No.4, pp.513–528 (2002).
7) Veal, B., Li, K. and Lowenthal, D.: New Methods for Passive Estimation of TCP

Round-Trip Times, *Proc. Passive and Active Measurement Workshop* (2005).
8) Shyu, M.L., Chen, S.C. and Ranasingha, C.: Router Active Queue Management for Both Multimedia and Best-Effort Traffic Flows, *Proc. IEEE ICME'04*, pp.451–454 (2004).
9) Jiang, H. and Dovrolis, C.: Passive Estimation of TCP Round-Trip Times, *ACM Computer Communication Review*, Vol.32, pp.75–88 (2002).
10) Jaiswal, S., Iannaccone, G., Diot, C., Kurose, J. and Towsley, D.: Inferring TCP Connection Characteristics Through Passive Measurements, *Proc. IEEE INFOCOM*, pp.1582–1592 (2004).
11) 大坐畠智，川島幸之助：IP ヘッダの ID フィールドを用いた TCP 通信中の利用可能帯域推定法，電子情報通信学会技術研究報告，ネットワークシステム，Vol.105, No.278, pp.95–98 (2005).
12) Ott, T.J., Lakshman, T.V. and Wong, L.: SRED: Stabilized RED, *Proc. INFOCOM*, pp.1346–1355 (1999).

(平成 21 年 5 月 11 日受付)
(平成 21 年 10 月 2 日採録)

星原　隼人

平成 16 年電気通信大学情報工学科卒業．平成 18 年同大学大学院情報システム学研究科修士課程修了．現在，博士課程在学中．パッシブ推定と通信品質保証に関する研究に従事．



古賀　久志（正会員）

平成 7 年東京大学大学院理学系研究科修士課程修了．同年（株）富士通研究所入社．平成 14 年東京大学大学院理学系研究科博士課程修了．博士（理学）．平成 15 年電気通信大学大学院情報システム学研究科講師．現在，電気通信大学大学院情報システム学研究科准教授．専門は離散アルゴリズム，スケジューリングアルゴリズム．近年は，データ工学，構造的パターン認識の研究にも従事．

渡辺　俊典（正会員）

昭 46 年東京大学工学部航空学科卒業．同年日立製作所入社，中央研究所，システム開発研究所．経営生産，LSI 設計，非線形最適化，学習機械，並列分散推論（ICOT プロジェクト）等，諸システムの開発に従事．平成 4 年より電気通信大学大学院情報システム学研究科教授．工学博士．電子情報通信学会，IEEE 各会員．専門はメディアデータ自動解析および情報システム表現・解析．