社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE

ハイブリッド型 TCP 輻輳制御の特性解析と映像配信応用

甲藤 二郎　藤川 知樹　蘇 洲

早稲田大学 理工学術院 〒169-8555 東京都新宿区大久保 3-4-1
E-mail: {katto, tomoki, suzhou}@katto.comm.waseda.ac.jp

あらまし 本稿では、ハイブリッド型 TCP 輻輳制御の特性解析について報告を行う。近年、TCP-Reno に代表されるロスベース（ウィンドウベース）の輻輳制御と、TCP-Vegas に代表される遅延ベース（レートベース）の輻輳制御を組み合わせたハイブリッド型の TCP 輻輳制御方式の提案が盛んに行われている。しかし、その解析モデルの検討は決して十分に行われているとは言い難いため、本稿では、その性能評価とパラメータ設定を可能とするための簡易な解析モデルの構築を試みる。そして、その解析結果として、ハイブリッド型 TCP 輻輳制御は、パケットロスが頻繁に発生する環境下では（ロスベース型の輻輳制御に比べて）パケットロスに対する耐性を有し、安定したチャネル環境下では TCP-Reno に対する良好な親和性を提供することを示す。さらには、解析モデルの応用として、遅延ベース輻輳制御に基づく TCP フレンドリレート制御方式を提案し、この手法が、特にパケットロスが頻繁に発生する環境下において、従来よりも優れたスループット特性を実現できることを示す。

キーワード TCP 輻輳制御、特性解析、TCP フレンドリレート制御

Performance Analysis of Hybrid TCP Congestion Control and Its Application to Visual Content Distribution

Jiro KATTO　Tomoki FUJIKAWA　Su ZHOU

Dept. of Computer Science, Waseda University, 3-4-1 Okubo, Shinjuku, Tokyo 169-8555, Japan
E-mail: {katto, tomoki, Suzhou}@katto.comm.waseda.ac.jp

Abstract This paper presents mathematical framework for hybrid TCP congestion control. Recently, several hybrid TCP congestion control methods have been proposed, in which loss-based features and delay-based features of the TCP congestion control are combined in a smart way. However, since its analytical performance has not yet been clarified sufficiently, we try to develop simple mathematical framework to evaluate and tune its performance. The results prove efficiency and robustness of the hybrid congestion control when random packet losses frequently happen, and also prove its good friendliness to TCP-Reno when stable channels are available. We furthermore present delay-based TCP friendly rate control, which provides better throughput performance than classical TFRCs.

Keyword TCP Congestion Control, Performance Analysis, TCP Friendly Rate Control

1. はじめに

近年、帯域遅延積の大きい広帯域ネットワークを対象に、従来の TCP-Reno[1]に代わるさまざまな輻輳制御方式(TCP Variants)が提案されている。これらは大きく、(a) ロス(ウィンドウ)ベース方式、(b) 遅延(レート)ベース方式、(c) 両者のハイブリッド方式、に分類される。ロスベース方式は、TCP-Reno と同じく、パケットロスを輻輳検出の手段に用いる一方で、いわゆる AIMD 制御を広帯域ネットワークに適した制御に変更するものである。遅延ベース方式は、RTT (R を輻輳検出手段に使用するもので、パケットロスの発生前に初期輻輳を検知できることから、単独で使用する場合はロスベース方式よりも優れたスループット性能を実現できる。しかし一方で、ロスベース方式と共存させた場合、輻輳制御メカニズムが自発的にレートを下げてしまう欠点が知られている。そこで、ロスベース方式と遅延ベース方式の利点を組み合わせた各種のハイブリッド方式の提案が行われている。

TCPW (TCP-Westwood) [2,3,4] は、それ自身はハイブリッド動作をするものではないが、その定式化が後述する多くのハイブリッド方式に影響を与えている。例えば、セッション中に帯域推定 (BE: Bandwidth Estimation) やレート推定 (RE: Rate Estimation) を行い、過度のウィンドウ減少などの問題を回避する。Gentle HS-TCP [5] と TCP-Africa [6] は、先駆的なハイブリッド輻輳制御方式であり、計測 RTT の値に応じて、TCP-Reno と高速モード (High-Speed TCP) を切り替える。CTCP [7] は、ロスモードと遅延モードの二種類の輻輳ウィンドウを用意し、ロスモードのウィンドウが遅延モードのウィンドウを追い越した時点で、二つのモードを切り替える。ARENO [8] は、TCPW のメカニズムを拡張し、RTT 値の関数として連続的な輻輳ウィンドウ制御を行う。CTCP と ARENO は、共に空き帯域がある時は積極的に輻輳ウィンドウを増大し、RTT が増加すると TCP-Reno として振る舞うことで、スループット性能と親和性の両立を実現している。YeAH-TCP [9] と TCP-Fusion [10] はこれらをさらに拡張したもので、輻輳ウィンドウの増減方法やバッファ中の滞留パケット数の制御に工夫を加えている。

ハイブリッド輻輳制御は、ロスベースと遅延ベースの制御の切り替えが正確に行われる限りにおいて、優れたスループット性能 (efficiency) を実現し、かつ TCP-Reno との親和性 (friendliness) も保証する。近年は、この他に RTT 公平性 (fairness) や小規模バッファ (small buffer)への対応も求められるが[11]、非常に望ましい性質である。その一方で、ハイブリッド輻輳制御の理論解析手段は決して十分に整備されているとは言

い難く、現状では、多数存在する制御パラメータを、シミュレーションを通じてヒューリスティックに決めざるを得ない。[5][8] には解析評価手法も示されてはいるが、ハイブリッド輻輳制御の汎用的な評価に適しているとは言い難い。

TCP 輻輳制御の解析は、TCP-Reno の解析モデル [12,13] の提案以来、さまざまな提案が行われ、TCP-Vegas [14]、FAST-TCP [15,16]、TCPW [17] などへの拡張や精緻化も図られている。特に[16]では、TCP 輻輳制御の挙動を、バッファリングによる超過レートの平滑化に着目する積分リンクモデル (Integrator link model) と、セルフクロッキングによる超過レートの平滑化に着目する静的リンクモデル (Static link model) に分類し、さらには両者を統合した結合リンクモデル (Joint link model) を提案している。ただし、この検討は非常に精緻な一方で、対象がリンク帯域を使い切る場合に限定されるように見える。

一方、TCP 解析モデルのマルチメディア通信応用として、TCP フレンドリレート制御 (TFRC: TCP Friendly Rate Control) が知られている。これは、エンドホストが観測可能なパケットロス率と RTT を用いて、TCP-Reno の輻輳制御に等価なレートを計算する [18, 19,20]。特に VTP (Video Transport Protocol) [20] は、TCPW 的なモデルに基づく定式化が行われ、従来の TFRC 方式よりも良好な特性が示されている。

本稿は以下のように構成される。2 章では、筆者らが以前に提案したハイブリッド輻輳制御方式である TCP-Fusion について説明する。3 章では、はじめにロスベース、遅延ベース、ハイブリッドの理想的な挙動に関するモデル化の検討を行い、続いて TCP-Fusion 方式のモデル化の検討を行う。さらには、遅延ベース方式の解析モデルに基づく、新たな TFRC 方式の検討を行う。最後に 4 章でまとめを述べる。

2. TCP-Fusion [10]

図 1 に、ルータのバッファサイズがリンクの帯域遅延積 (BDP: Bandwidth-Delay Product) よりも小さい条件化で、別の TCP-Reno フローと競合している場合の TCP-Fusion の輻輳制御の挙動を示す。他のハイブリッド方式と同様に二つの輻輳ウィンドウ (cwnd と reno_cwnd) を保持し、reno_cwnd は競合する TCP-Reno の動作をエミュレートする。

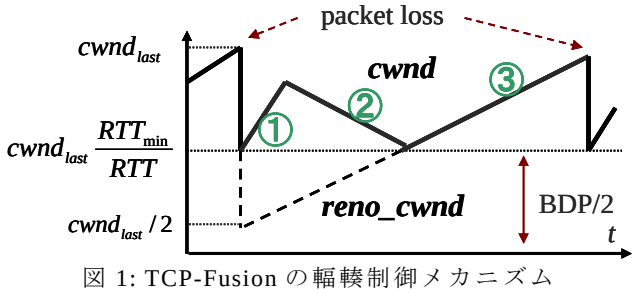


図 1: TCP-Fusion の輻輳制御メカニズム

図 1 でパケットロス発生後、reno_cwnd はウィンドウサイズを半分にするが、cwnd は TCPW のメカニズムに従い、次式に従ってウィンドウサイズを減少する。

$$cwnd_{new} = \max\left(\frac{RTT_{\min}}{RTT}cwnd_{last}, \frac{cwnd_{last}}{2}\right) \quad (1)$$

ここで $cwnd_{new}$ はパケットロス発生時のウィンドウサイズ、RTT_{mn} はセッション中に観測された RTT の最小値 (途中でバッファされずに送受信された RTT 値) を示す。(1)式右辺第一項により、ウィンドウサイズを過剰に減少することが回避され、また第二項により、バッファサイズが BDP よりも大きい (パケットロス後も即座にバッファリングされる) 場合は、reno_cwnd の値が設定される。

その後の輻輳回避フェーズは、次式に従って、TCP-Vegas に類似した遅延モードと、reno_cwnd に従ったロスモードのハイブリッド制御を行う。

$$cwnd_{new} = \begin{cases} cwnd_{last} + W_{inc}/cwnd_{last}, & \text{if } diff < \alpha \\ cwnd_{last} + (-diff + \alpha)/cwnd_{last}, & \text{if } diff > 3*\alpha \\ cwnd_{last}, & \text{otherwise} \end{cases} \quad (2)$$

$$\text{cwnd}_{\text{new}} = reno\_cwnd, \text{ if } cwnd_{new} < reno\_cwnd$$

このとき、$diff$ は次式で定義される変数であり、

$$diff = cwnd \cdot \frac{RTT - RTT_{\min}}{RTT} \quad (3)$$

バッファ内のパケット数の見積りを与える。この値が α よりも小さい場合は空き帯域があると判断し、W_{inc} で cwnd を急速に増加させる(図中の①)。逆に $3*\alpha$ よりも大きい場合は初期輻輳と判断し、cwnd を減少させる(図中の②)。それ以外の場合は安定状態と判断し、cwnd の値を保持する(図中省略)。そして、reno_cwnd が cwnd に追い付いた場合は、次のパケットロスが発生するまで、TCP-Reno として動作させる(図中の③)。

以上の手順で、二つの変数 α と W_{inc} は性能に影響を与える重要なパラメータとなるが、これらの具体的な値は以下のように設定する。まず、エンドホストの TCP スタックの検出可能な最小時間を $D_{\min}$ とすると、ルータが保持すべき最小のバッファサイズ G は次式で与えられる (この値を下回ると、エンドホストが検出する RTT は、パケットがバッファされるか否かに関わらず、常に RTT_{mn} になる)。

$$G = \frac{B*D_{\min}}{PS} \quad (4)$$

ここで B はリンク帯域幅、PS は送信パケットサイズであり、B は TCPW-BE のメカニズムを用いて推定する。一方、ボトルネックリンクを N 個の TCP フローが共有している場合を考える。この場合、個々のフローが α 個のパケットをバッファ内に保持しようとするが、これがバッファの総容量を超えるべきではない($G \geqq N*\alpha$)。よって、α を次式で設定する。

$$\alpha = \frac{G}{N} = \frac{(B/N)*D_{\min}}{PS} \approx \frac{RE*D_{\min}}{PS} \quad (5)$$

ここで RE は各フローが共有すべき公平レートであり、TCPW-RE のメカニズムを用いて推定する。一方、cwnd を増加させ、ちょうど帯域を使い切ったところで、さらに W_{inc} 個のパケットを増加させる状況を考える。このとき、W_{inc} 個の増加がバッファオーバーフローを招くことは望ましくなく、次式に従って W_{inc} を設定する。

$$W_{inc} \leq G = \frac{B*D_{\min}}{PS} \quad (6)$$

3. 特性解析

3.1. 理想モデルの特性比較

本節では、ロスベース、遅延ベース、ハイブリッドの三つの輻輳制御方式の理想的な挙動 (モデル) を定義し、解析的な特性比較を行う。

3.1.1. シングルフローモデル

図 2 に示す接続モデルを考える。ここでは送受信ホストが回線を占有し、中間のリンクがボトルネックになっている。まず、(バッファオーバーフローによる) パケットロス率を p、パケットロス発生時の cwnd 値を w とすると、TCP-Reno の輻輳回避の振る舞いから

$$p = \frac{8}{3w^2} \tag{7}$$

が成立することが知られている[12,13]。このロス率 p は、無線環境や RED ルーターにおけるランダムロスにも適用可能との指摘があり[12]、以下、本稿でも同様の仮定を行う(強引な仮定ではあるが、オーバーフローとランダムロスを一つのパラメータ w で表す)。さらに、帯域遅延積に相当するパケットの個数を W で表す。

ここで、ロスベース、遅延ベース、ハイブリッドの輻輳回避の挙動を以下のように定義する。

- ロスベース：RTT ラウンドあたり cwnd を一つ増加し、パケットロス時には cwnd を半分にする (TCP-Reno)
- 遅延ベース：RTT の増加を招かないまま、BDP を完全に使い切る。
- ハイブリッド：ロスベースと遅延ベースの二つのウィンドウを保持し、前者が大きい場合はロスベースとして、後者が大きい場合は遅延ベースとして動作する。

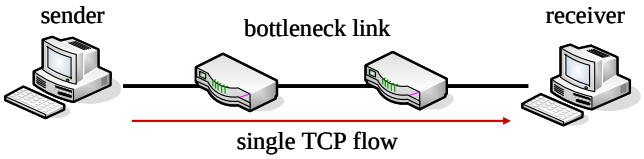


図 2: 接続トポロジー (シングルフローモデル)

このモデルの下で、三方式の挙動は、変数 w と W の関係に従い、図 3 に示す三通りに分類される。

(i) W<w/2 の場合：ルータバッファが BDP よりも大きく、パケットが常にバッファリングされる。
(ii) w/2≤W<w の場合：ルータバッファが BDP よりも小さく、ロスベース方式で cwnd を半減後、空き帯域が生じる。
(iii) w≤W の場合：パケットロス率が大きく、ルータに溜まる前にロスが発生する。

仮定に従い、遅延ベース方式は常に BDP を使い切るように動作する。ハイブリッド方式は、(a)の場合はロスモードで、(c)の場合は遅延モードで動作し、(b)の場合はロスモードと遅延モードを適応的に切り替える。一方、RTT の変化に着目すると、パケットがバッファリングされない間は RTT_{mn} のまま変わらず、バッファリングされるとその滞留量に応じて RTT が増加する。表 1 は、三方式それぞれについて、一回の輻輳回避ラウンドあたりの送信パケット数とラウンド経過時間をまとめたものである。TCP ではまた、パケットロスの検出後、パケットの再送を行うが、再送パケットがさらに連続して廃棄されるとタイムアウトが発生する。これは、ロスベース方式の場合、次式で推定される[13]。

$$t_{RTO,loss} = T_0 \cdot \left(1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6\right)/(1-p) \tag{8}$$

ここで T_0 は TCP の再送タイマ値である。遅延ベース方式とハイブリッド方式の場合は、表 1 に示した送信パケット数(K としている)の比として、

$$t_{RTO,delay} = \frac{K_{delay}}{K_{loss}} \cdot t_{RTO,loss} \tag{9}$$

に従ってタイムアウトのペナルティを課す。最終的に、表 1 に示した送信パケット数とラウンド経過時間、ならびに上記のタイムアウト時間から

$$\frac{transmitted\ packets}{elapsed\ time\ +\ timeout\ penalty} \tag{10}$$

によって各方式のスループットを推定する。

3.1.2. 競合フローモデル

図 4 に示す接続モデルを考える。この場合も同様に、変数 w と W の関係に従って、図 5 に示す三通りに分類することができる。ただし、ここでは遅延ベース方式は考えず、また w は二フロー競合であることを考慮して、半分の値にスケーリングしている。

(i) W<w の場合：ルータバッファが BDP よりも大きく、パケットが常にバッファリングされる。
(ii) w≤W<2w の場合：ルータバッファが BDP よりも小さく、ロスベース方式で cwnd を半減後、空き帯域が生じる。
(iii) 2w≤W の場合：パケットロス率が大きく、ルータに溜まる前にロスが発生する。

ハイブリッド方式は、(a)の場合はロスモードで、(c)の場合は遅延モード(BDP とロスベース方式の差を使い切るモード)で動作し、(b)の場合はロスモードと遅延モードを適応的に切り替える。タイムアウトのペナルティは前節と同様であり、表 2 に示す送信パケット数をラウンド経過時間、ならびにタイムアウト時間から、(10)式に従ってスループットを推定する。

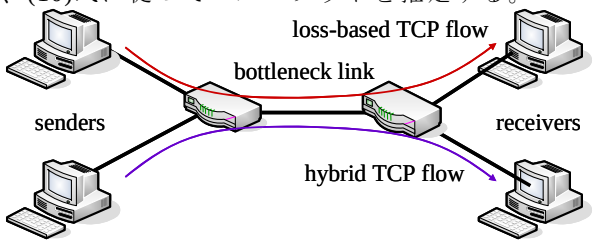
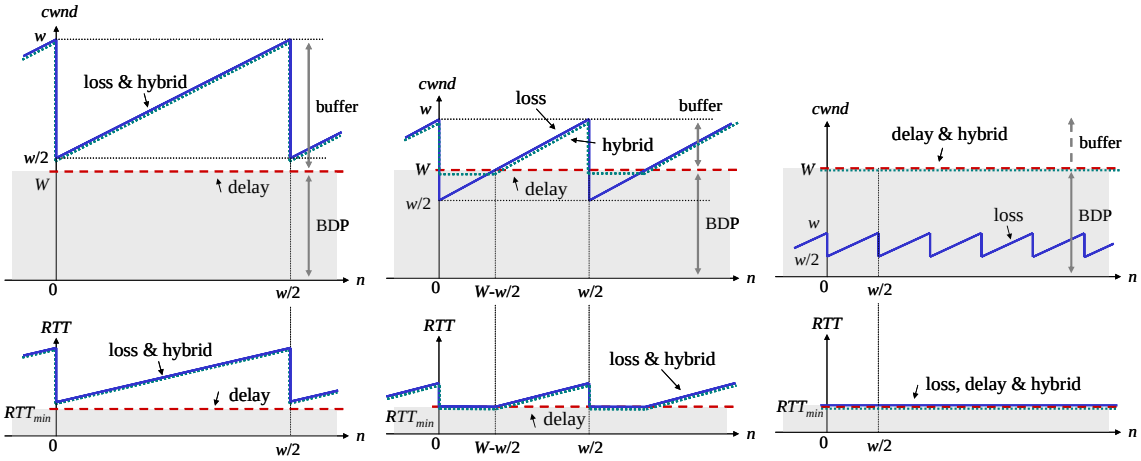


図 4: 接続トポロジー (競合フローモデル)

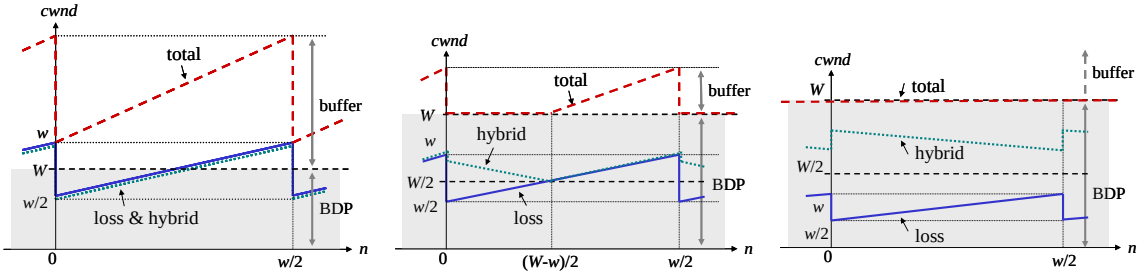
3.1.3. 実験結果

エンドホストとルータのリンク速度を 1Gbps、ルータ間(ボトルネックリンク)のリンク速度を 100Mbps、エンドホスト間の RTT を 40ms (アクセスリンク遅延 1ms、ボトルネックリンク遅延 18ms)、とした場合の評価結果を図 6 に示す。ここではまた、ns-2 [21] を用いて、TCP-Fusion、ARENO、CTCP、FAST-TCP、TCP-Reno (SACK オプション) について行ったシミュレーション結果も併記する。競合フローモデルの場合は、各 TCP フローと TCP-Reno を競合させた場合のスループットを示しており、TCP-Reno の結果は、TCP-Reno 二本を競合させた場合の一方の評価結果である。また図中の縦の破線二本は、変数 w と W の関係に従って定まる三分類の境界を示している(左から(i)、(ii)、(iii))。



(i) $W < w/2$　　(ii) $w/2 \le W < w$　　(iii) $w \le W$

図 3: 輻輳ウィンドウと RTT の挙動 (シングルフローモデル)



(i) $W < w$　　(ii) $w \le W < 2w$　　(iii) $2w \le W$

図 5: 輻輳ウィンドウの挙動 (競合フローモデル、RTT は省略)

表 1: 輻輳回避ラウンドあたりの送信パケット数と経過時間 (シングルフローモデル)

TCP	CA round	(i) $W < w/2$	(ii) $w/2 \le W < w$	(iii) $w \le W$
Loss	transmitted packets	$\frac{3}{8}w^2$	$\frac{3}{8}w^2$	$\frac{3}{8}w^2$
	elapsed time	$\frac{1}{2}w \cdot RTT_{\min} + \frac{1}{8}(3w^2 - 4wW) \cdot \frac{PS}{B}$	$\frac{1}{2}w \cdot RTT_{\min} + \frac{1}{2}(w-W)^2 \cdot \frac{PS}{B}$	$\frac{1}{2}w \cdot RTT_{\min}$
Delay	transmitted packets	$\frac{1}{2}w \cdot W$	$\frac{1}{2}w \cdot W$	$\frac{1}{2}w \cdot W$
	elapsed time	$\frac{1}{2}w \cdot RTT_{\min}$	$\frac{1}{2}w \cdot RTT_{\min}$	$\frac{1}{2}w \cdot RTT_{\min}$
Hybrid	transmitted packets	$\frac{3}{8}w^2$	$\frac{1}{2}w \cdot W + \frac{1}{2}(w-W)^2$	$\frac{1}{2}w \cdot W$
	elapsed time	$\frac{1}{2}w \cdot RTT_{\min} + \frac{1}{8}(3w^2 - 4wW) \cdot \frac{PS}{B}$	$\frac{1}{2}w \cdot RTT_{\min} + \frac{1}{2}(w-W)^2 \cdot \frac{PS}{B}$	$\frac{1}{2}w \cdot RTT_{\min}$

表 2: 輻輳回避ラウンドあたりの送信パケット数と経過時間 (競合フローモデル)

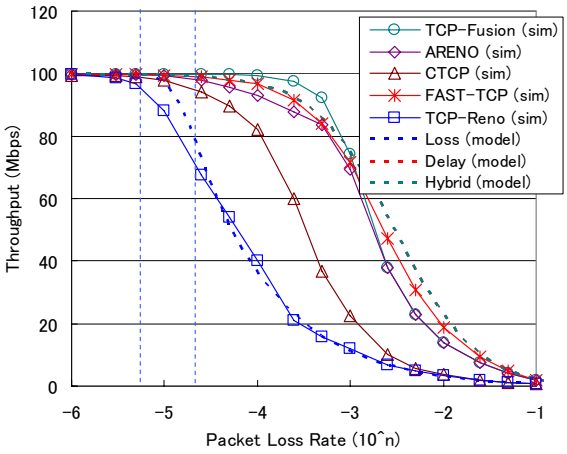
TCP	CA round	(i) $W < w$	(ii) $w \le W < 2w$	(iii) $2w \le W$
Loss	transmitted packets	$\frac{3}{8}w^2$	$\frac{3}{8}w^2$	$\frac{3}{8}w^2$
Hybrid	transmitted packets	$\frac{3}{8}w^2$	$\frac{3}{8}w^2 + \frac{1}{4}(W-w)^2$	$\frac{1}{2}w \cdot W - \frac{3}{8}w^2$
(common)	elapsed time	$\frac{1}{2}w \cdot RTT_{\min} + \frac{1}{4}w(3w-2W) \cdot \frac{PS}{B}$	$\frac{1}{2}w \cdot RTT_{\min} + \frac{1}{4}(2w-W)^2 \cdot \frac{PS}{B}$	$\frac{1}{2}w \cdot RTT_{\min}$

図 6 において、遅延ベースとハイブリッドの理論値は、(iii)の区間(右の区間)では完全に重なっている。この結果から、理論値はほぼシミュレーション値に対応しており、提案モデルの有効性が実証される。また、FAST-TCP は遅延ベースの例として用いたが、低ロス時の親和性が低いことなどもわかる。

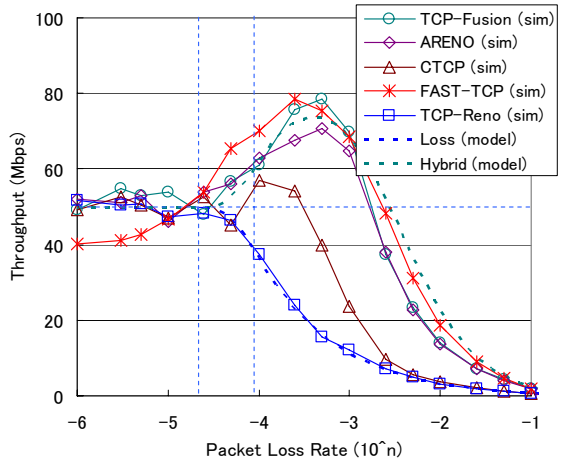
3.2. TCP-Fusion 拡張

前節のハイブリッド方式は、解析の容易さを重視して挙動の簡素化を図っていたが、本節では、TCP-Fusion における変数 α と W_{inc} を反映したモデル作成を試みる。

図 7 は、競合フローが存在しない場合の、パケットロス発生直後の TCP-Fusion の挙動を示している。まずパケット再送のために 1 RTT ラウンドが費やされ、(1)式に従って BDP に相当する W 個から輻輳回避ラウンドが再開される。cwnd は、1 RTT ラウンド毎に W_{inc} 個増加されるが、(6)式に従う場合、$\alpha = W_{inc}$ になるため、



(a) シングルフローモデル



(b) 競合フローモデル

図 6: 評価結果 (model:解析、sim:シミュレーション)

高々1 RTT ラウンドで cwnd の増加は完了し、安定状態に移行する。以降、ロスベースウィンドウが追い付くまでバッファに α 個滞留されるようにパケットを注入し、ロスモードに移行する。図 7 で横軸は RTT ラウンド数を表しており、図中の時刻 a は α / W_{inc} =1 で与えられる。また図中には、ロスベースウィンドウが BDP に達する時刻と cwnd に追い付く時刻も示している。

次に図 8 は、競合フローが存在する場合の TCP-Fusion の輻輳ウィンドウの挙動を示している。パケット再送のラウンド終了後、TCP-Fusion は $W/2$ 個 (BDP の半分)から輻輳回避を再開し、1 RTT ラウンド毎に W_{inc} 個ずつ cwnd を増加する。そして、バッファ滞留パケットが α 個に達すると、RTT の増加を回避するために徐々に cwnd を減少し、やがてロスベースウィンドウが追い付き、ロスモードに移行する。図中には、cwnd が理想ハイブリッド方式の輻輳ウィンドウと交差する時刻 b、バッファ内パケット数が α に達する時刻 c、ロスベースウィンドウが cwnd に追い付く時刻 d を示しているが、これらはそれぞれ次式で与えられる。

$$b = \frac{W-w}{1+W_{inc}},\ c = \frac{W-w+\alpha}{1+W_{inc}},\ d = \frac{W-w+\alpha}{2} \qquad (11)$$

詳細は省略するが、前節と同様の手順 (送信パケット数、経過時間、タイムアウトペナルティの算出) により、解析的なスループットを算出することができる。これに従い、シミュレーションとの対比実験として、変数 α と W_{inc} の影響の評価を行った。図 9 には、競合フローモデルにおいて α を 1/8 倍、1 倍、8 倍した場合の結果のみを示すが、要点は以下の通りである。

- α を大きくすると、TCP-Fusion のスループットは増加するが、TCP-Reno のスループットを若干押し下げる。一方、α を小さくしても変化は小さい。
- W_{inc} を小さくすると、TCP-Fusion のスループットは減少するが、TCP-Reno のスループットは変化しない。一方、W_{inc} を大きくしても、低ロス率時の変化は小さい。

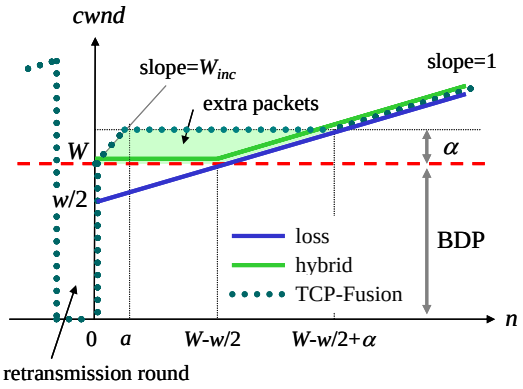


図 7: TCP-Fusion の輻輳ウィンドウ(シングルモデル)

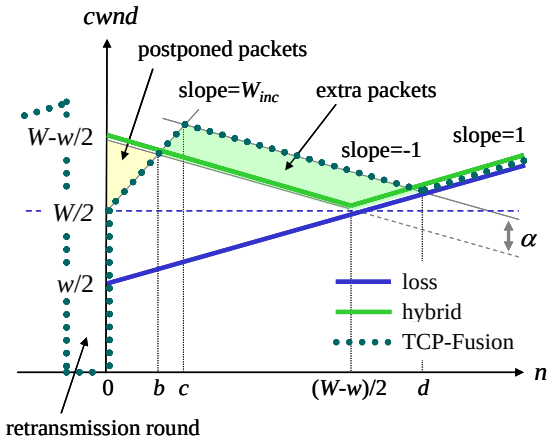


図 8: TCP-Fusion の輻輳ウィンドウ(競合モデル)

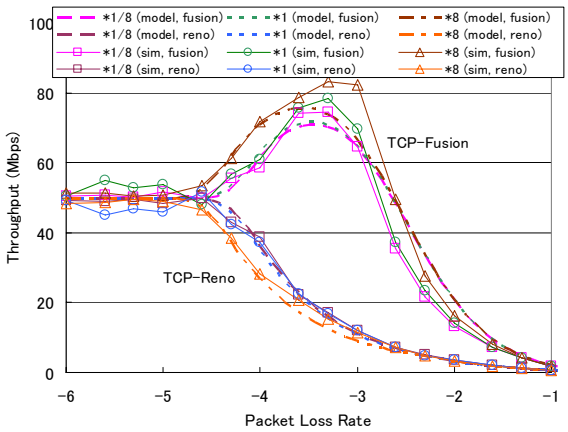


図 9: パラメータ α の影響の評価

3.3. 遅延ベース TFRC

映像配信応用を想定し、既存の TFRC に代わり得る方式提案として、遅延ベースの TFRC (D-TFRC) の検討を行った。手順としては、図 10 に示す遅延ベース輻輳制御の競合フローモデルに基づく定式化を行った。

まず、M を競合フロー数、β を最小時間解像度 $D_{\min}$ で送信可能なパケット数とする。さらに、変数 m を定義し、各競合フローは $m/M\cdot\beta$ 個のパケットをバッファ内に留めようとすると仮定する。すると、これまで述べてきた同様の手順に従い(手順は省略)、各競合フローのスループットは次式で見積もることができる。

$$R_{delay}=\frac{\frac{w}{2}\left(\frac{W}{M}+\frac{m}{M}\cdot\beta\right)\cdot PS}{\frac{1}{2}w\left(RTT_{\min}+m\cdot D_{\min}\right)+t_{RTO,delay}} \tag{12}$$

この式に対して、遅延ベース制御とロスベース制御の送信パケット数の比

$$\frac{K_{delay}}{K_{loss}}=\frac{W\cdot w/2}{3/8\cdot w^2}=\frac{4W}{3w}, \tag{13}$$

並びに、(7)式、$\beta=\frac{B*D_{\min}}{PS}$、$B=\frac{W*PS}{D_{\min}}$ を用いることで、(12)式は次式に変形される。

$$R_{delay}=\frac{\frac{B}{M}\left(1+\frac{m}{M}\cdot\frac{D_{\min}}{RTT_{\min}}\right)}{\left(1+m\cdot\frac{D_{\min}}{RTT_{\min}}\right)+p\cdot\frac{B}{PS}\cdot t_{RTO,loss}} \tag{14}$$

ここでさらに理想ハイブリッド制御として、パケットをバッファリングしない、すなわち $D_{\min}=0$ とすると、

$$R_{delay}=\frac{B/M}{1+p\cdot B/PS\cdot t_{RTO,loss}} \tag{15}$$

が与えられる。この(15)式は直感的に理解しやすく、帯域 B を M 人で共有する時に、分母のパケットロス率(及びタイムアウトペナルティ)が大きい場合には、実効レートが低下する様子を定式化している。

図 11 には、従来の TFRC 方式[18,19]との比較として、2 フロー競合(M=2)時の遅延ベース TFRC のスループット特性を示す。この図で D-TFRC が(15)式、変数 m を添えたものが(14)式に基づく解析結果で、m=1、すなわち滞留パケット数をスケーラブルに減少させる方がスループット特性が優れていることもわかる。

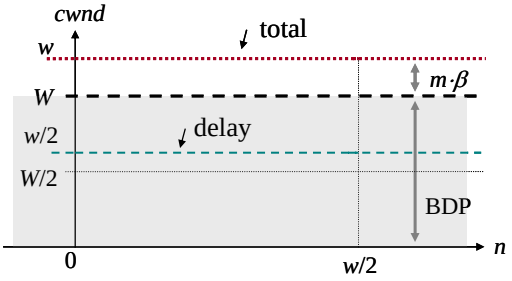


図 10: 遅延ベース輻輳制御の競合フローモデル

4. おわりに

本稿ではハイブリッド型 TCP 輻輳制御の理論解析手法の提案を行い、シミュレーションを併用した評価結果として良好な結果を得た。ただし、今回は高々2 フロー競合時の評価しか行っておらず、今後はマルチフローや短時間フロー、並びにリンク層再送を行う無線環境の評価などに検討対象を広げる。また、遅延ベース TFRC を始めとする方式について、エミュレータや実ネットワークを用いた映像配信実験も進める。

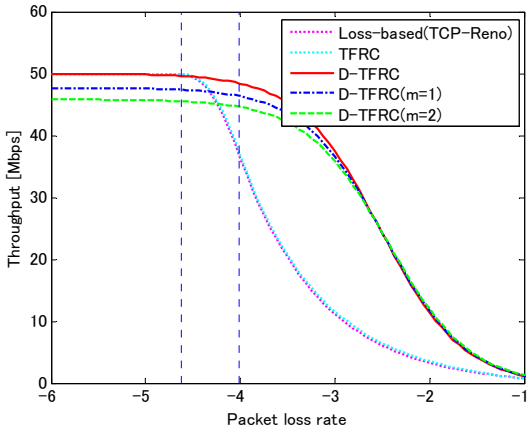


図 11: 遅延ベース TFRC のスループット特性

文　献

[1] M.Allman et al: "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," RFC 2581, Apr.1999.

[2] C.Casetti et al: "TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links", ACM Mobicom 2001, Jul.2001.

[3] R.Wang et al: "Efficiency/Friendliness Tradeoffs in TCP Westwood", IEEE SCC 2002. Jul.2002.

[4] R.Wang et al: "Adaptive Bandwidth Share Estimation in TCP Westwood", IEEE Globecom 2002, Nov.2002.

[5] K.Tokuda et al: “Performance Analysis of HighSpeed TCP and Its Improvement for High Throughput and Fairness against TCP Reno Connections”, PfHSN 2003, Mar.2003.

[6] R.King et al: “TCP-Africa: An Adaptive and Fair Rapid Increase Rule for Scalable TCP”, IEEE INFOCOM 2005, Mar.2005.

[7] H.Shimonishi et al: “TCP-Adaptive Reno for Improving Efficiency-Friendliness Tradeoffs of TCP Congestion Control Algorithm”, PFLDnet 2006, Feb.2006.

[8] K.Tan et al: “Compound TCP: A Scalable and TCP-Friendly Congestion Control for High-speed Networks”, PFLDnet 2006, Feb.2006.

[9] A.Baiocchi et al: “YeAH-TCP: Yet Another Highspeed TCP”, PFLDnet 2007, Feb.2007.

[10] K.Kaneko et al: “TCP-Fusion: A Hybrid Congestion Control Algorithm for High-speed Networks”, PFLDnet 2007, Feb.2007.

[11] Y.Gu et al.: "Congestion Control for Small Buffer High Speed Networks", IEEE INFOCOM 2007, May.2007.

[12] M.Mathis et al: "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm", ACM SIGCOMM CCR, Vol.27, No.3, Jul.1997.

[13] J.Padhye et al: "Modeling TCP Throughput: A Simple Model and its Empirical Validation", ACM SIGCOMM 1998, Oct.1998.

[14] C. Samios et al: "Modeling the Throughput of TCP Vegas", ACM SIGMETRICS 2003, June 2003.

[15] C.Jin et al: “FAST TCP: Motivation, Architecture, Algorithms, Performance”, IEEE INFOCOM 2004, Mar.2004.

[16] A.Tang et al: “An Accurate Link Model and Its Application to Stability Analysis of FAST TCP”, IEEE INFOCOM 2007, May.2007.

[17] E.Altman et al: "Analysis of TCP Westwood+ in High Speed Networks", PFLDnet 2006, Feb.2006.

[18] M. Handley et al.: "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 3448, Jan.2003.

[19] E. Kohler et al: "Datagram Congestion Control Protocol (DCCP)", RFC 4340, Mar.2006.

[20] G.Yang et al: "Smooth and Efficient Real-time Video Transport in the Presence of Wireless Errors", ACM Trans. MCCA, pp.109-126, May.2006.

[21] “ns-2 network simulator”, http://www.mash.cs.berkley.edu/ns.