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.

TCP-AFEC: TCP によるエンドホスト間の帯域確保のための FEC 冗長度動的決定手法

津川　知朗† 　藤田　範人†† 　浜　崇之†† 　下西　英之†† 　村瀬　勉††

† 大阪大学大学院情報科学研究科　〒 565–0871 大阪府吹田市山田丘 1–5
†† NEC システムプラットフォーム研究所　〒 211–8666 神奈川県川崎市中原区下沼部 1753
E-mail: t-tugawa@ist.osaka-u.ac.jp, n-fujita@bk.jp.nec.com, t-hama@cb.jp.nec.com,
h-shimonishi@cd.jp.nec.com, t-murase@ap.jp.nec.com

あらまし　Forward Error Correction (FEC) 技術を用いてパケットの冗長化を行うことは，パケット廃棄を低減させてスループットを安定させることができるなど，TCP によるストリーミング通信においても非常に有効である．しかしながら，冗長度を必要以上に大きくした場合には，ネットワーク帯域の利用効率が低下する．冗長度の動的決定手法に関しては，これまでにも数多くの研究が行われているが，それらの手法をそのまま TCP に適用した場合には，冗長度が小さくなりすぎて要求された転送レートを維持できないなど，十分な効果が得られない．そこで，本稿では TCP において，要求された帯域を確保しつつ無駄なネットワーク帯域の消費を最小限に抑えるための冗長動的決定手法を提案する．シミュレーションによる性能評価を通じて，提案した冗長度動的決定手法がネットワーク環境に関わらず，常に最適な冗長度に自動的に設定されることを確認する．

キーワード　Forward Error Correction (FEC)，帯域確保，ストリーミング，TCP

TCP-AFEC: An Adaptive FEC Code Control for End-to-End Bandwidth Guarantee

Tomoaki TSUGAWA†, Norihito FUJITA††, Takayuki HAMA††, Hideyuki SHIMONISHI††, and Tutomu MURASE††

† Graduate School of Information Science and Technology, Osaka University　Yamadaoka 1–5, Suita, Osaka 565–0871 Japan
†† System Platforms Research Laboratories, NEC Corporation
1753 Shimonumabe, Nakahara-ku, Kawasaki, Kanagawa, 211–8666, Japan
E-mail: t-tugawa@ist.osaka-u.ac.jp, n-fujita@bk.jp.nec.com, t-hama@cb.jp.nec.com,
h-shimonishi@cd.jp.nec.com, t-murase@ap.jp.nec.com

Abstract　Forward Error Correction (FEC) is useful for stable TCP video streaming due to its packet loss resilience. However, redundancy has to be appropriately determined so that redundant packets do not waste network resources between sender and receiver endhosts. Although many researchers have proposed mechanisms which determine redundancy adaptively based on network conditions, these mechanisms cannot be applied directly to TCP extended with FEC because they do not consider TCP congestion control. In this paper, we propose a new adaptive FEC code control scheme suitable for the extended TCP (TCP-AFEC), which dynamically sets redundancy required to achieve requested bandwidth based on both packet loss pattern and congestion window size. Simulation results show that the proposed scheme performs better than previous FEC control approaches on various network conditions.

Key words　Forward Error Correction (FEC)，Bandwidth guarantee，Streaming，TCP

1. はじめに

近年，BIGLOBE ストリーム [1] や YouTube [2] などの VOD (Video on Demand) サービスが急速に普及してきており，それに伴って動画像のストリーミング通信に対する需要も高まっている．従来，ストリーミングなどのリアルタイム通信においては，その適応性などからトランスポート層プロトコルとして UDP が多く用いられてきたが，UDP による通信は企業内に存在するファイアウォールや家庭のブロードバンドルータなどによって遮断されてしまい，サービスを提供できない環境も存在する．これに対して，TCP [3] を用いる場合にはそれらのネットワーク機器の影響を受けずに通信を行うことができるため，TCP によるストリーミング通信が注目されている．しかしながら，TCP は通信品質の面から見るとストリーミング通信に適さないことが指摘されている [4]．TCP はパケット廃棄を検知すると輻輳ウィンドウサイズを大きく減少させるため，一時的にストリーミング通信に必要な転送レートを確保することができずに遅延が増大し，映像が止まるなどの品質の劣化を招く．

従来の TCP は，ネットワークの輻輳状況のみを考慮して輻輳ウィンドウサイズを増減させるため，先に述べたような問題が発生する．そこで，アプリケーションから要求された転送レートを考慮して輻輳ウィンドウサイズを増減させることによって帯域を確保する TCP がこれまでに数多く提案されている [5, 6]．これらの TCP を用いることで，品質を劣化させることなく VOD サービスを提供することができる．しかしながら，輻輳ウィンドウサイズの制御による帯域確保 TCP は，要求される帯域が増加するなど条件が厳しくなると十分に機能しなくなることが考えられる．そのため，そのような場合においても要求された帯域を確保できるような TCP が必要となる．

帯域確保を実現するための別の方法として，FEC 技術 [7] を TCP に適用することが考えられる．先に述べたように，従来の TCP はパケット廃棄を契機として輻輳ウィンドウサイズを減少させる．そこで，図 1 のように TCP 層の下部に FEC 機構を設けて廃棄されたパケットを復号することによって，TCP からパケット廃棄を隠蔽し転送レートの低下を防ぐことができる．この方法を用いる場合，送信ホストに加えて受信ホストにも修正を加える必要があるが，輻輳ウィンドウサイズの制御による帯域確保方式よりも高い効果を期待できる．また，TCP の輻輳制御機構と FEC 技術を併用することで，要求された転送レートを満たすことができないほどネットワークに輻輳が発生している場合に，UDP を用いてストリーミング通信を行う場合とは異なり輻輳崩壊を回避できるなど [8, 9]，ネットワークの輻輳状況に応じて最適な制御が選択されることも期待できる．

FEC 技術による冗長化は先に述べたようにいくつかの利点が存在するが，冗長度を必要以上に大きくした場合にはネットワーク帯域の利用効率が低下するため，ネットワーク環境に応じて最適な冗長度に設定する必要がある．冗長度の動的決定手法に関してはこれまでにも UDP によるリアルタイム通信や無線通信に関する研究分野などで数多くの手法が提案されているが [10–12]，それらの冗長度動的決定手法をそのまま TCP に適用した場合には十分な効果が得られない．そこで本稿では，TCP において，要求された帯域を確保しつつ無駄なネットワーク帯域の消費を最小限に抑えるための冗長度動的決定手法を提案する．シミュレーションによる性能評価を通じて，提案した冗長度動的決定手法がネットワーク環境に関わらず，常に最適な冗長度に自動的に設定されることを確認する．また，輻輳ウィンドウサイズの制御による帯域確保 TCP と比較することによって，それらの方式に対する FEC 技術を用いた冗長化による帯域確保 TCP の優位性も明らかにする．

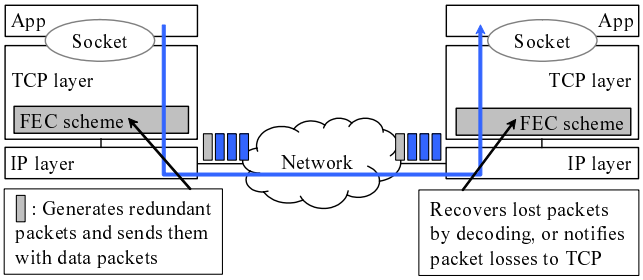


図 1　FEC 技術による冗長化を用いた TCP による通信．

以下，2 章では冗長度の動的決定手法における関連研究について説明する．3 章では，TCP における冗長度動的決定手法を提案し，そのアルゴリズムについて説明する．4 章では，シミュレーションにより，提案方式の性能の確認および輻輳ウィンドウサイズの制御による帯域確保 TCP に対する FEC 技術を用いた冗長化による帯域確保 TCP の優位性を明らかにする．最後に 5 章で，本稿のまとめと今後の課題を示す．

2. 関連研究

ネットワーク環境に応じて冗長度を動的に決定する手法に関しては，UDP によるリアルタイム通信や無線通信の分野などでこれまでに数多くの手法が提案されている [10–12]．例えば，文献 [10, 11] においては，RTCP (RTP Control Protocol) [13] によって得られる情報に基づく動的な冗長度決定手法が提案されている．しかしながら，文献 [10] でも指摘されているように，RTCP においては 5 秒に 1 回しかレポートが送信されないため，ネットワーク状況の変化に完全に適応することはできないことが考えられる [13]．また，これらの手法では目標となるパケット廃棄率の値が必要となるが，ストリーミング通信のように，パケット廃棄率の許容値を明示的に決定できないようなアプリケーションも数多く存在する．そのため，そのようなアプリケーションによる通信においては，パケット廃棄率の目標値を決定することが難しく十分に機能しないことが考えられる．

一方，文献 [12] では，冗長度を決定するためのタイマを用意し，そのタイマのタイムアウトおよびパケット廃棄を契機として冗長度を増減させることによって，ネットワーク環境に応じて動的に冗長度を決定する手法が提案されている．この手法は，文献 [10, 11] とは異なり，受信ホストからの特定の情報やパケット廃棄率の目標値などを必要としないため，先に述べたような問題は発生しない．しかしながら，この手法を TCP にそのまま適用した場合には，輻輳ウィンドウサイズの値に関わらず常に同じように制御するため，冗長度が小さくなりすぎて要求された転送レートを維持できないなど十分な効果が得られない．

3. 提案手法のアルゴリズム

本稿では，文献 [12] において提案されている手法へ，TCP の輻輳ウィンドウサイズに基づいて冗長度の増減を決定する手法を併用する新しい動的な冗長度決定手法を提案する．3.1 節

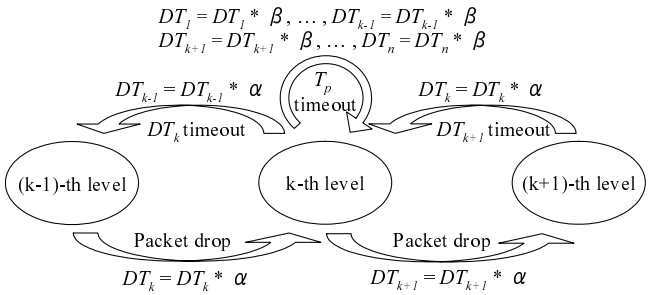


図 2　状態遷移図.

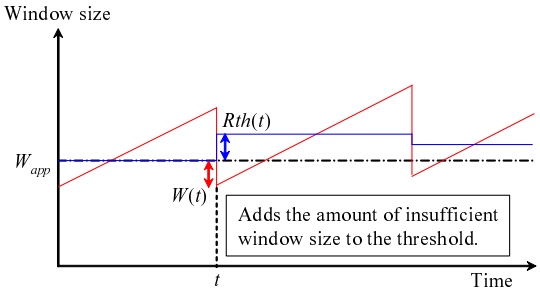


図 3　輻輳ウィンドウサイズおよび閾値 Rth の変化.

では，既存の動的な冗長度決定手法の概要を説明し，その手法をそのまま TCP に適用したときの問題点について議論する．3.2 節では，3.1 節で述べた問題点の解決策，および提案手法のアルゴリズムについて説明する．

提案手法を用いてネットワーク環境に応じた最適なパケットの冗長化を行い，TCP からパケット廃棄を隠蔽して一時的な転送レートの減少を防ぐことによって，スループットを安定させることも期待できる．これによって，ネットワーク帯域を有効に利用でき，コネクション数の多重度を増加させることができる．また，ストリーミング通信においては，映像を途切れなく再生するために受信ホストで動作するアプリケーションにバッファを用意するが，スループットを安定させることによって必要なバッファサイズを小さく抑えられることも期待できる．これらの点における，提案手法の優位性を 4.2 節で示す．

3.1　既存の動的な冗長度決定手法

文献 [12] で提案されている既存の動的な冗長度決定手法においては，DT (Drop Timer) と呼ばれるタイマを用意し，そのタイマに基づいて冗長度を増減させる．具体的には，図 2 に示されるように，DT がタイムアウトした場合には冗長度を減少させ，DT がタイムアウトするまでにパケット廃棄が発生した場合には冗長度を増加させるという制御方法を用いる．

既存手法は，DT のタイムアウト時間の長さを調節することによって，動的に冗長度を決定する手法である．タイムアウト時間が長いほどタイムアウトまでにパケット廃棄が発生する確率が高くなり，冗長度も大きくなりやすい．逆に，タイムアウト時間が短いほどタイムアウトが頻発し，冗長度も小さくなりやすい．既存手法においては，このタイムアウト時間は，2 種類のパラメータ α，β の値に依存する．これらのパラメータは，DT のタイムアウト時間を更新するためのパラメータであり，$\alpha > 1$，$0 < \beta < 1$ の範囲内の定数である．パケット廃棄や DT のタイムアウトによって状態遷移が起こると，DT の次回のタイムアウト時間は α 倍に増加される．また，状態遷移が起こらない間においても，DT のタイムアウト時間は一定時間ごとに更新される．このとき，タイムアウト時間は更新される度に β 倍に減少する．したがって，α の値が大きいほど冗長度も大きくなりやすく，β の値が小さいほど冗長度も小さくなりやすい．既存手法の詳細なアルゴリズムおよびその性能の解析については文献 [12] を参照されたい．

既存手法をそのまま TCP に適用した場合には，十分に機能しないことが考えられる．その原因のひとつに，TCP の輻輳制御機構の性質が挙げられる．TCP では，輻輳ウィンドウサイズを増減させることによって転送レートを調節する．そのため，輻輳ウィンドウサイズが小さいときに DT のタイムアウトによって冗長度を減少させるとパケット廃棄が発生しやすくなり，結果として十分な輻輳ウィンドウサイズを確保できない．また，別の問題として，既存手法は α および β のパラメータへの依存度が大きいため，パラメータ設定が難しいことが考えられる．先にも述べたように，既存手法は 2 種類のパラメータを用いて DT のタイムアウト時間を調節することによって冗長度を増減させる．そのため，パラメータ設定によっては不必要に冗長度が大きくなるなど十分な効果を発揮できない．4.1 節では，シミュレーションによる評価を通じて，既存手法のパラメータ設定が難しいことを明らかにする．

3.2　TCP-AFEC: TCP Adaptive FEC

本節では，提案手法である TCP-AFEC のアルゴリズムについて説明する．3.1 節で述べたように，既存手法をそのまま TCP へ適用した場合には，十分な効果を発揮できない．そこで，TCP-AFEC では既存手法へ TCP の輻輳ウィンドウサイズに基づいて冗長度を動的に決定する手法を併用する．この手法を併用することによって，提案手法は TCP においてもネットワーク環境に応じて最適な冗長度に自動的に設定される．

提案手法では，始めにアプリケーションからの要求転送レートを満たすために必要なウィンドウサイズを導出する．アプリケーションから要求される転送レートを R bps，その TCP コネクションの RTT の平均値を $\overline{T_{rtt}}$ sec とおくと，その TCP コネクションが要求転送レートを満たすために必要なウィンドウサイズ W_{app} は次式で表される．

$$W_{app} = R \cdot \overline{T_{rtt}} \tag{1}$$

今，時刻 t においてパケット廃棄，または再送タイマのタイムアウトにより輻輳ウィンドウサイズが減少したとする．このとき，減少した後の輻輳ウィンドウサイズを $W(t)$ とおくと，$W(t)$ が W_{app} よりも小さい場合にはアプリケーションからの要求転送レートを満たすことができない．そのため，輻輳ウィンドウサイズが不足している間は，通常よりも冗長度を大きくすることによって早期にアプリケーションからの要求転送レートを満たすことのできる状態へ回復することを目指す．TCP-AFEC では，通常よりも冗長度を大きくするかどうかを判定するために閾値を設定する．この閾値は，パケット廃棄，または再送タイマのタイムアウトが発生する度に更新される．パケット廃棄，または再送タイマのタイムアウトが発生した時刻 t における閾値を $Rth(t)$ とおくと，$Rth(t)$ は次式で表される．尚，輻輳ウィンドウサイズの最小値は 1 であるため，$Rth(t)$ が取り得る値の範囲は $W_{app} \leqq Rth(t) \leqq 2 \cdot W_{app} - 1$ となる．

$$Rth(t) = W_{app} + \max(W_{app} - W(t), 0) \tag{2}$$

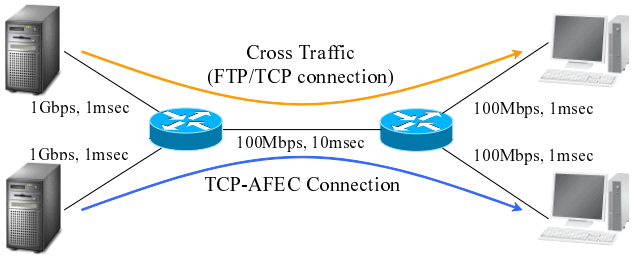


図 4　ネットワークモデル.

輻輳ウィンドウサイズが $Rth(t)$ よりも小さい場合には，DT (Drop Timer) のタイムアウトが発生しても冗長度を下げないというアルゴリズムを既存手法へ組み込むことにより，輻輳ウィンドウサイズがアプリケーションからの要求転送レートを満たすための十分な大きさを確保できていないときに冗長度を減少させすぎることを防ぐ．輻輳ウィンドウサイズが $Rth(t)$ よりも大きい場合は，既存手法のアルゴリズムに従って冗長度を増減させる．

提案手法の利点の一つとして，パラメータ設定の難しさが改善されること挙げられる．3.1 節でも述べたように，既存の冗長度動的決定手法には，2 つのパラメータ α，β の値の設定が難しく，パラメータ設定によっては要求された転送レートを満たすことができないことが考えられる．これに対して，提案手法は輻輳ウィンドウサイズを常に監視し，アプリケーションから要求された転送レートを満たすために必要と判断すると，DT のタイムアウトが発生しても冗長度を減少させない手法を組み込むことによって，パラメータ設定への依存度を減少させ，既存の冗長度動的決定手法では要求転送レートを満たすことができないような場合でも最適な冗長度に設定される．

4. 性 能 評 価

本章では，3 章で提案した動的な冗長度決定手法の有効性をシミュレーションにより評価する．シミュレーションは，ns-2 [14] を用いて行う．4.1 節では，提案手法がネットワーク環境によらず最適な冗長度に自動的に設定されることを確認する．また，4.2 節では，輻輳ウィンドウの制御による帯域確保 TCP と比較を行うことによって，それらの方式に対する FEC 技術を用いた冗長化による帯域確保 TCP の優位性を明らかにする．

シミュレーションに用いるネットワークモデルを図 4 に示す．クロストラヒックとして FTP を用いたデータのダウンロードによる TCP トラヒックを想定し，N 本の TCP コネクションを確立させてデータ転送を行うことによって発生させる．このようなネットワーク環境の下で，$(20+k,k)$ 符号によってデータパケットを冗長化する TCP を用いてデータ転送を行い，その性能を評価する．尚，$(20+k,k)$ 符号とは，20 個のパケットに対して k 個の冗長パケットを付加して冗長化することを表し，このときの冗長度は $\frac{20+k}{20}$ である．また，送信ホストのアプリケーションは映像ストリーミングを模擬しており，アプリケーションから TCP へは一定のビットレートでデータが渡される．

4.1 基本性能の確認

4.1.1 静的な冗長度決定手法との比較

図 5 は，ボトルネックリンクにおいてランダムなパケット廃棄が発生したときの，平均冗長度および要求転送レート達成割合の変化を表したものである．尚，達成割合とは，実効スループットがどれだけ要求転送レートを満たすかを表した値である．このシミュレーションにおいてはクロストラヒックは存在しない．また，アプリケーションから TCP へは 5 Mbps のビットレートでデータが渡される．すなわち，要求転送レートも 5 Mbps となる．図 5(b) を見ると，冗長度が 1.05 や 1.15 に設定された場合は，ランダムパケット廃棄率が大きくなると実効スループットが要求転送レートである 5 Mbps を満たすことができないことが分かる．逆に，冗長度が 1.25 に設定された場合は，達成割合の観点から見ると良い性能を示しているが，ランダムパケット廃棄率が小さいときには，必要以上に冗長度が大きくネットワーク帯域を無駄に消費していることが分かる．これに対して，提案手法である TCP-AFEC は常に要求転送レートを満たすことができ，かつできるだけ冗長度を小さく抑えていることが分かる．

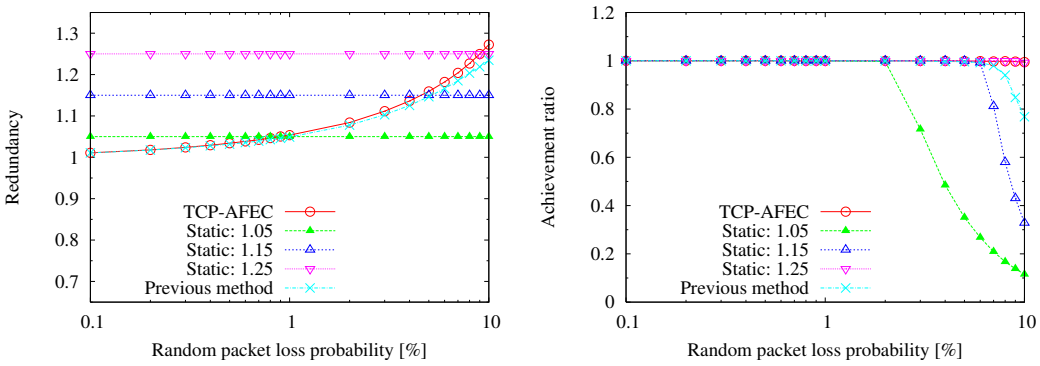
4.1.2 既存の動的な冗長度決定手法との比較

図 6 は，要求転送レートを変化させたときの平均廃棄率および要求転送レート達成割合の変化を表したものである．クロストラヒックを発生させる TCP コネクション数は 10 本であり，ボトルネックにおいてランダムなパケット廃棄は発生しない．また，既存の動的な冗長度決定手法については，冗長度を増減するためのパラメータである β を 0.9，0.95，および 0.99 に設定した場合の結果を示している．これらの図を見ると，既存手法においては，β の値が小さい場合は，要求転送レートが大きくなるとそれを満たすことができなくなる一方で，β の値が大きい場合は，条件によっては必要以上に冗長度が大きくなりネットワーク帯域を無駄に消費することが分かる．すなわち，既存手法はパラメータへの依存度が大きく最適なパラメータ設定が難しいことが分かる．これに対して，提案手法である TCP-AFEC は，常に輻輳ウィンドウサイズを監視することによって現在必要な冗長度を判断する手法を併用しているため，詳細なパラメータ設定を行わずとも要求転送レートを満たすための最適な冗長度に自動的に設定される．

4.2 輻輳ウィンドウ制御による帯域確保 TCP との比較

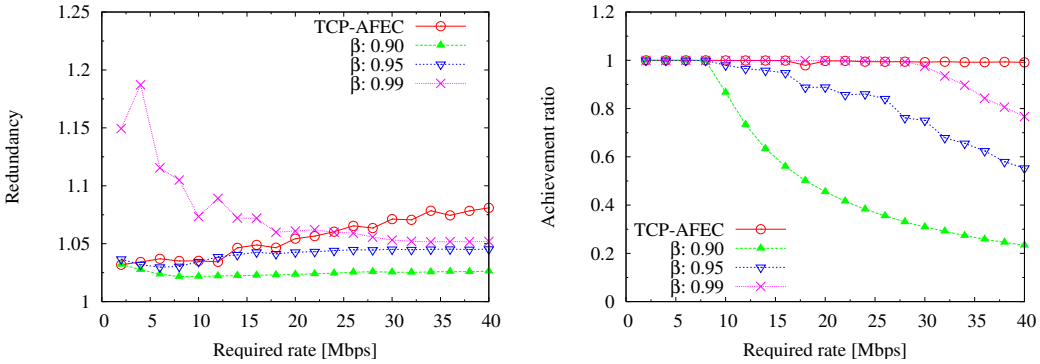
本節では，文献 [6] で提案されている TCP-AV と比較を行い，輻輳ウィンドウサイズの制御による帯域確保 TCP に対する提案手法の優位性を明らかにする．帯域を確保するための輻輳ウィンドウサイズの制御方法としては，確保する帯域に基づいて輻輳ウィンドウサイズを固定しておき，パケット廃棄やタイムアウトが発生しても輻輳ウィンドウサイズを減少させない手法も考えられる．しかしながら，この手法を用いた場合，ネットワークに重度の輻輳が発生したときに輻輳崩壊が発生してしまうことが考えられる．また，輻輳ウィンドウサイズを確保する帯域に基づいて固定する手法は，競合するトラヒック量が多い場合には，輻輳が発生している状況においても大量のパケットをネットワークに送出するため，バースト的なパケット廃棄が発生して転送効率が低下するため帯域を確保することができないことも指摘されている [5]．したがって，輻輳ウィンドウサイズを固定する手法は，今回の評価では比較対象としない．

図 7 に，ネットワーク環境を様々に変化させたときの要求転送レート達成割合を比較した図を示す．図 7(a) は，クロストラヒックを発生させる TCP コネクション数を 10 本に固定し，要求転送レートを変化させたときの結果，図 7(b) は，クロストラヒックは発生させず要求転送レートを 5 Mbps に固定し，ボ



(a) 平均冗長度.　　(b) 要求転送レート達成割合.

図 5　静的な冗長度決定手法との比較.



(a) 平均冗長度.　　(b) 要求転送レート達成割合.

図 6　既存の動的な冗長度決定手法 [12] との比較.

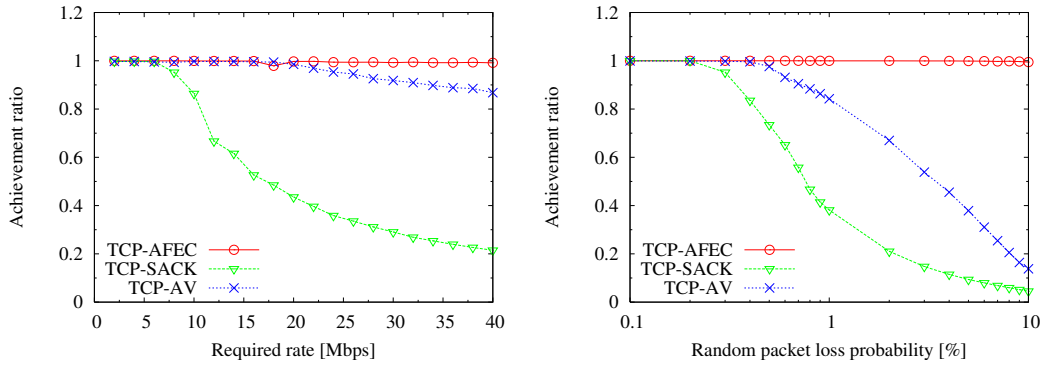
トルネックにおいてランダムなパケット廃棄を発生させたときの結果を示している．これらの図を見ると，いずれの場合においても環境が厳しくなるにつれて，TCP-AV は要求転送レートを満たすことができなくなることが分かる．これは，TCP-AV は条件が厳しくなると要求された帯域を確保しようと輻輳ウィンドウサイズを激しく変化させるために，スループットが安定しないためであると考えられる．これに対して，提案手法である TCP-AFEC は厳しい環境においても要求転送レートを満たすことができることが分かる．

図 8 は，クロストラヒックを発生させる TCP コネクション数を変化させたときに最大何本の TCP コネクションが要求転送レートを満たすことができるか比較した結果を表したものである．この図を見ると TCP-AFEC は，特にネットワークが輻輳しているときに，TCP-AV よりも多くの TCP コネクションを多重することができることが分かる．これは，先にも述べたように TCP-AV は厳しいネットワーク環境のもとでは輻輳ウィンドウサイズの変化が激しいため，ネットワーク帯域の利用効率が低下するためであると考えられる．それに対して TCP-AFEC は，冗長化によって余計なパケットをネットワークに送出するが，復号によってパケット廃棄を隠蔽するため，TCP が転送レートを低下させず常に一定の転送レートでパケットを送信しているためネットワーク帯域の利用効率が良い．

最後に，受信ホストのアプリケーションが持つバッファについての比較を行う．動画像のストリーミングにおける評価指標の一つとして，受信ホストのアプリケーションで必要なバッファサイズが挙げられる．図 9 にクロストラヒックを発生させる TCP コネクション数を変化させたときの，必要なバッファサイズの変化を示す．尚，この図では必要なバッファサイズの 95%分位点を示している．この図を見ると，必要なバッファサイズの観点から見ても，TCP-AFEC は TCP-AV よりも良い性能を示すことが分かる．これも，競合する TCP コネクション数が増加してネットワークが輻輳するにつれて，TCP-AV は帯域を確保しようと輻輳ウィンドウサイズを激しく変化させるためであると考えられる．これらの結果から，輻輳ウィンドウサイズの制御による帯域確保 TCP に対する提案手法の優位性が明らかになったと言える．

5. おわりに

本稿では，FEC 技術を適用した TCP において，要求された帯域を確保しつつ無駄なネットワーク帯域の消費を最小限に抑えるための冗長度動的決定手法を提案した．シミュレーションによる性能評価を通じて，提案手法である TCP-AFEC がネットワーク環境に関わらず，常に最適な冗長度に自動的に設定されることを示した．また，輻輳ウィンドウの制御による帯域



(a) 要求転送レートが変化する場合.　　(b) ランダムパケット廃棄率が変化する場合.

図 7　要求転送レート達成割合の比較.

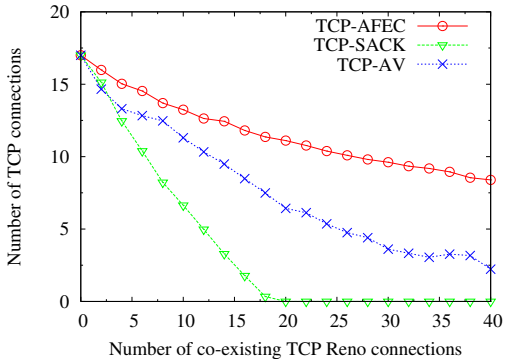


図 8　要求転送レートを満たした TCP コネクション数の変化.

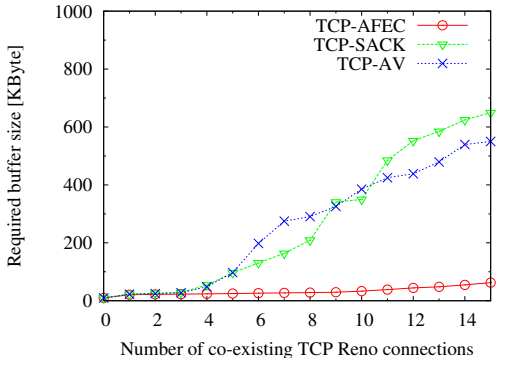


図 9　受信ホストで必要なバッファサイズの変化 (95%分位点).

確保 TCP に対する FEC 技術を用いた冗長化による帯域確保 TCP の優位性も明らかにした.

今後の課題としては，提案手法の性能の数学的な解析および提案手法の実装と実ネットワーク上での評価などが挙げられる.

謝　　辞

本研究の一部は，総務省戦略的情報通信研究開発推進制度「ネットワークサービスの早期展開を実現するオーバレイネットワーク基盤の研究開発」の支援を受けた．また，本稿を執筆するにあたり，大阪大学の村田 正幸教授および長谷川 剛助教授には多くの有用な助言を頂いた．ここに記して謝意を示す.

文　　献

[1] BIGLOBE ストリーム Home Page. available from `http://broadband.biglobe.ne.jp/`.

[2] YouTube Home Page. available from `http://www.youtube.com/`.

[3] J. B. Postel, “Transmission control protocol,” *RFC 793*, Sept. 1981.

[4] S. Floyd, M. Handley, J. Padhye, and J. Widmer, “Equation-based congestion control for unicast applications,” in *ACM SIGCOMM 2000*, Aug. 2000.

[5] K. Yamanegi, G. Hasegawa, and M. Murata, “Congestion control mechanism of TCP for achieving predicatable throughput,” in *Proceedings of Australian Telecommunication Networks and Applications Conference (ATNAC 2006)*, Dec. 2006.

[6] H. Shimonishi, T. Hama, and T. Murase, “TCP congestion control enhancements for streaming media,” in *Proceedings of IEEE Consumer Communications and Networking Conference (CCNC 2007)*, Jan. 2007.

[7] R. E. Blahut, *Theory and Practice of Error Control Codes*. Reading, Massachusetts: Addison-Wesley, 1983.

[8] S. Floyd and K. Fall, “Promoting the use of end-to-end congestion control in the Internet,” *IEEE/ACM Transactions on Networking*, vol. 7, pp. 458–472, Aug. 1999.

[9] B. P. Lee, R. K. Balan, L. Jacob, W. K. G. Seah, and A. L. Ananda, “TCP tunnels: Avoiding congestion collapse,” in *Proceedings of IEEE Conference on Local Computer Networks (LCN 2000)*, Nov. 2000.

[10] J. C. Bolot, S. Fosse-Parisis, and D. Towsley, “Adaptive FEC-based error control for Internet telephony,” in *Proceedings of IEEE INFOCOM 1999*, pp. 1453–1460, Mar. 1999.

[11] C. Padhye, K. J. Christensen, and W. Moreno, “A new adaptive FEC loss control algorithm for voice over IP applications,” in *Proceedings of IEEE International Performance, Computing and Communication Conference (IPCCC 2000)*, pp. 307–313, Feb. 2000.

[12] J. S. Ahn, S. W. Hong, and J. Heidemann, “An adaptive FEC code control algorithm for mobile wireless sensor networks,” *Journal of Communications and Networks*, vol. 7, pp. 489–499, Dec. 2005.

[13] H. Schulzrinne, S. Casner, R. Frederick, and V. Jaobson, “RTP:A transport protocol for real-time applications,” *RFC 3550*, July 2003.

[14] The VINT Project, “UCB/LBNL/VINT network simulator - ns (version 2).” available from `http://www.isi.edu/nsnam/ns/`.