社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2007－ＱＡＩ－22（８）
2007／1／19

複数経路を活用したTCP-Friendlyなストリーミングシステムの設計と実装

渡部 郁恵† 岡部 寿男†† 中村 素典††

† 京都大学大学院情報学研究科, 京都府
†† 京都大学学術情報メディアセンター, 京都府

あらまし 近年、双方向に映像・音声の伝送を行うTV会議システムのようなリアルタイム型アプリケーションが普及してきている。このようなシステムにおいて高品位映像を低遅延で伝送するには、広帯域な経路と、遅延やパケットロスの影響を抑える手法が必要になる。従来はパケットロスへの低遅延の対策としてFECが使われているが、バースト的なパケットロスに対する耐性や狭帯域経路の利用に限界があった。そこで我々はこれまでFECとPacket Path Diversityによる冗長配送を組み合わせた方式を提案・実装してきた。さらに今回はTFRCのレート制御アルゴリズムを利用することにより、自動的に利用可能な帯域にあわせた冗長度で冗長配送を行うことで、ユーザの負担を軽減するとともに、不適切なレート設定によりネットワーク状況が不安定になるという問題を解決した。
キーワード リアルタイム・ストリーミング, 前方訂正誤り符号 (FEC), Packet Path Diversity, TFRC

Design and Implementation of TCP-Friendly Multi-Path Streaming System

Ikue WATANABE†, Yasuo OKABE††, and Motonori NAKAMURA††

† Graduate School of Infomatics, Kyoto University, Kyoto, 606-8501 Japan
†† Academic Center for Computing and Media Studies, Kyoto University, Kyoto, 606-8501 Japan

Abstract Recently, real-time applications like TV conferencing systems that interactively transmit image and voice are becoming widly used. These systems need broadband route and technique for curbing the influence of packet losses without unnecessary delay to transmit the high-definition video with low latency. Existing systems which use FEC as a measure against packet losses have limits in the tolerance to burst packet losses and the use of the narrowband route. Then, we have proposed and implemented a system which combined FEC with redundant transmission by Packet Path Diversity. In this paper, we have adopted the rate control algorithm of TFRC to set the sending rate automatically to the available bandwidth and have enabled to reduce the user's load and solve the problem that the network situation becomes unstable by the improper rate setting.
Key words Realtime Streaming, FEC, Packet Path Diversity, TFRC

1. はじめに

インターネットの普及と広帯域化により、インターネット上でのマルチメディア・コミュニケーションが様々な形で実現されるようになってきた。これに伴い、TV会議システムや遠隔講義システムなどの双方向に映像・音声の伝送を行うリアルタイム型アプリケーションへの注目も高まってきており、インターネットを用いたマルチメディア・コミュニケーションは今後益々増加すると考えられる。しかし、このような通信が増加すると、ネットワーク上を流れるトラフィックが増大し、パケットロスなどの障害が頻発することが予想される。したがって、これらの障害の影響を抑え、マルチメディア・アプリケーションによるサービスを安定してを提供する仕組みが必要となる。

Robst [1] などの既存のアプリケーションではパケットロスへの対策としてForward Error Correction(FEC) [2] を使用している。FECは再送を伴わずに受信者側で誤り訂正できるような符号化方式を指し、遅延をあまり増加させないという特徴がある。しかしながら、複数のパケットが連続して損失してしまうバースト的なパケットロスに対処すると、遅延が増加してしまう。また、これらの既存システムには狭帯域な経路を利用するための手法がないため、十分な帯域が確保できないときには高品位映像を低遅延で伝送することは不可能であった。

そこで、我々はこれまで FEC と Pacekt Path Diversity(PPD) [3] を組み合わせた方式を提案・実装してきた [4]。PPD とは、送信者・受信者間に複数の経路が存在する場合に、その複数の経路を同時に利用してデータを伝送する手法を指す。この手法を用いることにより、狭帯域な経路を効率的に利用したり、バースト的なパケットロスの影響を抑えたりすることができる。しかし、このシステムにもユーザが手動で送信レートを設定することにより生じる様々な問題があった。

そこで本研究では、FEC と PPD に加え、TCP-Friendly Rate Control(TFRC) [5] のレート制御アルゴリズムに注目し、システムが自動的に利用可能な帯域にあわせた冗長度で冗長配送を行う手法を提案する。TFRC は TCP [6] ストリームと公平にネットワーク帯域を共有できる輻輳適応型レート制御として有効であると考えられている。しかしながら、TFRC はレート制御が上位のアプリケーションに与える影響を考慮していないため、リアルタイム型マルチメディア・アプリケーションのような細かなレート変更に不向きなシステムには適用することができないかった [7]。今回提案する手法では、FEC 及び PPD と TFRC を併用することによりこの問題を解決し、どのようなリアルタイム・アプリケーションでも TFRC を使用することができるようになった。また、送信レートの変更を自動化することにより、ユーザの負担を軽減するとともに、不適切なレート設定によってネットワークの状況が不安定になるという問題も解決することができた。

ここで章立てについて説明する。第 2 章でパケットロスへの対策とその手法を用いて実装された既存システムを紹介する。続いて、第 3 章でシステムの提案を行い、これに基づいて第 4 章でシステムを設計する。最後に、第 5 章で本研究をまとめる。

2. パケットロスへの対策と既存システム

遠隔講義システムのように双方向に映像・音声を伝送してマルチメディア・コミュニケーションを実現するリアルタイム型アプリケーションにおいて、パケットロスは映像や音声の乱れや途切れの原因となる。したがって、これらのアプリケーションによるサービスを安定して提供するためには、パケットロスの影響をなければならない。

本章ではパケットロスの影響を抑えるための手法として FEC と PPD を簡単に説明し、これらを用いて設計された既存のシステムとして、Robst と Path Diversity EMON(PDEMON) を紹介する。

2.1 パケットロスへの対策

2.1.1 Forward Error Correction

再送を伴わずに受信者側で誤り訂正できるような符号化方式を、Forward Error Correction(FEC) と呼ぶ。FEC は、データブロックから冗長なデータを生成し、もとのデータと冗長なデータの両方を伝送することで、データブロックのビット誤りや欠損を少ない遅延で復元することができる。

Reed-Solomon 符号を用いた FEC では、同じ長さの K 個のデータに対し、R 個の冗長なデータを生成して伝送する。この K+R 個のデータの組を FEC ブロックと呼ぶ。FEC ブロック中の K+R 個のデータのうち、任意の K 個のデータを受信すれば K+R 個すべてのデータを復元できる。しかし、R+1 個以上のデータが損失してしまうとデータの復元が不可能になるため、短時間に大量のロスが発生するバーストロスに弱い。また、FEC は冗長なデータを伝送することで伝送量を増やすため、狭帯域が原因でパケットロスが発生している場合に使用すると、かえってパケットロスが増加し、品質を低下させてしまう。

したがって、バースト的なパケットロスが発生する経路や狭帯域な経路を使用する場合は、他の手法を検討しなければらない。

2.1.2 Packet Path Diversity

Packet Path Diversity(PPD) は送信者・受信者間の複数の経路に同時にデータを送信する手法であり、負荷分散、冗長配送 [8] という 2 つの手法と組み合わせることで、より効果を上げることができる。

負荷分散とは、パケットを複数の経路に分散して配送する手法のことである。狭帯域な経路を使用する場合、負荷分散を用いると帯域の不足に伴うパケットロスを発生させずにすむ。

冗長配送とは、同じパケットを複数回冗長に送信する手法ことである。複数の独立な経路が存在する場合、それらの経路に冗長にパケットを送信すると、一部の経路で機器の障害やバーストパケットロスによる比較的長い通信の断絶が生じても、その影響を小さくすることができる。同じ割合で冗長なパケットを送信する場合、FEC で冗長化して 1 本の経路に全てのパケットを送信するよりも、複数の経路に同じパケットを冗長配送する方が遅延が減少し、品質も向上する [4]。

ただし、本論文では互いに共有部分を持たない経路を独立な経路と呼ぶ。

2.2 既存のシステムと問題点

遠隔講義向けに、FEC や PPD の手法を用いて開発されたストリーミング・システムには Robst や PDEMON などがある。

Robst は広島大学で開発された、高品質動画像をインターネット上で送受信するソフトウェアである。この Robst は、FEC を用いた伝送を実現しており、パケットロスによる映像/音声への影響を抑制することができる。また、送信開始時に FEC の冗長度を指定することができる。ただし、その他のパラメータ (送信レート等) を指定することはできない。つまり、Robst は常に一定のレートでデータを送信し続けるため、ネットワーク上を流れるトラフィックが増加して利用可能な帯域が送信レート以下になった場合、大量のパケットロスが発生して通信不可能になってしまう。また、このとき伝送不可能な量のパケットを送信するため、Robst が通信不可能になるだけでなく、輻輳制御を行っている他のストリームまで阻害する結果となる。さらに、Robst は 1 本の経路に FEC で冗長化したデータと元データを送信するだけなので、バースト的なパケットロスへの対策が不十分である、狭帯域な経路では通信できないなどの問題もある。

一方、PDEMON は、パケットロスへの対策として FEC と PPD を組み合わせて用いる。FEC を用いるため、Robst と同様にパケットロスを受信者側で回復し、パケットロスによる映

像・音声の乱れや途切れを抑制することができる。また、PPDを利用して複数の経路に同時にデータを送信することにより、複数の狭帯域な経路で高品位映像を伝送したり、バースト・パケットロスへの耐性を増すことができる。さらに、PDEMONではデータ伝送中に複数のパラメータを変更することができる。変更可能なパラメータは、各経路の送信レート、FECのブロックサイズ、FECの冗長度の3つである。しかし、これらの値の決定と変更はユーザが手動で行わなければならないため、ユーザは通信中常にネットワーク状況を監視していなければならない。特に、PDEMONではPPDにより複数の経路を使用した通信を行うため、モニターするデータ量が多く、ネットワークに関する高度な知識が必要となる。また、これらのパラメータの値を決定するための明確な指針がなく、ユーザの勘に頼ったシステムの運営が必要となる。従って、ユーザが利用可能な帯域以上の送信レートを設定てしまった場合、その経路上でパケットロスが増加し、その経路を利用する全てのアプリケーションの通信に悪影響が及ぶ。

そこで、本研究ではユーザの負担を軽減し、ネットワークを安定した状態に保つために、システムが自動的に送信レートを決定する仕組みを検討する。

3. システムの提案

3.1 自動化の検討

現在利用されているリアルタイム型アプリケーションのほとんどは、トランスポート層プロトコルにUDP[9]を用いている。UDPのような輻輳制御されていないトラフィックが急激に増加すると、やがて輻輳制御機構を持つTCPなどのトラフィックまで阻害し、ネットワークが不安定な状態に陥ってしまう。過去にもバックボーンネットワークにおいて輻輳崩壊と呼ばれる問題が発生しており[10]、今後リアルタイム・アプリケーションの増加により同様の問題が発生する恐れがある。また、ネットワークが安定した状態にあっても、帯域のほとんどがUDPストリームに占有されているような場合、TCPなどの輻輳制御を行うプロトコルを利用するサービスはデータを全く送信できなくなる。

したがって、リアルタイム型マルチメディア・アプリケーションによるストリーミングを含む全てのサービスを安定して提供するためには、全てのアプリケーションで適切に輻輳制御を行い、ネットワークを安定した状態に保つ必要がある。この輻輳制御とは、利用可能帯域にあわせて送信レートを増減させるだけでなく、他のストリームと帯域を共有できるように利用帯域を調整することを指す。

そこで本研究ではTFRC(TCP-Friendly Rate Control)に注目し、TCPストリームと公平に帯域を共有しながら自動的に利用可能な帯域にあわせた冗長度で冗長配送を行うシステムを提案する。

3.2 TFRC(TCP-Friendly Rate Control)

TFRCはインターネットにおいて、TCPデータ通信と公平かつTCPより安定したエンド間ユニキャスト通信を行うための輻輳適応型レート制御アルゴリズムである。TCPと公平な通信を実現するため、TFRCではネットワークの状態推定に基づいてTCPコネクションのスループットを予測し、データ送出レートを決定する。

3.2.1 送信ホストの動作

送信ホストはデータ・パケットにシーケンス番号、タイムスタンプ、送信ホスト側で推定したRTT(Round Trip Time)を付加し、規定の送信レートで受信ホストにデータを送信する。

送信ホストは、受信ホストから送信された制御パケットを受け取ると、制御パケットに含まれる送受信時刻から新たなRTTの観測値を計算し、TCPと同様のローパスフィルタを適用してRTTの推定値Rを導出する。その後、ネットワークの負荷状態を表すこれらのパラメータから式(1)を用いてTCPコネクションのスループットを予測し、これを新たな送信レートとする。

また、4RTT以内に次の制御パケットを受け取れなかった場合は、送信レートを約半分に減らす。

$$X = \frac{s}{R\sqrt{\frac{2}{3}bp} + t_{RTO}(3\sqrt{\frac{3}{8}p})p(1+32p^2)} \quad (1)$$

ただし、Xは送信レート、sはパケットサイズ、Rは推定したRTT、pはパケットロス率、T_{RTO}はTCPの再送タイムアウト、bはTCP acknowledgementで通知されるパケット数である。

3.2.2 受信ホストの動作

受信ホストはデータ・パケットを受信すると、新たなパケットロス率を計算する。ただし、1RTTにつき1度だけウィンドウサイズを小さくするというTCPのメカニズムにならい、1RTT内に発生したパケットロスはその回数に関わらず1回のロス・イベントとみなし、ロス・イベントの発生率をパケットロス率とする。

また、受信ホストは1RTTに1回以上制御パケットを送信ホストに向けて送出する。この制御パケットは最後に受信したデータ・パケットのタイムスタンプ、受信ホストが最後のデータパケットを受け取ってから制御パケットを生成するまでの経過時間、前の制御パケット送信後の受信レート、パケットロス率の情報を含む。

3.2.3 TFRCのストリーミング・システムへの適用

TFRCを用いればTCPと公平に帯域を分け合うように輻輳制御することが可能になるが、レート制御が上位のアプリケーションに与える影響を考慮していないため、TFRCをそのまま動画像等を送受信するストリーミング・システムに適用することはできない。

これらのシステムにおいて、送信レートの変化にあわせて実際に送信するデータ量を増減させるために、従来は、コーデック・レートを変更する、パディングを行うなどの手法が用いられてきた[11]。しかしながら、1RTTに1回のように、頻繁にコーデック・レートを変更すると、ユーザが感じるアプリケーションレベルの通信品質が極端に劣化する[7]。また、アプリケーション・データを持たないパケットでパディングを行うと、貴重な帯域を無駄に消費してしまうことにもなる。

そこで、本研究では、PPD により複数の経路を同時に利用し、また FEC による冗長データを増減させることにより、頻繁なコーデック・レートの変更とパディングを行わずに、TFRC をストリーミング・システムへ適用する手法を提案する。例えば、2 本の独立な経路を同時に利用して 30Mbps のビデオを送信する場合を考える。時刻 t_1 に経路 1 の送信レートが 35Mbps、経路 2 の送信レートが 15Mbps であったが、時刻 t_2 に経路 1 のトラフィックが増加して経路 1 の送信レートが 25Mbps、経路 2 の送信レートが 20Mbps に変化したとする。経路 1 のみを利用した場合には、時刻 t_2 に送信レートがビデオのコーデック・レートを下回るため、コーデック・レートを変更せずにデータを送信し続けることはできない。しかし、経路 1 と経路 2 を併用した場合には、時刻 t_1, t_2 どちらの場合も送信レートの合計は 30Mbps を上回るため、コーデック・レートを変更する必要はない。このように複数の独立な経路を同時に利用すると、ある経路の送信レートが急激に減少しても他の経路でその変化をカバーすることができる。また、送信レートの合計がビデオのコーデック・レートを上回る場合には、その余剰帯域を利用して FEC 冗長データを送信することでパケットロスへの耐性を増すこともできる。

このように、本手法を利用することにより、頻繁な送信レートの変更に不向きなリアルタイム型アプリケーションでも、TFRC を利用して TCP ストリームと公平に帯域を共有しながら輻輳制御を行い、かつ送信可能レートを自動的に変更できるようになる。

4. システムの設計

4.1 想定する環境

本研究では PPD により複数の経路に同時にデータを送信する。経路に重複する箇所が存在すると、その部分で障害が発生した場合そこを通る全ての経路の障害の影響が出て、重複箇所がボトルネックとなる。このため、できる限り重複のない独立な経路を利用することが望ましい。

複数のキャンパスを持つ大学において、キャンパス間で遠隔講義を行う場合を考える。複数のキャンパスを持つ大学などでは、キャンパス間に専用のネットワーク回線を引いていることが多い。また、障害発生に備えて、独立な複数の経路が用意されている。図 1 に 3 つのキャンパスを持つ大学の例を示す。この大学の場合、各キャンパス間は 1Gbps の帯域を持つネットワークで接続されている。例えばキャンパス A とキャンパス B の間で遠隔講義をする場合、キャンパス A とキャンパス B の間には 2 つのキャンパスを直接結ぶネットワーク A を通る経路と、ネットワーク C →キャンパス C →ネットワーク B のようにキャンパス C を経由する経路の 2 つが存在する。この 2 つの経路は重複箇所のない独立な経路である。

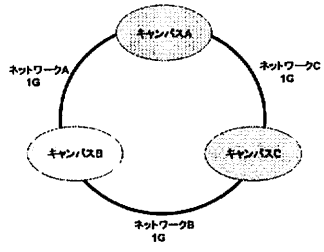

図 1 複数のキャンパスを持つ大学の例

本研究では、このように送信者・受信者間に複数の独立な経路が用意されいるとする。

4.2 中継ホスト

TFRC は RTP [12] を拡張して実現されている [13]。現在利用されているストリーミング・システムのほとんどはアプリケーション層プロトコルとして RTP を使用しているため、既存のシステムで TFRC を用いるためにはシステムの拡張が必要となる。しかし、全ての既存システムを拡張するのは時間的・金銭的コストから考えて不可能である。

4.2.1 中継ホストとしての実装

本研究では提案するシステムを中継ホストとして実装し、送信ホスト・受信ホスト間にこの中継ホストを入れるだけで、PPD 及び FEC と TFRC を使用可能にする。

図 2 は図 1 の大学において、キャンパス A とキャンパス B の間で遠隔講義を行うとき、キャンパス A で撮影した講義をキャンパス B に送信する場合の例である。ただし、ネットワーク A を通る経路を経路 1、キャンパス C を経由する経路を経路 2 とする。

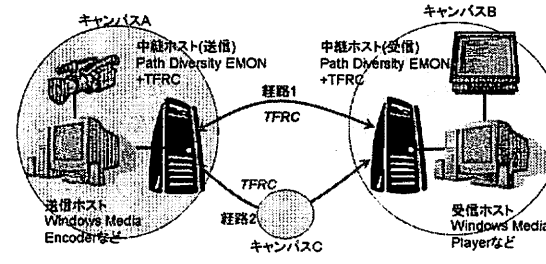

図 2 中継ホスト

送信側のキャンパス A にはカメラと、カメラに接続された送信ホストを用意する。この送信ホストには Windows Media Encoder などの既存のストリーミング・システムが入っている。次に本システムをインストールした中継ホスト (送信) を用意する。まず、送信ホストは通常通りカメラから画像や音声を取り込んでエンコードし、中継ホスト (送信) に送信する。データを受信した中継ホスト (送信) は FEC でエンコードした後、各経路にデータを振り分け、TFRC ヘッダを付加して中継ホスト (受信) に経路ごとに送信する。このように、本システムは経路ごとに TFRC を行い、経路ごとの送信レートを計算してデータの振り分けに利用する。

受信側のキャンパス B には本システムをインストールした中継ホスト (受信) と、Windows Media Player などの既存のプレイヤーをインストールした受信ホストを用意する。中継ホスト (送信) からのデータを受信した中継ホスト (受信) は、経路ごとのパケットロス率の計算と TFRC ヘッダの除去、FEC の

デコード行った後、データを受信ホストに送信する。このデータを受信した受信ホストは、通常通りデータを再生する。

このように中継ホスト (送信) と中継ホスト (送信) で PPD 及び FEC と TFRC に必要な処理を全て行うことで、送信ホスト・受信ホストには何も変更を加えることなく、これらの手法を利用することが可能になる。

中継ホストの詳細な設計については 4.3 節で説明する。

4.2.2 中継ホストの多重化

本システムを利用することにより、中継ホスト間でさらに経路を多重化することができる。この例を図 3 に示す。

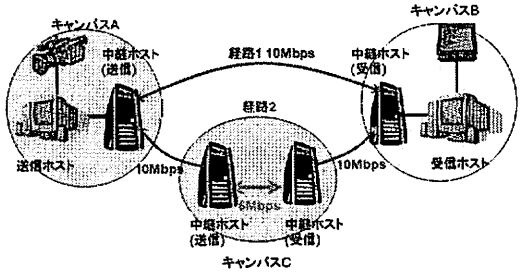

図 3 中継ホストの多重化

図 3 ではキャンパス C にさらに中継ホストを設置し、経路を多重化している。例えば、経路 1 と経路 2 では最大 10Mbps の速度でデータを送信できるはずであったが、キャンパス C では、6Mbps の回線または 2Mbps の回線しか確保できなかったとする。キャンパス A の中継ホスト (送信) とキャンパス B の中継ホスト (受信) のみ使ってデータを送信すると、経路 1 では最大 10Mbps、経路 2 では最大 6Mbps の速度でデータを伝送するので、合計で最大 16Mbps の帯域しか使うことができない。一方、図 3 のように、キャンパス C に更に中継ホストの組を置いて、この間でも PPD によって経路を 2 つ同時に利用すると経路 2 の帯域は最大 8Mbps となり、経路 1 と経路 2 をあわせると最大 18Mbps の帯域を使うことができるようになる。

このように、経路上の特定の部分のみが狭帯域な場合や、パケットロスが大量に発生するような場合、中継ホストを多重化することで利用できる帯域を増加させたり、冗長化によりパケットロスの影響を抑えたりすることが可能となる。

4.3 中継ホストの動作

4.3.1 中継ホスト (送信) の動作

本節では、送信ホストから RTP パケットを受信したとき及び中継ホスト (受信) から制御パケットを受信したときの動作をそれぞれ説明する。

まず、中継ホスト (送信) が送信ホストから RTP パケットを受信したときの動作を説明する。ただし、全ての FEC ブロックにおいて、i 番目に受信したストリーム・データを $data_i$、j 番目に生成した FEC 冗長データを $data_{i+j}$ とする。

(1) 送信ホストから RTP パケットを受信する。

(2) 受信した RTP パケットをシーケンス番号でソートする。

(3) 受信した RTP パケットが K 個になったら FEC でエンコードし、R 個の FEC 冗長データを生成する。このとき、受信した RTP パケットには新たな RTP ヘッダを、FEC 冗長データには FEC ヘッダと RTP ヘッダをそれぞれ付加する。

(4) (3) で処理した FEC ブロックのデータに、振り分け配列に従って TFRC ヘッダを付加して各経路に送信する。ただし、ストリーム・データと FEC 冗長データは異なるポートを用いる。

図 4 に中継ホスト (送信) が中継ホスト (受信) に送信するストリーム・データパケットを、図 5 に FEC 冗長データパケットを示す。振り分け配列については 4.4 節で説明する。

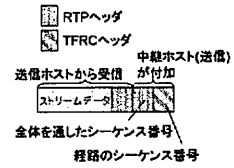

図 4 ストリーム・データパケット

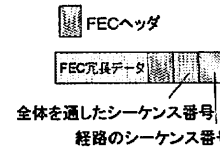

図 5 FEC 冗長データパケット

次に中継ホスト (送信) が中継ホスト (受信) から経路 l の制御パケットを受け取ったときの動作について述べる。

(1) 中継ホスト (受信) から制御パケットを受け取る。

(2) 制御パケットに含まれる送受信時刻から経路の往復時間 (RTT) の観測値を計算する。

(3) RTT の観測値にローパスフィルタを適用して RTT の推定値 R を計算する。

(4) 式 (1) に従い経路 l の送信レート X_l を求める。

(5) 振り分け配列を更新する。

4.3.2 中継ホスト (受信) の動作

中継ホスト (受信) の動作について説明する。

(1) 中継ホスト (送信) から経路 l を経由したデータを受信する。

(2) TFRC ヘッダのシーケンス番号で経路ごとにソートする。ただし、ストリーム・データと FEC 冗長データは別々にソートそる。

(3) TFRC ヘッダのタイムスタンプ、シーケンス番号、RTT から経路 l のパケットロス率を計算する。

(4) 最後に経路 l に制御パケットを送信してから RTT 以上時間が経過していたら、中継ホスト (送信) に向けて、経路 l の制御パケットを送信する。

(5) TFRC ヘッダを除去する。

(6) RTP ヘッダの (全体を通した) シーケンス番号で、全てのパケットをソートする。ことのきも、ストリーム・データと FEC 冗長データは別々にソートする。

(7) FEC でデコードする。

(8) RTP ヘッダを除去する。

(9) 受信ホストに向けて送信する。

4.4 送信可能レートにあわせたパケットの振り分け

4.3.1 節で述べたように、中継ホスト (送信) は制御パケット

を受信すると、経路 i に FEC ブロックの何番目のデータを送信するかを示す振り分け配列 array を更新する。本節ではこのアルゴリズムを説明する。ここで、記号を定義しておく。

- K：FEC の 1 つのブロックに含まれるストリーム・データの個数 (自然数)
- R：FEC の 1 つのブロックに含まれる FEC 冗長データの個数 (自然数)
- N：FEC のブロックサイズ ($N = K + R$)。15 に固定。
- X：送信するストリームのデータ量 (一定)
- x_i：経路 i の送信レート。
- p_i：経路 i のパケットロス率
- n：経路の本数

まず、ストリームを送信する前に各経路の統計情報を測定し、FEC のパラメータ N, K, R を決定する。ただし、N が大きくなると遅延が増加し、小さすぎるとバースト的なパケットロスへの耐性が減少するため、本システムでは妥当であると思われる $N = 15$ として実装を行った。全経路に送信できるデータの合計は $\sum_i x_i$、全ての経路で合計 $\sum_i x_i p_i$ のパケットが失われるので、$\frac{\sum_i x_i p_i}{\sum_i x_i} \leqq \frac{R}{N} < 1$ となるように R を設定すると、負荷分散を行った場合にも全てのパケットを受信できる。

FEC のパラメータを決定したらストリーム伝送を開始する。FEC を用いて冗長化することにより Xbps のストリームは $X' = \frac{N}{K} X$bps になる。中継ホスト (送信) は各経路の送信レートが変更されるたびに、以下アルゴリズムに従って配列 array を変更する。

（1） 変数を 0 で初期化する。

（2） $i = 0$ から $i = n - 1$ まで、全ての経路について array を更新する。

（a） $x_i \geqq X'$ なら、任意の k について array[i][k]=1 とする。

（b） $x_i < X'$ なら、FEC ブロックに含まれる N 個のデータのうち、経路 i に何個を送信できるか調べる。$\frac{k_i+1}{N} > \frac{x_i}{X'}$ のとき、経路 i には k_i 個のデータを送信できるので、array[i] の $(\sum_{l=1}^{i-1} k_l - 1)\%N$ 列から $(\sum_{l=1}^{i} k_l - 1)\%N$ 列までを 1 に変更し、経路 i の調査を終了する。

各 FEC ブロックの l 番目のパケットについて、array[i][l] が 1 ならば経路 i にパケットを送信する。

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
array[0]	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
array[1]	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0
array[2]	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1

表 1 配列 array の例

ここで、配列 array の例を示す。3 本の経路 ($n = 3$, 経路 0, 経路 1, 経路 2) を使い、FEC エンコード後に送信するデータの合計が 10Mbps($X' = 10$) となるようなストリームを送信するとする。経路 0、経路 1、経路 2 の送信レートがそれぞれ 12Mbps、8Mbps、6Mbps($x_0 = 12, x_1 = 8, x_2 = 6$) のとき、配列 array は表 1 のようになる。これより、例えば 8 番目のデータは経路 0 と経路 1 に送信する。

5. まとめ

本論文では、リアルタイム型マルチメディア・アプリケーションにおいて、パケットロスへの対策として FEC と PPD を併用することでより安定してデータの送受信が行えることを説明した。さらに、これらのアプリケーションが他のストリームを阻害しないように、TFRC を利用して送信レートを自動的に変更することを提案し、設計・実装した。

今後は本システムの有効性を検証する実験を行う予定である。

文献

[1] 西村 浩二, 近堂 徹, 田島 浩一, 岸場 清悟, 相原 玲二, "大学間遠隔コミュニケーションのための高品質動画像伝送システム", 学術情報処理研究, No.7, pp.43-52, *2003*

[2] L. Rizzo, "Effective Erasure Codes for Reliable Computer Communication Protocols", ACM Computer Communication Review, *1997*

[3] S. Mao, Y. Wang, S. Lin, S. Panwar, "Video transport over ad hoc networks with path diversity", ACM Mobile Computing and Communication Review, *2002*

[4] 渡部郁恵, "複数経路を活用したバーストパケットロスに強いストリーミングシステム", 電子情報通信学会 2005 年度総合大会ポスターセッション, *2005*

[5] M. Handley, S. Floyd, J. Padhye and J. Widmer, "TCP Friendly Rate Control (TFRC)", RFC3448, *2003*

[6] Marina del Rey, "TRANSMISSION CONTROL PROTOCOL", RFC793, *1981*

[7] 宮林正樹, 若宮直紀, 村田正幸, 宮原秀夫, "TCP データ通信との公平性を考慮した輻輳適応型レート制御による MPEG-4 動画像通信", 電子情報通信学会 技術研究報告 (IN2000-218), pp.193-200, *2001*

[8] C. Perloms and O. Hodson, "Options for Repair of Streaming Media", RFC2354, *1998*

[9] J.Pste, "User Datagram Protocol", RFC768, *1980*

[10] S.Floyd and J.Kempf, "IAB Concerns Regarding Congestion Control for Voice Traffic in the Internet", RFC3714, *2003*

[11] T. Phelan, "Strategies for Streaming Media Applications Using TCP-Friendly Rate Control", INTERNET-DRAFT draft-ietf-dccp-tfrc-media-00.txt, *2005*

[12] H. Schulzrinne, S. casner, R. Frederick and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC3550, *2003*

[13] Ladan Gharai, "RTP Profile for TCP Friendly Rate Control", Internet Engineering Task Force INTERNET-DRAFT draft-ietf-avt-tfrc-profile-06.txt, *2006*