# FEC on IPv6 for Reliable Multicast

米山 清二郎　　砂原 秀樹
Seijiro Yoneyama　Hideki Sunahara

奈良先端科学技術大学院大学
Nara Institute of Science and Technology

**概要**

IETF の RMT(Reliable Multicast Transport)WG で提案されている Building Blocks 方式の基礎となる FEC(Foward Error Correction) を IPv6 上で実装・評価した。物理的に巨大化したネットワークで受信者の多いマルチキャスト通信サービスは、ネットワーク輻輳を引き起こす可能性がある。そこで、既存のマルチキャストプロトコルと FEC を組み合わせることが、ネットワーク輻輳を抑える有効な方法であると考えられている。本稿では、FEC の仕様を検討し、IPv6 上で FEC を実装した。まず、FEC の基本性能であるスループット・処理速度を評価した。つぎに、FEC のパケットロスに対する復元率を測定し有効性を評価した。さらに、実トラヒックとして Digital Video ストリームを転送し、FEC のデータ再生品質を評価した。以上の結果から、本稿で実装した FEC が信頼性のあるマルチキャスト通信を実現する上での有効な指標を得ることができた。

## 1　はじめに

インターネットはトラヒック量および接続ノード数の増加が急激に進行し、エンドエンド間は地球的規模に拡大している。また、新しいアプリケーションの登場により、従来の Point-to-Point 型の通信形態から、エンドエンドでの大規模なマルチキャストサービスの提供が必要とされてきている。このような大規模マルチキャストサービスを提供する上で重要となる項目には、パケット転送遅延およびネットワークに輻輳を引き起こさないことがあげられる。現在、IETF の RMTWG [1] で、これらの項目を満足する信頼性のある大規模マルチキャストサービスを実現するアーキテクチャとして Building Blocks 方式が検討されている。その検討の中で、前方誤り訂正制御方式 [2] [3] [4] (FEC: Foward Error Correction) が有効な制御方式の一つとして提案されている。

マルチキャスト通信におけるパケット転送遅延およびネットワークの輻輳の主な原因としてパケットロスがある。しかし、現在のマルチキャスト通信は UDP を用いて実験されており、通信路上で生じるパケットロスに対する信頼性は提供していない。パケットロスに対する信頼性を保証する方法には、ARQ(Automatic Repeat reQuest) に基づくパケット再送による Feedback 型と、受信側で訂正する Open Loop 型がある。

Feedback 型は、エンドエンド間の距離が小さくリアルタイム性を要求されないメディアの場合、また、受信ノード数が少ないマルチキャスト通信の場合には有効である。しかし、大規模化したネットワークで、かつ、受信ノード数が多いマルチキャスト通信の場合では、以下の 3 つの理由によりマルチキャスト通信には適応しない。(a) 再送による遅延が増大する。(b) マルチキャストツリーで観測されるパケットロス・エラー確率は、受信ノードの数に比例して大きくなる。したがって、従来の再送による方法では再送パケットのオーバヘッドが非常に大きくなる。(c) ほとんどの受信ノードが正しく受信しても、受信できていない受信ノード

が存在すればパケットの再送を全ての受信ノードに対して行うためトラヒックが膨大になる [5] [6]。このように、転送遅延、パケットロスに対する信頼性を保証しつつ大規模なマルチキャスト通信を実現するためには、Feedback 型では制限がある。

これに対して、Open Loop 型の FEC は、実データに対して予め誤り訂正コードを付加して送信し、転送中に誤りが生じた場合には受信ノードが訂正コードを用いて自発的に訂正する。FEC は、この機構により再送による遅延・オーバヘッドの増加を防止できるため、大規模マルチキャスト通信には欠かせない技術であると認識されている。

RMTWG は FEC をマルチキャストプロトコル内に組み込む、という方向性を打ち出しているため、実装することが急務である。そこで、本稿では FEC を IP 層で実装する。本来、トランスポート層で実装すべきところを、IP 層である理由は、下位層データリンク技術に依存せず任意のデータリンクを経由するエンドエンドの通信に対して FEC を適用できること、トランスポート層に非依存とすることにより、多くのマルチキャストプロトコルとの組合せが可能になるだけでなく、多くのアプリケーションに対しても適用可能になることである。また、IPv6 は豊富なアドレス空間をもつため、受信者の多い大規模なネットワークで、エンドエンドで信頼性のあるデータ通信を提供できる。

誤り訂正符号には Reed Solomon 符号 ($G(x) = x^8 + x^4 + x^3 + x^2 + 1$) を用いている[1]。RMTWG では、他に Tornade 符合、LT 符合が提案されているが、それらと比較して Reed Solomon 符合は、処理速度が早い、生成できる誤り訂正符合数がスケールしにくい、誤り訂正符合による消費帯域が大きい、という特徴がある [7]。現状のルータの動作には、帯域の増加よりパケット数の増加の方が負荷が大きいため、ネットワーク輻輳を引き起こす可能性を考慮すると ReedSolomon 符合が適していると考えられる。また、データ部をエンコードすることなく誤り訂正符号を生成するため、パケットロスがない場合、FEC 未実装ノードでもデータの受信が可能である。

[1]Reed-Solomon 符号は 1 バイトを 1 シンボルとする誤り訂正符号である。

# 2 提案方式

## 2.1 評価項目

本稿で実装した FEC は、IP パケットを分割し誤り訂正コードを付加する手法を用いている。このとき、IPv6 フラグメント機能 [8] と分割機能を統一し、FEC を IPv6 フラグメントの拡張機能として実装した。これは、IPv6 フラグメント機能と FEC が類似した機構であるため、2 つを統一することで処理の軽減を図るためである。分割手法の利点としては、細かいデータを取り扱うことにより、到着するパケット間のジッタを小さくすることでストリームデータの再生品質を高く維持できることが挙げられる。

FEC の評価項目として以下の 4 つを挙げる。

1. スループット・処理速度
   FEC は誤り訂正コードの生成・復元処理が高負荷であるため、FEC の処理が限界に達した場合、結果として通信性能の低下する。また、他のマルチキャストプロトコル・アプリケーションとの併用を考慮すると、十分なスループット・処理速度を保証する必要がある。

2. FEC の有効性
   受信ノードの多いマルチキャスト通信ではパケットロスに対する再送オーバヘッドがネットワークの輻輳を引き起こす原因となる。そのため、実ネットワーク環境でのパケットロスに対する FEC の有効性を評価しなければならない。

3. 誤り訂正コードによる帯域増加
   受信者の多い大規模なマルチキャスト通信では、誤り訂正コードによる帯域の増加より、再送オーバヘッドのほうがルータに与える負荷は大きい。しかし、帯域の増加による負荷もネットワーク輻輳の原因の一つであり、

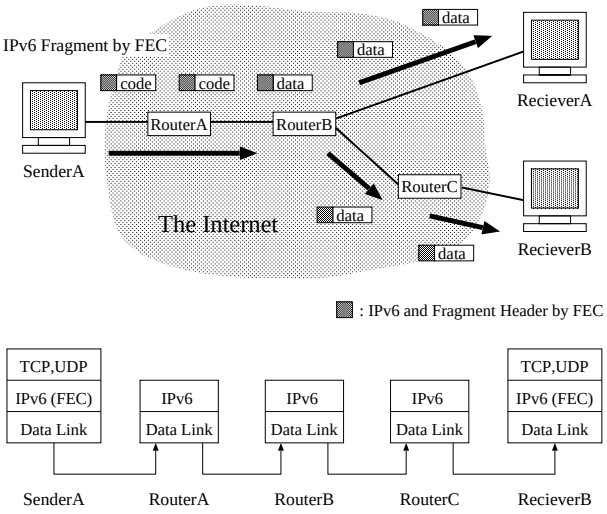


図 1: FEC の使用形態

ネットワークの特性に合わせた最適な誤り訂正コード数を求めなければならない。

4. データ品質の維持
   IP パケットを分割する手法は、ジッタを小さくすることにより高いデータ品質を保証できる。しかし、FEC は高負荷な演算処理を必要とするため、逆にジッタが大きくなる可能性がある。そのため、実データを用いたデータ品質の評価が必要である。

## 2.2 動作概要

FEC はエンドエンド間の通信で用いることを基本としている。使用形態は図 1 となる。送信ノードは実データ (図 1 では data) の他に、誤り訂正用コードを (図 1 では code) として送信する。IPv6 フラグメント機能の拡張として実装しており、実データパケットも誤り訂正用コードパケットも、途中ルータでは IPv6 フラグメントパケットとして転送される。したがって、転送途中ルータでは、誤り訂正コードの生成/復元の演算処理を行わない。

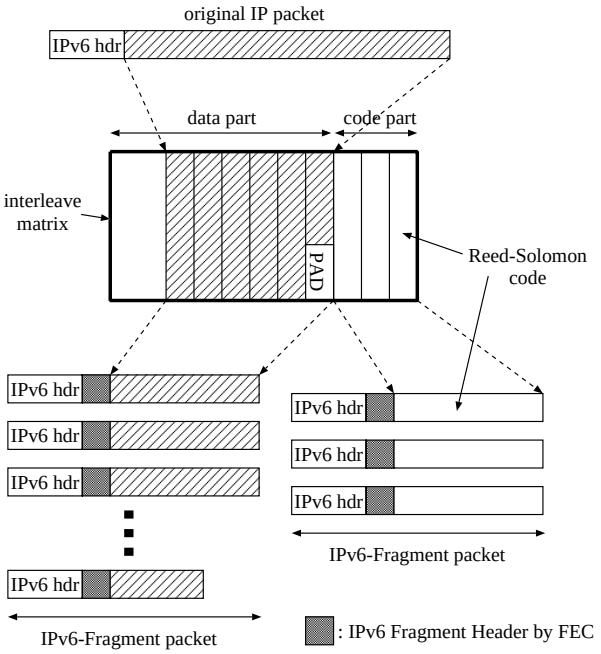


図 2: 送信処理

## 2.3 送信の動作

図 2 に送信ノードの処理手順を示す。interleave matrix のデータは、IPv6 フラグメントパケットとしてデータリンク層に渡す。

1. トランスポート層から得た Original IP packet の実データ部を Interleave matrix の data part へ縦方向に書き込み、Reed-Solomon 符合を code part へ生成する。このとき、data part の最終列にパディング (PAD) を行う。

2. 行毎に誤り訂正符号化の計算をし、code part の同じ行に書き込む。

3. data part の各列、code part の各列に対して、IPv6 ヘッダ (IPv6-hdr) を付与する。このとき、IPv6 フラグメントヘッダの Reserved フィールドに以下の情報を書き込む (図 3)。

   - Max Data pkt (7bit): その Interleave matrix が含んでいたデータ列数を示す。Max Data pkt フィールドは 0〜127 ま

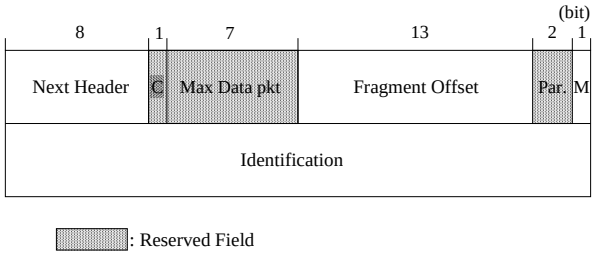


図 3: IPv6 Fragment Header Format

で表記可能なことから、IP パケットは 128 個までフラグメント可能。

- Par. (2bit): 各 Interleave matrix に対して何シンボルの訂正シンボルを付与しているかを示す。各Interleave matrix 毎に付与する訂正シンボル数が異なっていることも可能。本実装では訂正シンボル数は 1〜4 としている。
- C (1bit): そのペイロードがデータ (0)、誤り訂正コード (1) であるかを示す。

4. data part 最終列のパディング (PAD) を取り除く。

5. リアセンブルもしくは復元された IP パケットをデータリンク層に順次渡す。

## 2.4 受信の動作

図 4 に受信ノードの処理手順を示す。

1. データリンク層から得た IP パケットは、IPv6 フラグメントパケットとして扱われ、フラグメントキューに保持する。

2. data part が全て揃えば Original IP packet にリアセンブルする。code part の誤り訂正コードは不要なので廃棄する。

3. code part が到着しても data part が到着していないなら誤り訂正を行う機会を待つ[2]。

[2]実データをフラグメントしたパケット数を n、それに対して付与した誤り訂正コード数を m とするとき、受信ノードに到着した全パケット数が $n$ 以上であれば訂正可能な範囲である。パケットが n 個到着し Original IP packet が復元可能になった時点で復元を開始する。

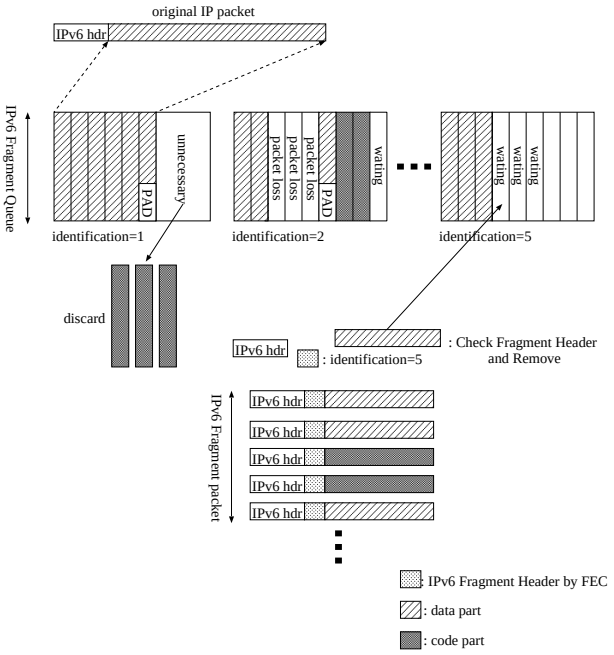


図 4: 受信処理

4. リアセンブル、もしくは、誤り訂正により復元した Original IP packet を順次トランスポート層へ渡す。

5. パケットロスが多く訂正能力を越えた場合は、IPv6 フラグメント機能のタイマーに合わせて、誤り訂正を中止しフレームごと廃棄する。

## 2.5 送信パケットの Reorder

実ネットワークでのパケットロスは、パルス的にロスするパターンと、複数個連続してロスするバーストパターンがある。FEC は数十個連続する長いバーストの復元は不可能であるが、数個連続するバーストに対しても誤り訂正能力は低い。しかしパケットの送信パターンを Reorder することにより、短いバーストに対しては訂正能力を維持できると考えられている (図 5)。本稿で実装した FEC は、図 5 の Reorder 機能をオプションとして実装している。

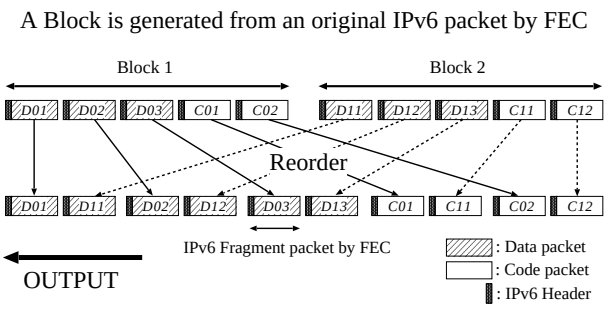


図 5: Reorder オプション

### 2.6 FEC パラメータの設定

本実装による FEC には、指定できるパラメータが存在し、アプリケーションやネットワークの特性に応じて変更できる。FreeBSD の kernel state を設定する sysctl コマンド上で設定可能である。

- sysctl -w net.inet6.ip6.fec_payload=96
  IPv6 フラグメントサイズを指定する。デフォルトは 96byte である。このサイズ変更によりパケットの分割数を制御できる。分割数が増加すれば、到着するパケット間のジッタを小さくできるが、処理負荷が大きくなる。

- sysctl -w net.inet6.ip6.fec_parity=1
  各 IP パケットに対して付与する誤り訂正コード数を指定する。本実装では 1〜4 を指定可能である。誤り訂正コード数を増やすことにより復元率は高くできるが、処理速度は遅くなり、使用帯域が増加する。

## 3 性能評価

以下の 3 つの項目を評価することで、2.1 節で挙げた 4 項目を評価した。

- 基本性能 (スループット・処理速度)
  無償で公開されている通信性能評価ツールである netperf [9] を用いてスループットを測定し、ping6 コマンドにより RTT(Round Trip Time) を測定し処理速度を評価する。

- パケットロスに対する復元率
  実ネットワークにおけるパケットのロスパターンをパルス、バーストの 2 種類に分類し復元率を測定することで、FEC のパケットロスに対する有効性を評価する。さらに、FEC は誤り訂正能力以上のバースト的なパケットロスに対して効果が低いが [2]、送信ノードにおいてパケット送信パターンを再配置 (Reorder) することで、誤り訂正能力を維持できることを示す。複数のパケットロス率に対して復元率を測定することにより、ネットワークのパケットロス率に合わせた付加する誤り訂正コード数の指標を得る。この指標により、冗長なコードによる帯域増加を避けることができる。

- データの再生品質
  データの再生品質として、実データとして Digital Video(DV) を転送し評価する。DV は処理負荷の高い大容量ストリームであるため、ルータ性能を同時に評価する。

送信ノード、受信ノードの計算機環境を表 1 に示す。IPv6 には KAME [10] を用いた。

表 1: 使用した主なソフトウェアとそのバージョン

| | |
|---|---|
| OS | FreeBSD 3.3-RELEASE |
| | KAME 19991108-snap |
| CPU | PentiumIII 500MHz |
| Memory | 256MB |
| NIC | Ethernet 100Mbps |

### 3.1 netperf による測定

実験環境として他の機器の影響が出ないように

- CPU 時間、メモリ使用量などを無制限にする
- 余計なプロセスを実行しない
- ２台の計算機をクロスケーブルで接続

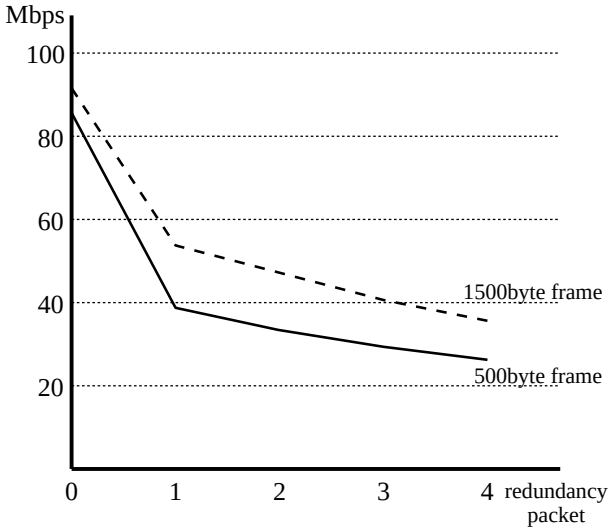


図 6: スループットの計測

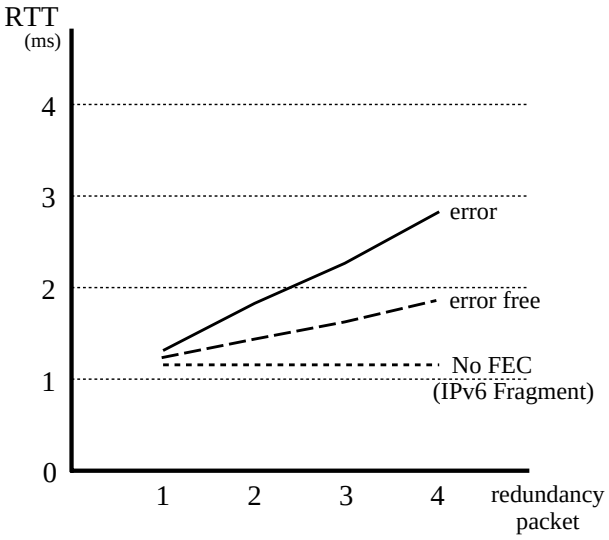


図 7: Round Trip Time の計測

とした。これにより、FEC を使用しない場合の限界性能を推定できる。

IPv6 を利用できるように変更したパッチ [11] を適用した netperf2.1pl3 を用いて、FEC を使用した場合、使用しない場合のスループットを測定した (図 6)。IPv6 パケットが 500byte と 1500byte の場合について測定し、各々の IPv6 フラグメントのペイロード長を 104byte、304byte とし、実データパケットを 5 個にフラグメントした。誤り訂正コードは 1〜4 個付与した。図 6 で誤り訂正コード数が 0 のときは IPv6 フラグメントせず、FEC を使用しない場合のスループットを示す。

500byte の転送において、FEC を使用しない場合、約 85Mbps 程度のスループットが得られる。FEC を使用した場合、26〜38Mbps のスループットが得られ、約 30〜45%に低下する結果となった。1500byte の転送において、スループットの低下率は若干減少した。したがって、FEC の演算処理によりスループットは大きく低下したため、大容量のトラヒックに対して性能劣化を引き起こす可能性がある。

### 3.2 ping6 による測定

実験環境は、1 台のルータを経由したプライベートネットワークで測定した。KAME に実装されている ping6 コマンドを用いて 1500 byte の IPv6 パケットの転送を行い、RTT(Round Trip Time) を計測することにより、送信時と受信時の演算処理性能を評価した。IPv6 フラグメントのペイロード長は 304 byte とし、データ数 5 に対し誤り訂正コードパケット数を 1〜4 とした場合の各々について測定した (図 7)。パケットロスは、パルス的なロスを発生する機能をルータのカーネル内に実装した。ルータ性能は、CPU400MHz、メモリ 128MB である。横軸を誤り訂正コードパケット数、縦軸を RTT とする。

- No FEC は、通常の IPv6 フラグメントの RTT を示す。パケットロスは発生させない。ただし、IPv6 フラグメントペイロード長は RFC2460 では PathMTU と規定されているが、本実験では 304byte と設定した。

- error free は誤り訂正コードを生成するがパケットロスは発生させない。つまり、受信ノードで FEC による復元は行われないため、誤り訂正コード生成の処理時間のみを示す。

- error は誤り訂正コードを生成しパケットロスも発生させる。つまり、受信ノードでFECによる復元を行なうため、誤り訂正コード生成およびFECによる復元の合計処理時間を示す。

図 7 より、誤り訂正コードを 4 個付与したとき、送信時の処理時間は 0.94ms であり、受信時の処理時間は 1.05ms である。これより送信処理より受信処理の負荷が高いことがわかる。エンドーエンド間でのスループットは、パケットロスがない場合は送信ノードの符合化の処理速度に依存し、パケットロスがある場合は受信ノードの復号化の処理速度に依存する。これにより、CPU 負荷の高いアプリケーションを本実装上で用いる場合は、受信ノードの計算機性能がボトルネックとなる可能性がある。

## 3.3 FEC による復元率の測定

パケットロスパターンとしてパルス、バーストを発生する機能を 3.2 節で用いたルータのカーネル内に実装した。復元率は、このルータを経由したプライベートネットワークで測定した。

### 3.3.1 パルス的なパケットロス

パルス的なパケットロスをランダムに1〜10%発生させた結果を図 8 に示す。分割データ数は 5 とした。

復元率は、実データパケット数を n、誤り訂正コード数を m、パケットロス率を p とするとき、n-1 以上受信ノードに到着する確率として以下の式で表される。

$$\sum_{i=0}^{m}\{\prod_{j=0}^{i-1}\frac{n+m-j}{i-j}\times(1-p)^{(n+m)}\times(\frac{p}{1-p})^{i}\} \quad (1)$$

図 8 は (1) 式に従った結果となった。これより、FEC はパルス的なパケットロスに対して、ほぼ完全に復元することができ、有効な誤り訂正制御方式であるといえる。

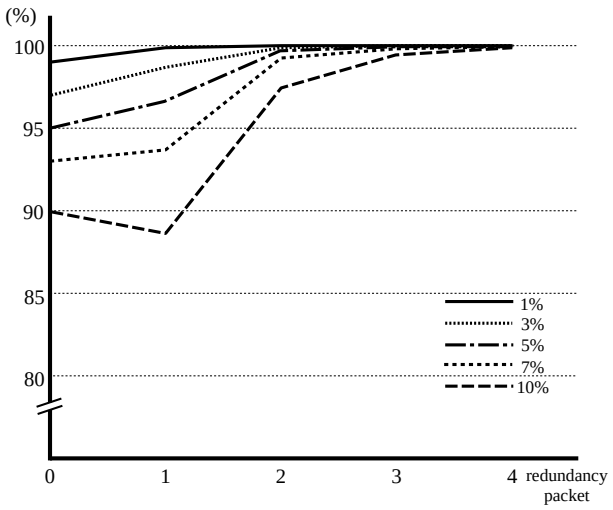


図 8: 復元率 (パルス)

### 3.3.2 バースト的なパケットロス

n 個連続してパケットロスするバーストが確率 p で発生するとき、トータルのパケットロス率は次の (2) 式で表される。

$$\frac{np}{(1-p)+np} \quad (2)$$

5 個連続してパケットロスするバーストをトータルのパケットロス率 5%で発生させたときの復元率を図 9 に示す。また、Reorder オプションを用いたときの復元率を図 10 に示す。

Reorder オプションなしの場合、FEC による効果がほとんどないことがわかる。これに対して Reorder オプションありの場合、復元率は図 8 と同程度の復元率となった。したがって、短いバースト的なパケットロスに対しても、Reorder することで復元率を高く維持することができる。

## 3.4 Digital Video 転送による評価

実験環境として、3.2 節で用いたルータを経由したプライベートネットワークで測定した。DV 機器を接続するために、IEEE1394 インターフェースを取り付けた送信・受信ノードを準備した。送信ノードには DV カメラを、受信ノードには DV デッキを接続し、DV カメラで撮影されている映

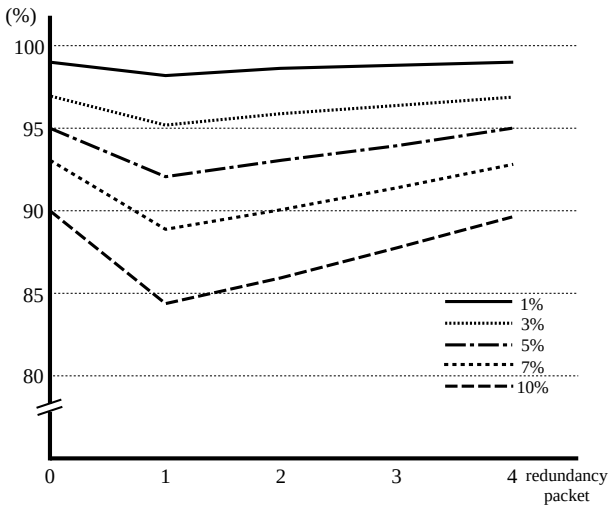


図 9: 復元率 (バースト、Reorder なし)

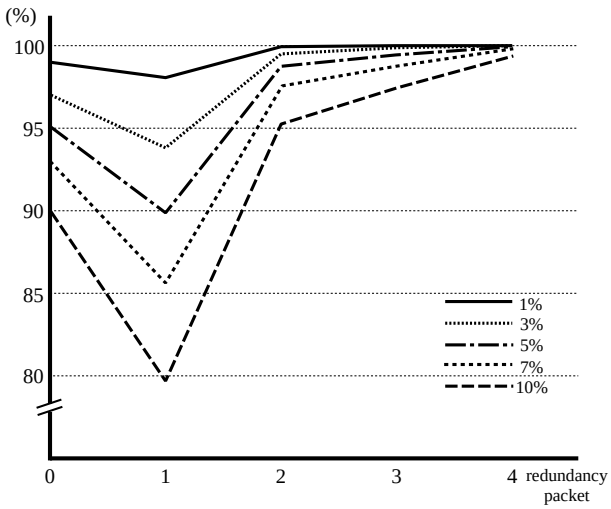


図 10: 復元率 (バースト、Reorder あり)

像を受信ノードに送信した。受信ノードでは、受信した映像を DV デッキを介しモニタに出力する (図 11)。実験に利用した FreeBSD には IEEE1394 のドライバを適用し、DV の送受信アプリケーションとして DVTS [12] パッケージを使用した。

- パルスパターンのパケットロスをランダムに 5%発生させた環境での結果を表 2 に示す。FEC を使用しない場合、ブロックノイズが多く表示され、音声も途切れが目立ち、聞きとりにくい。しかし、FEC を使用することにより、画質・音声ともパケットロスがない品質と同程度まで改善した。ただし、誤り訂正コードを 4 個付与した場合は、画質・音声とも急激に劣化した。これは、DV は約 32Mbps のトラヒックであるため、送受信ノードの処理負荷が高い。ここでは受信ノードの FEC 復元処理が DV の受信処理に追いつかないためと考えられる。このとき、誤り訂正コードを含めたトラヒックは約 50Mbps であった。

- バースト的なパケットロスをトータルで 5%発生させた環境で、Reorder オプションを使用したときの結果を表 3 に示す。Reorder オプションを使用しない場合は、画質・音声ともに劣化するが、Reorder オプションを使用することにより、画質・音声ともに改善した。しかし、誤り訂正コードを 3 個付与したとき画質・音声の劣化が見られ、4 個付与したときに急激に劣化した。これは、Reorder したことにより、受信ノードで保持するパケット数が増加し、その結果、メモリの消費による FEC 復元処理の性能劣化を引き起こしたためと考えられる。

表 2: パルス (loss 率 5%)

| 訂正コード | 画質ノイズ | 音声 |
|---|---|---|
| 1 | 多 | 聞きとれず |
| 2 | 少 | 途切れがなくなる |
| 3 | なし | 良好 |
| 4 | 多 | 聞きとれず |

### 3.5 実験結果に対する考察

実験結果として、次の 4 つの知見が得られた。

- FEC を使用したときスループットは 30〜45% に低下した。

- 本実装の FEC は、送信側より受信側の処理負荷が高く、受信ノードの計算機性能がボト

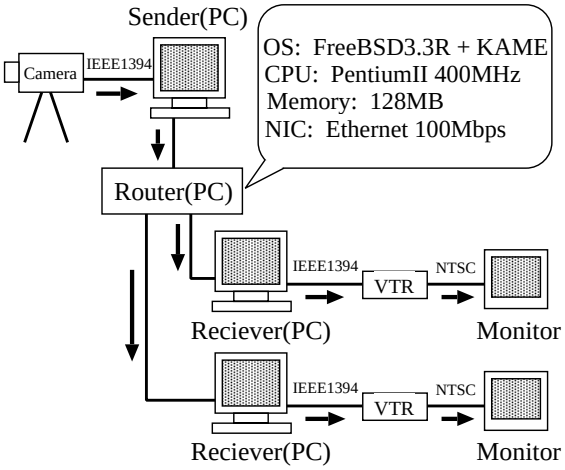


図 11: DV 転送実験環境

表 3: バースト (loss 率 5%、Reorder あり)

| 訂正コード | 画質ノイズ | 音声 |
|---|---|---|
| 1 | 多 | 聞きとれず |
| 2 | 少 | 途切れがなくなる |
| 3 | 少 | 途切れがなくなる |
| 4 | 多 | 聞きとれず |

ルネックとなる可能性がある。

- FEC は、パケットロスがパルス的に発生する場合、非常に高い確率で復元できる。また、ネットワークのパケットロス率に合わせた付与する誤り訂正コード数の指標が得られた。短いバーストがパルス的に発生する場合には、復元はほぼ不可能であった。しかし、Reorder オプションを用いることで復元率を高く維持することができた。

- 大容量トラヒックとして DV を転送した結果、画質・音声ともパケットロスがない品質と同程度まで改善した。DV のトラヒックが 50Mbps 前後となるとき FEC 処理の動作限界となった[3]。

[3]FEC 処理の限界による性能劣化と帯域の増加を避けるために、データの特性に合わせて選択的に FEC を適用することができる。ビデオ会議を目的とした通信システムにおいては画質より音声が重視される。そこで、音声と動画を UDP ポート番号により別ストリームとして生成する DVTS オプションを利用して FEC を音声にのみ適用した結果、トラヒックの増加は約 2Mbps となり音声の途切れはほぼ改善できた。

FEC はパルス的なパケットロスに対して有効である。さらに、短いバーストパケットロスに対してもパケットの送信パターンを Reorder することで復元率を維持できた。しかし、ネットワークが輻輳してきた時に発生する長いバーストパケットロスには、十分な誤り訂正ができない。訂正能力を高めつつ、ネットワークの輻輳を引き起こさないためには、付与する冗長パケット数、Reorder する Block 数などのパラメータの動的な調整が必須である。実際のネットワークの利用可能帯域を動的に知り、それに応じたレートでパケットを送信できるような機構を考える必要がある。

DV を転送する実験では、PC ルータを経由したネットワークで大容量トラヒックを発生させた。パケット数が増加することにより、ルータの動作に負荷がかかるという問題が懸念されたが、PC ルータは安定して動作し、FEC による実データの再生品質も高く維持できた。

## 4 今後の課題

RMTWG では、SRM [13]、PGM [14] などの Feedback 型のマルチキャストプロトコルと Open Loop 型の FEC を組み合わせ、受信者の多い大規模なネットワーク上で信頼性のあるマルチキャストプロトコルを実現していく Building Block 方式が提案されている。しかし、まだ実際に Feedback 型と Open Loop 型を組み合わせたマルチキャストプロトコルの実装例は少ない。そこで、本稿で実装した FEC とトランスポート層でのマルチキャストプロトコルとを組み合わせて実装し評価することで RMTWG の提案する Building Block 方式を検証したい。現状のマルチキャストプロトコルと FEC を組み合わせることで、パケットロスに対する再送オーバーヘッドを軽減し、受信者の多い大規模なネットワークで高い信頼性のあるマルチキャスト通信を実現することを目指す。

## 5 まとめ

受信者の多い大規模なマルチキャスト通信サービスを実現するためには、再送オーバヘッドを軽減できるOpen Loop型のFECが有効だと考えられている。しかし、FECは送信時に冗長なデータを生成し、受信時にパケットロスを復元する演算処理を行うため、通信性能が低下する。そこで、本稿では実際にIPv6上でFECを実装し性能を評価した。

その結果、通信性能としてスループットの低下が見られた。しかし、実トラヒックとして処理負荷が高く大容量トラヒックを発生するDVの転送が可能であり、画像・音声などのデータ品質に関して高く維持できた。

FECは、パケットロスがパルス的に発生する場合は非常に高い確率で復元することができる。短いバーストに対しても、パケットの送信パターンをReorderすることで復元率を維持することができる。しかし、長いバーストに対してはFECによる復元は不可能である。FECの誤り訂正力を維持するためには、実際のネットワークの状態にあわせたパラメータの動的な調整が必要である。

## 6 謝辞

本研究に際して、数々の有意義な助言を頂きましたWIDEプロジェクトのReliable Multicast WGの方々に感謝致します。

また、研究を進めていくうえで貴重な助言を頂きました奈良先端科学技術大学院大学の出水法俊氏、中村豊氏に感謝致します。

## 参考文献

[1] “IETF Reliable Multicast Transport WG”
URL:http://www.ietf.org/html.chaters/rmt-charter.html

[2] 金井, 松澤, 角田, 江崎,
“IP レベルの前方誤り制御プロトコル (IP-FEC) の実装と評価”,
インターネットコンファレンス’98 論文集, pp.129-137(1998)

[3] L. Rizzo,
“Effective erasure codes for reliable computer communication protocols”,
Computer Communication Review, Vol.27, no.2, pp.24-36, April 1997

[4] L. Rizzo, L. Vicisano,
“A Reliable Multicast data Distribution Protocol based on software FEC techniques”,
The Fourth HPCS’97 IEEE Workshop, June 1997

[5] B. Braden, D. Clark, J. Crowcroft, et.al.,
“Recommendations on Queue Management and Congestion Avoidance in the Internet”,
RFC2309, April 1998

[6] B. Whetten, L. Vicisano, R. Kermode, M. Handley, S. Floyd, M. Luby
“Reliable Multicast Transport Building Blocks for One-to-Many Bulk-Data Transfer”,
Internet Draft, March 2000

[7] M. Luby, L. Vicisano, L. Rizzo, J. Gemmell, J. Crowcroft, B. Lueckenhoff
“Reliable Multicast Transport Building Block: Forward Error Correction Codes”,
Internet Draft, July 2000

[8] S. Deering, R. Atkinson,
“Internet Protocol version 6 Specification”,
RFC2460, December 1998

[9] “The Public Netperf Homepage”
URL: http://www.netperf.org/

[10] “The KAME Project”
URL: http://www.kame.net/

[11] “The KAME Project, ftp site”
ftp://ftp.kame.net/pub/kame/misc/netperf-21pl3-20000721.diff.gz

[12] A. Ogawa,
“DVTS(Digital Video Transport System)”
URL: http://www.sfc.wide.ad.jp/DVTS/

[13] S. Floyd, V. Jacobson, S. McCanne,
“A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing”
Proc ACM SIGCOMM 95, Aug 1995 pp. 342-356

[14] D. Farinacci, S. Lin, T. Speakman, and A. Tweedly,
“PGM reliable transport protocol specification”
Internet Draft, Aug 1998