卒業論文　　2004年度 (平成16年度)

パケットフロー特性に着目した映像音声配信モデルの研究

慶應義塾大学 環境情報学部

氏名：松園 和久

指導教員

慶應義塾大学 環境情報学部

村井 純

徳田 英幸

楠本 博之

中村 修

南 政樹

平成16 年12 月29 日

卒業論文要旨 2004 年度 (平成 16 年度)

パケットフロー特性に着目した映像音声配信モデルの研究

本研究では，ネットワークを用いたリアルタイム映像ストリーミングシステムに特化したパケット伝送最適化機構の構築を行った．

リアルタイム映像ストリーミングをネットワークを用いて行う際の必要条件として受信側におけるバッファオーバーラン，アンダーランへの配慮と，ネットワーク伝送遅延の揺らぎ吸収の２点が挙げられる．

本研究ではネットワーク伝送遅延の揺らぎは長時間発生するものではなく，一時的なものが殆どであることに着目し，パケット伝送最適化モデルを提案し，モデルに基づいた設計と実装を行った．

本研究では，提案したモデルに基づき，４つのモジュールを作成した．受信側受信タイミング監視部では受信したパケットの到達間隔の計測を行う．計測された間隔は受信側レポート出力部によって送信側へと通知される．受信側バッファリング監視モジュールはアプリケーション層に存在し，アプリケーションの持つバッファ状況の監視，他モジュールへの通知を行う．送信側出力制御部では，受信側から通知されたレポートに基づいたパケット送信間隔の制御を行う．

本研究により，インターネットを介したリアルタイム映像転送システムの安定した送受信が品質を損なうこと無く可能となった．

キーワード
1. リアルタイム映像配信，2. バッファアンダーラン，3. パケット到達ジッタ , 4. フロー制御

慶應義塾大学 環境情報学部

松園 和久

Abstract of Bachelor's Thesis

Academic Year 2004

Research about distribution model of video and audio with paying attention to the packet flow characteristic.

In this research, packets forwarding optimization system for real time based video streaming on the network is constructed.

Care for "buffer over run" and "buffer under run" and assimilate the forwarding delay coming from the network are the requirements for real time based video streaming on the network.

This research focuses on that in most cases forwarding delay are temporary. Based on that point, this research proposes the packets forwarding optimization model and the system, which is based on the model, is designed and implemented.

In this research, 4 modules, which are based on the proposed model, is made. At the receiver side, "received timing monitor" module measures the interval for received packets. Measured interval timing is reported to the sender via "receiver side report module ". " Receiver side buffering monitor" located in application layer. It monitors the buffering situation and reports the information to other modules. "Sender side output control module" controls the packets interval, which is based on the receiver's report.

Through this research, real time based video streaming on the Internet without damaging the quality becomes possible.

Keywords :
1.realtime video streaming, 2.buffer underrun, 3.jitta of packet received flow control

Keio University , Faculty of Environmental Information

Kazuhisa Matuzono

目 次

図 目 次

表 目 次

第1章　序論

1.1　背景：IP ネットワーク上のリアルタイム映像・音声配信

バックボーンのみならず、エンドノードが繋がるデータリングの高帯域化に伴い、データサイズの大きなデータを送受信することが用意となった。

これにより、ストリーミング放送や、VoIP と言った映像・音声を用いたコミュニケーションが活発化している。特にテレビ会議や、ライブ配信などの場合、受信側で再生される品質だけでなく実時間性を考慮しなければならないケースが存在する。

またネットワークに繋がるエンドノードは従来 PC のみであったが、現在では情報家電、携帯端末などが混在するようになり、エンドノードの計算機資源格差はより大きなものになっている。そのため、計算機資源が潤沢な環境だけでなく、乏しい環境をも想定した上でリアルタイム高品質映像配信システムが求められている。

1.2　映像・音声配信における問題点

IP ネットワークを介した通信では、利用可能な帯域幅の変動によってパケットロス、伝送遅延の揺らぎが発生する。リアルタイム映像・音声配信では、パケットロス、伝送遅延の揺らぎは映像の移り方に悪影響を及ぼす。特に、計算機資源が限られている組み込み専用機器などでは、伝送遅延の揺らぎによる映像の写り方の悪影響は大きい。

しかし、現在の映像・音声配信技術は、輻輳状態によるパケットロスを防ぐ機能しかない。様々な再生環境で動作するメディアアプリケーションが混在する現在では、受信側の計算機資源を考慮した映像・音声配信システムを構築する必要がある。

1.3　本研究の目的と意義

本研究では、リアルタイム映像・音声配信システムを対象としており、受信側の計算機資源に応じた信頼性の高い高品質映像・音声配信システムの構築を目的とする。

送信側は受信側のフィードバック情報を基にパケットのフロータイプを変化させる。End to End モデルを前提としたネットワーク状況に対して効率的なパケット転送スケジューリングの構築を目標とする。

本研究では、ネットワークの帯域変化がパケット到達ジッタに現れることに着目した。送信側は受信側と協調し、受信側のパケット到達ジッタ、バッファリング状況に応

じてパケットの送出間隔を変化させる。フロータイプを変化させることで、受信側のパケット到達ジッタをできるだけ押え、バッファリング量を制御する。受信側の計算機資源の差異に関係することなく、リアルタイム映像配信の信頼性を向上させることが本研究の意義である。

1.4 本論文の構成

本論文は第2章において、映像・音声配信における要素技術を述べる。第4章において既存のフロー制御手法、帯域推測手法について述べる。第 4 章では第 3 章で述べた問題点を解決する手法を述べる。第5.1章において本システムの設計に関して述べ、第6章で実装に関して述べる．第7章で本システムの実装で既存の問題を解決したか否かを評価する．最後に第8章で本研究のまとめを行う．

第2章　本研究の技術的要素

本章では，本研究の要素技術として、IP ネットワーク上における映像・音声配信技術、TCP を例とした通信のフロー制御について述べる。

2.1　IP ネットワーク上における映像・音声配信

インターネットを介した映像配信は、１）映像データをすべて受信した後に再生を行なう（ダウンロード）方式と、受信したデータを逐次再生する方式（ストリーミング）方式の２つに大別される。図 2.1にダウンロード再生方式の概要図、図 2.2にストリーミング再生方式の概念図をそれぞれ示す。リアルタイム映像・音声配信には、ストリーミング再生方式が用いられる。

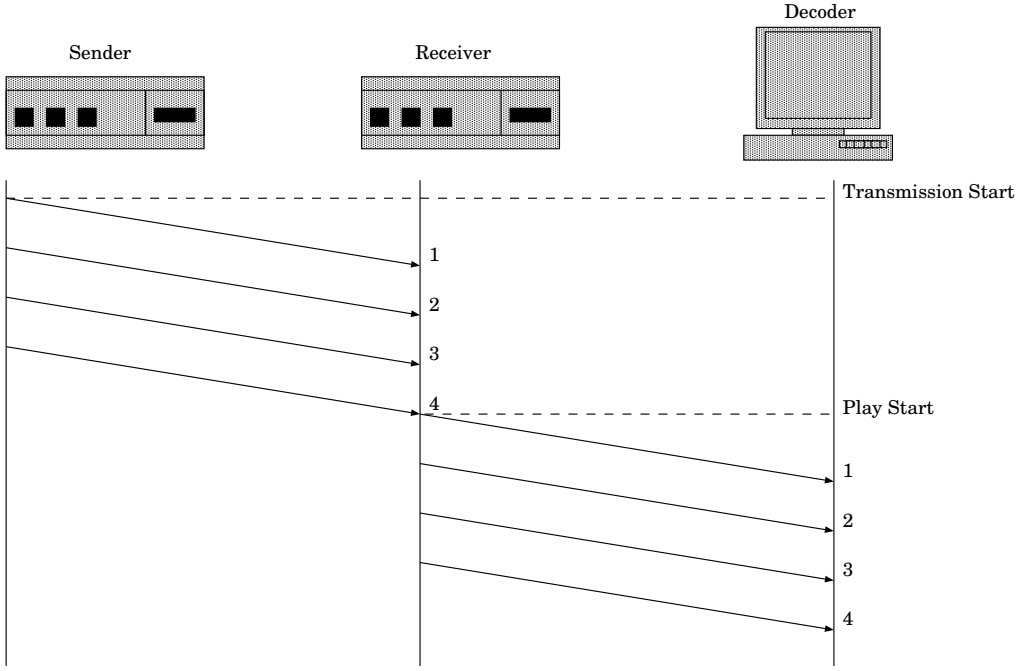

図 2.1: ダウンロード再生方式

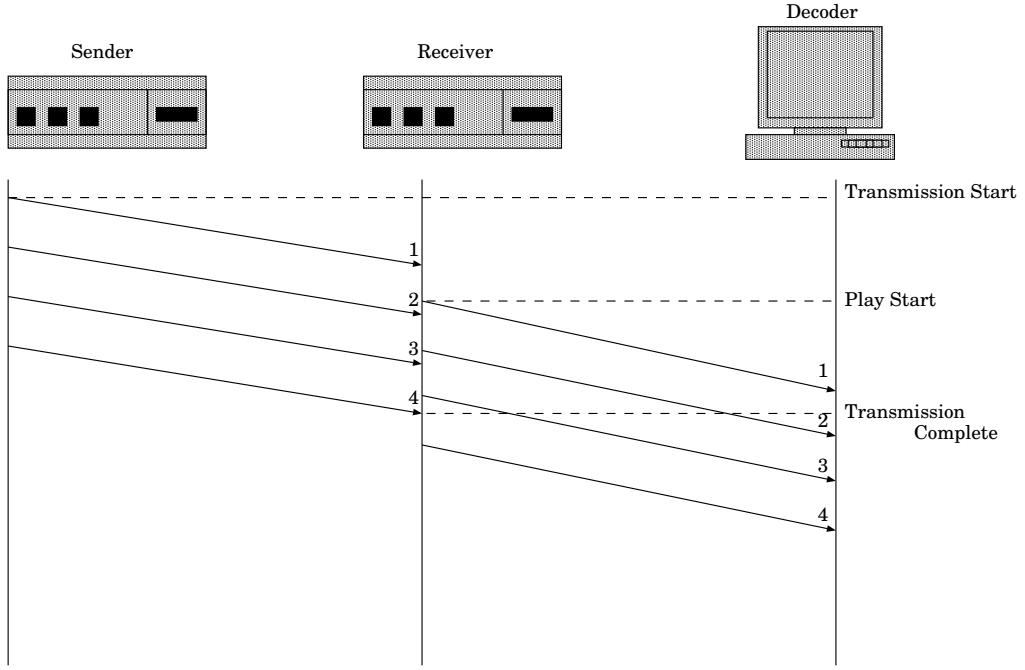

図 2.2: ストリーミング再生方式

2.2 ストリーミング再生方式の特徴

IP ネットワークはパケット交換方式かつ、ベストエフォート型のネットワークであるため、到達性、実行帯域幅、パケット到達時間、データの完全性は保証されない。しかし、ストリーミング再生において、映像の途切れが生じることなく視聴可能とするためには、クライアントが要求する時間内に適切な映像データが必要である。そのため、パケットの到着時間の揺らぎ (ジッタ) は受信側の再生品質に悪影響を及ぼす場合がある。ストリーミング再生方式におけるジッタの発生と再生との関係を以下の図 2.3に示す。

前半部ではジッタが発生することなく、一定の時間でパケットが伝送され、受信側では映像・音声の乱れが生じることなく再生できる。しかし、ジッタが発生することにより、途中から送信されているパケットは伝達時間に開きが生じている。ストリーミング再生では、映像データは逐次消費されていくため、映像データは必要とする時間内（デッドライン）に到着する必要があるが、この場合はデッドラインに間に合わず、受信側で適切に処理がなされていない。このように、デッドラインに間に合わない映像データは再生に利用されず破棄され、受信映像・音声の乱れにつながる。

2.2.1 バッファリング技術

パケットの到着時間の揺らぎ問題を解決するため、受信側がバッファリングを利用する手法がある。予め再生前に映像データをバッファリングし、逐次消費することで

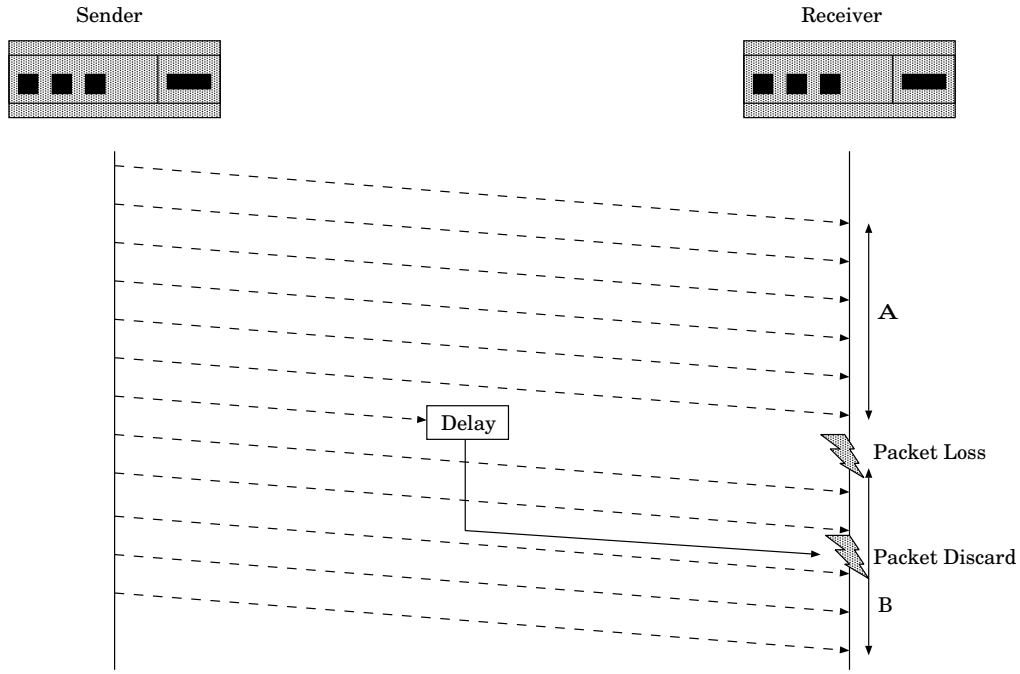

図 2.3: ジッタの発生と再生

パケットの遅延や揺らぎ、順序の不整合を軽減できる。図 2.4にパケットの到着間隔の時間関係を補正する仕組みを示す。

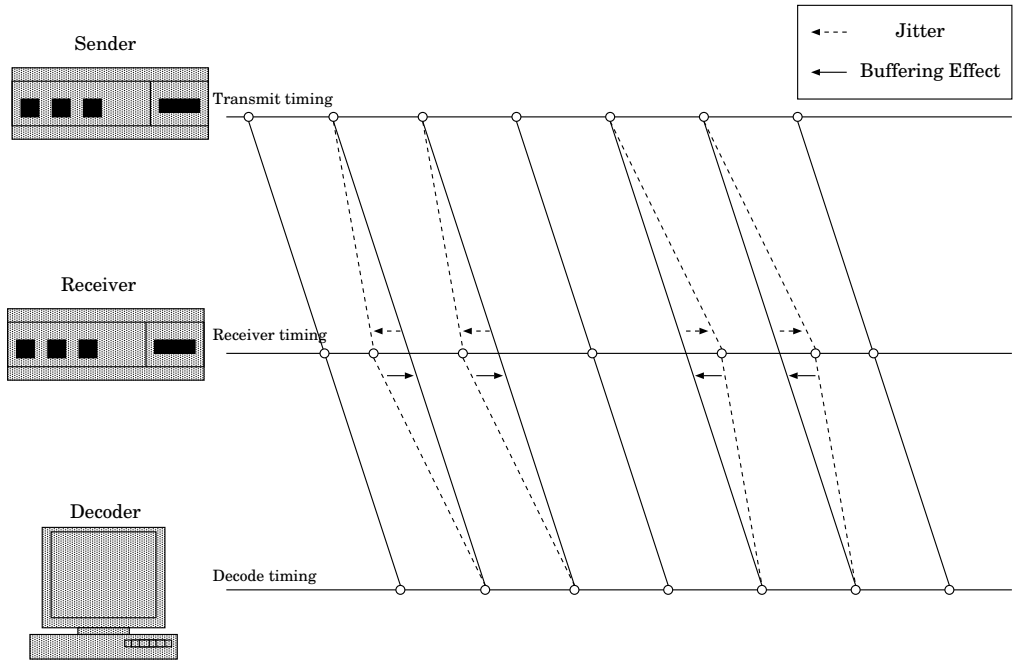

図 2.4: バッファリング技術

バッファリングの量を多く確保することによって、パケットの到着時間の遅れを吸収できる。しかし、バッファリングの量を多く確保すれば、その量に比例して再生までに時間がかかるため、実時間性は損なわれる。

2.2.2 ストリーミングの平滑化

ストリーミングを行なう配信サーバでは、CPU の負荷軽減、受信側のバッファリング量を増やすために、連続するパケットをバースト的にネットワークに出力する方式がとられる場合がある。例として、Windows Media[4] があげられる。しかし IP ネットワークにおいて、バーストトラフィックは、転送に必要な帯域幅の拡大や、他のトラフィックとの帯域競合を起こす要因となる。そのため、受信側までのネットワーク帯域幅が必要帯域幅より低い状態でバースティにパケットを送信した場合、途中経路でのルータの処理が追い付かず、パケットロスやジッタが発生して受信側での映像・音声の乱れが発生する。

この問題を解決するため、送出するパケットの出力間隔を調整し、ストリームデータの平滑化を行なう手法 [5] がある。ストリームデータを平滑化することで割り当て帯域幅を効率化し、パケットロス、ジッタを解消したストリーミングを行なえる。

2.3 IP ネットワーク上における通信のフロー制御

IP ネットワーク上ではベストエフォート型の通信が行なわれるため、パケットの配送に対する保証はない。また自立的な輻輳制御がない。そのため、輻輳を回避するには、エンドノード間で何らかの手法で経路ネットワーク状況を把握し、それに応じたフロー制御を送信側が行なう必要がある。　ここで、インターネットにおけるフロー制御手法の例として TCP について述べる。

2.3.1 TCP のフロー制御機構

TCP は効率の良いデータ転送を行なうために、スライディングウインドウ方式を用いて通信を行なう。送信側の TCP はパケットを転送した確認応答（ACK）を待ち、確認応答が来ないパケットを再送する。しかし 1 つのデータパケットを転送し、そのパケットに対する確認応答が来るまで新しいパケットの転送を待っていては帯域の有効な理由ができない。そのため、ウインドウと呼ばれる送信確認を待たずに転送できるパケットの範囲を設定する。ウインドウ内にあるパケットは、送達確認の到着を待たずに転送が行なわれる。従ってウインドウサイズが大きい程確認応答を待たずにデータ転送が行なえるため、高い転送速度で通信を行なうことができる。

TCP の転送速度は、ウインドウサイズにより決定されるが、このウインドウサイズは、送受信側のそれぞれの TCP がコネクション毎に保持するバッファスペースである。一般的な実装では、このバッファスペースは OS 上のカーネル内のメモリを利用してい

る。バッファのスペースは計算機毎によって限られているため、受信側は送信側に対し現在のバッファスペースの空き容量を通知する。この操作はウインドウ通知と呼ばれ、TCP のパケットヘッダのウインドウサイズフィールドの中に適切なウインドウサイズの値を指定し、データパケットや確認応答の送信時に送信側に伝達される。ウインドウ通知を受信した送信側は、ウインドウサイズの値を通知された値に変更し、送信側のデータ転送速度が受信側のバッファ範囲を越えないようにフロー制御を行なう。このフロー制御により、送信側と受信側の処理速度に違いを吸収できる。

2.3.2 TCP の輻輳制御

TCP における輻輳制御技術は、終端システムによるフロー制御技術を利用して実現されている。経路ネットワークで生じる輻輳に対処するため、ウインドウサイズに加えて、新に輻輳ウインドウと呼ばれる変数を管理する。TCP の送信ウインドウサイズは、受信側の通知するウインドウサイズと、輻輳ウインドウサイズを比較し、値の小さい方を用いる。　TCP の輻輳制御は 2 段階に分けることができ、2 つの通信段階はそれぞれスロースタート、輻輳回避と呼ばれる。
TCP は新しい通信を開始する場合、輻輳ウインドウを徐々に上げていき、使用帯域幅を上げていく。輻輳を検知した後、輻輳ウインドウサイズを減少させ、データの送出量を抑え輻輳を回避する。輻輳発生後の輻輳ウインドウサイズは TCP のバージョンによって異なる。輻輳ウインドウを徐々にまた上げていき、TCP のウインドウサイズを通信経路に最も適した大きさに近づけていく。ウインドウサイズが通信経路に適した大きさに近づくと、TCP は平衡状態に入り安定した通信を行なうことができる。この状態はセルフクロッキングと呼ばれる。図 2.5にセルフクロッキングの概念図を示す。

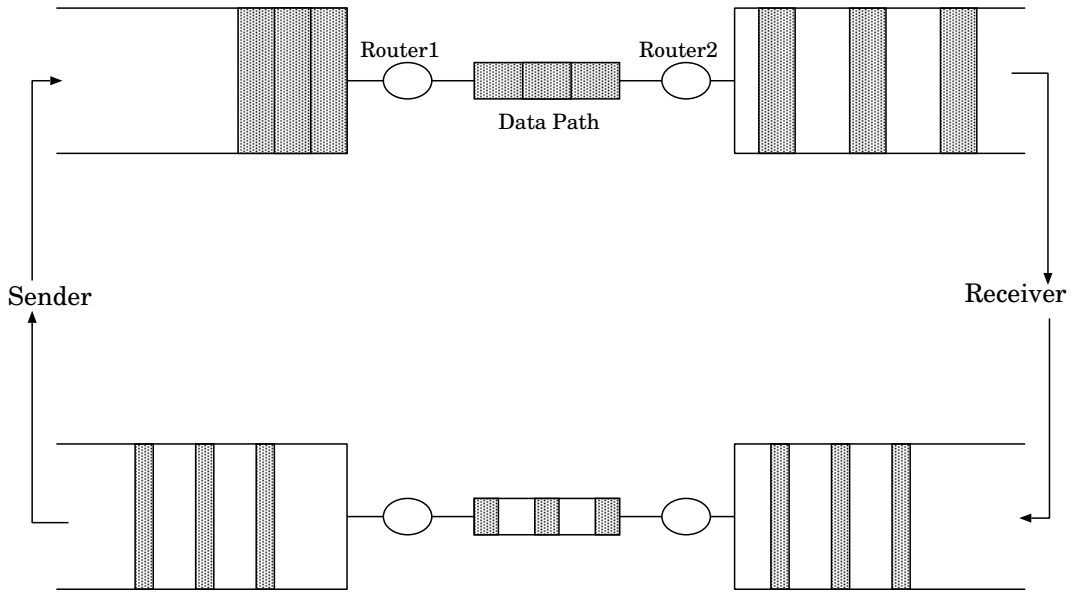

図 2.5: セルフクロッキング

連続するパケットが途中経路におけるボトルネックリンクを通過した場合、パケット間の距離が開く。一旦間隔が開いたパケットは、その後実行帯域幅が大きいネットワークを通過しても間隔が縮まることはない。従って受信側に伝送されたパケットの間隔は、ボトルネックとなるリンクの帯域幅に適合するものになる。確認応答は遅延確認応答アルゴリズムを使用しない場合、パケットの到着と同時に伝達される。そのため、確認応答の間隔はボトルネックリンクの帯域を示すことになる。
送信側の TCP は、確認応答を受信するとウインドウがスライドし、新しいパケットを転送する。そのため、受信側から送信された確認応答の間隔がパケット転送のトリガとなる。この循環により、TCP のパケット転送トリガはボトルネックリンクの帯域幅に適合し、安定した通信を行なえる。

2.4 本章のまとめ

本章では本研究の要素技術について述べた。IP ネットワーク上での映像・音声配信の概要として、ストリーミング技術の特徴について述べた。通信のフロー制御の例として、TCP におけるフロー制御の特徴について述べた。 第 3 章では、リアルタイム映像・音声配信におけるフロー制御、帯域予測手法の既存技術について述べる。

第3章　既存技術の特徴と問題点

本章ではIPネットワークを介した映像・音声配信におけるフロー制御の特徴を述べ、その問題点についてまとめる。

3.1　映像・音声配信におけるフロー制御

複数のユーザがネットワークの帯域を共有するパケット交換方式のIPネットワーク上では、利用可能な帯域幅が動的に変化する。インターネットを介した映像・音声配信には、経路ネットワークの動的な帯域変動に対応し、再生品質を維持するための幾つかのフロー制御技術がある。その特徴について述べる。

3.1.1　選択ビットレート

異なるビットレートに対応したファイルを予め用意し、ネットワークの状況に応じて動的に選択することによって使用帯域を調整する技術がある。以下に2つの技術について述べる。

- マルチビットレート
  マルチビットレートはビットレートの異なる複数のストリームを一つのファイルに同時にエンコーディングしたものである。ネットワークの状況を送信側と受信側でネゴシエーションを行ない、クライアントの利用可能な帯域に合わせたビットレートを選択する手法である。但し、専用のストリーミングサーバを必要とする。
  エンコーディングに時間を要するため、実時間性はなく、オンデマンドに視聴する蓄積配送型に対応した技術である。例としてSure Stream [9]が挙げられる。Sure Streamではネットワークの輻輳を検知した場合、動的に下位のビットレートへ自動で切替える機能を有する。しかし、輻輳を検知しビットレートを落す際に、受信側のバッファが枯渇し、再バッファリングが行なわれる場合がある。その場合、一時的に映像・音声の途切れが生じる。

- 階層符合化
  階層符合化とは動画層ストリームを幾つかの階層に分割して符合化する手法である[10]。低ビットレートで符合化したストリームを基本階層とし、原画像との差分情報に応じて次のビットレートで再び符合化する。差分情報に応じて繰返し符

合化を行なうことで階層を積み上げていく。
ネットワークの帯域に余裕がある場合は、全ての階層を受信することによって、原画像のビットレートのまま視聴できる。帯域に余裕がない場合、上位階層の送信は行なわず、下位層の送信を行ないビットレートの調整を行なう。複数の階層を用意することで、転送レートが異なった配信ができ、ネットワークの帯域幅と対応させることができる。
しかし、この方式は符号化/復号器を送信側と受信側が必要、符号化に時間を要するのに加えて、基本階層のデータが欠損された場合、上位階層も正しく復号できず快適な視聴ができない。

3.1.2 フレームレート制御

経路ネットワークの帯域幅の減少に対し、使用帯域幅の削減を行なう手法の一つとしてフレームレート制御がある。この方式を用いた代表例として DVTS[11] が挙げられる。
DV フォーマットはフレーム内圧縮方式を用いており、映像フレームが全て送信されなくても映像の再生が可能である。そのため、多少のデータの欠落は視聴をする上でのストレスにはならない。各フレームの画質を下げることは不可能であり、フレーム間引き機能だけを有する。音声データはノイズや音飛びの原因となるため、削減の対象とはしない。図 3.1フレームを間引いたストリームを示す。

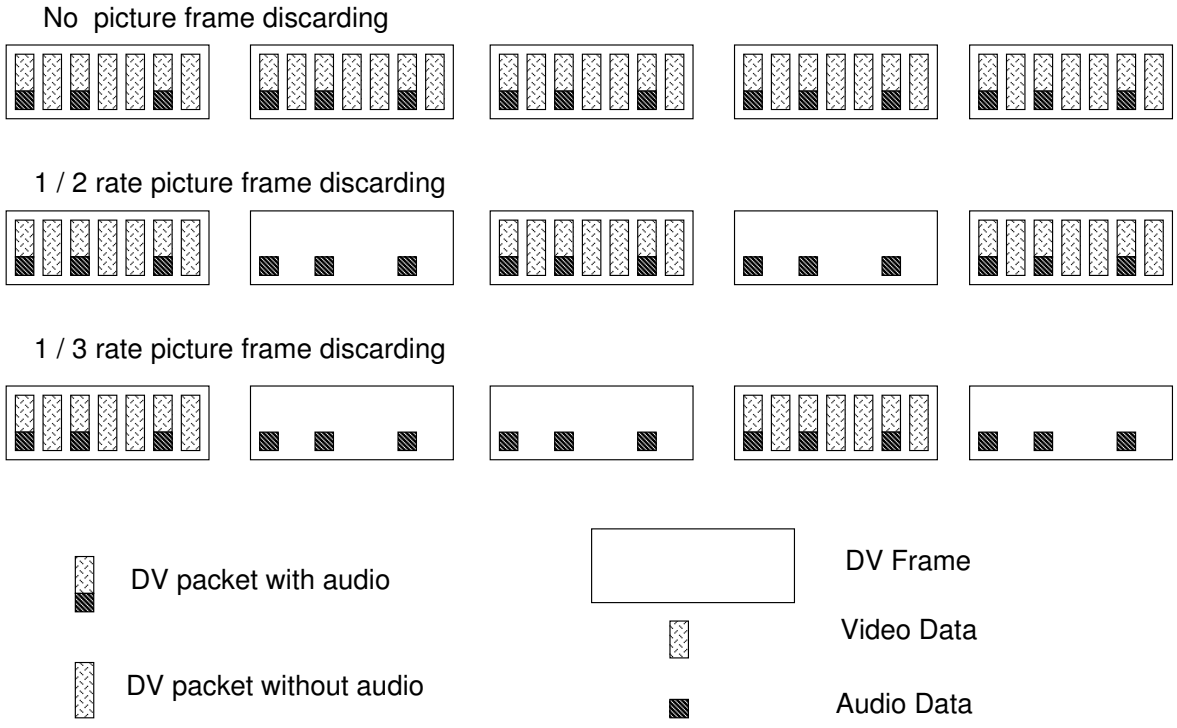

図 3.1: フレーム間引き

フレーム間引きを行なわないストリームと間引き率 1/2 のストリームを比較すると、映像データがあるフレーム数が半分になっている。DVTS ではフレームのデータが、フルフレーム時と比較して不足した場合、一つ前の映像フレームデータを再利用することによって、ノイズやバッファアンダーフロウを防ぐことができる。しかし、動きが多いシーンにおいては映像の滑らかさはなくなってしまう。

フレーム間引き機能に似た例として I ピクチャを減らす手法がある。MPEG2 フォーマットなどのフレーム間圧縮方式は、動き保証と両方向予測に基づいた圧縮を用いており、前後のフレームの差分から、現在のフレームの復号を行なう。
フレームの種類には、復号の基準となる I ピクチャと予測を行なう P ピクチャ、B ピクチャがある。これらの複数のフレームの集まりを GOP と呼ぶ。
転送レートは GOP1 つに対してフレーム数を増やすことで減少できる。GOP 1 つに含まれるフレーム数を増やし、予測を行なわないフレームである I ピクチャに対して、予測を行なうフレームである P ピクチャ、B ピクチャの数を増やすことで圧縮率を高めることが可能である。
しかし、動きの大きな部分ではデータ量が多くなり、バースト転送を行なう問題がある。また、I ピクチャを減らすことにより、映像の画質、滑らかさの極端な劣化を招く恐れがある。

3.2 帯域推測手法

TCP の輻輳制御アルゴリズム、再送処理は受信側が必要としている映像データの時間内への到達性まで保証していない。そのため、リアルタイム映像・音声配信は実時間通信に適した UDP を用いる。UDP を用いた End to End モデルでの帯域推測手法として、RTP/RTCP（Real Time Protocol/Realtime Transport Control Protocol) [7] の利用がある。

3.2.1 RTP/RTCP

RTP はリアルタイム性を重視した アプリケーション層におけるデータ転送プロトコルである。トランスポートプロトコルでやり取りするデータをアプリケーションが解釈できる単位で扱い、配信処理の細部はできるだけトランスポートプロトコル以外で対処する手法である。

RTP はパケットの順序番号、タイムスタンプなどの情報を付加する。そのため、受信側は映像の乱れに起因するパケットの到達順序、喪失、伝送遅延を計測することができる。しかし、RTP は情報を付加するのみであり、途中ネットワーク、受信側の状況に応じた対応は行なわない。　受信側が RTP セッションに対して得られた、パケットロス率、伝送遅延などのネットワーク状況を送信元に伝える手法として RTCP がある。RTCP によって定期的に得られるネットワーク状況を基に、送信側は転送レートなどを対応させることが可能である。

3.2.2 中継ノードの利用

経路ネットワークにおける中継ノードがネットワークの輻輳を検知し、受信側に対して通知する例として ECN(Explicit Congestion Notification)[12] がある。経路ネットワーク上におけるパケットの損失の多くは、ボトルネックとなる中継ノードのキューイングに洩れることに起因する。そこで、中継ノードがキューの状態を管理し、転送待ちパケット数がある一定のレベルになったら転送待ちのパケットの CE ビットを設定し、受信側がパケットの CE ビットを検査することでネットワークの状態を把握する。受信側は送信側にレポートすることで、輻輳状態になる前に送信側は転送方式を対応させ、パケットロスを解消できる。

3.3 既存手法の問題点

前節では、映像・音声配信におけるフロー制御手法、ネットワーク状況取得手法について述べた。ここで、それらの手法を用いた場合の問題点を述べる。

3.3.1 ネットワーク状況把握の重要性

第 2 章ではストリーミング配信において、映像データはデッドラインが確保されなければならないことを述べた。デッドラインを守る通信を行なうためには、ネットワーク状況を早く正確に検知しなければならない。その理由について述べる。

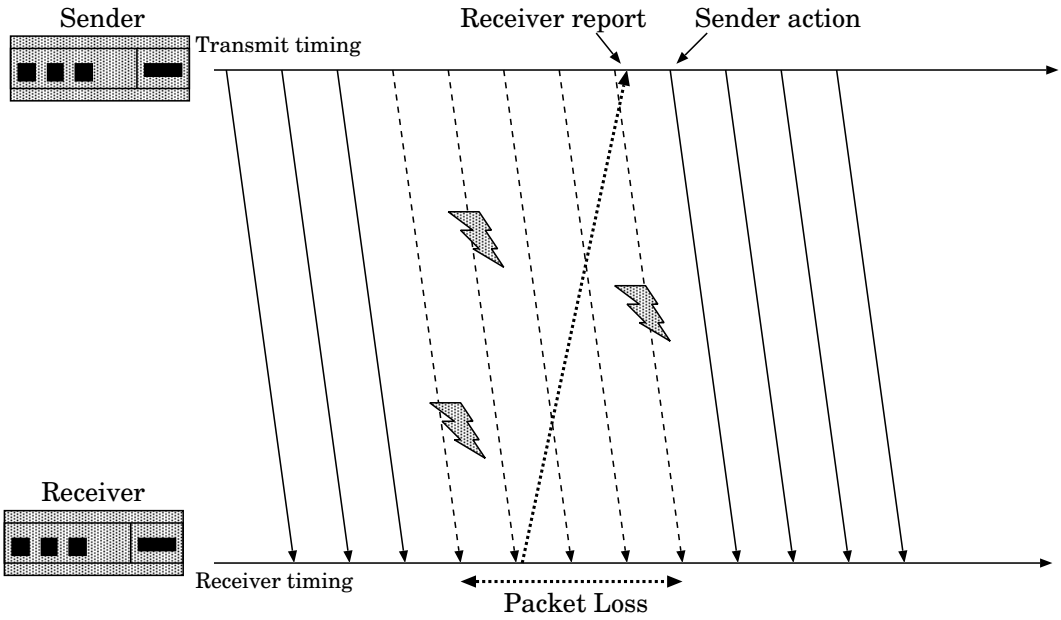

図 3.2: パケットロスによる送信側のトリガーアクション

実ネットワーク上において、実際にパケットロスが生じる場合は瞬間的なものが多

い。慢性的な輻輳状態に陥り、数秒間受信側にパケットが届かない状況は稀である [13]。また、パケットロスは連続的に発生する傾向があることが確認されている。

図 3.2では、パケットロスが発生し、受信側が送信側にレポートしている。そのレポートを受けとった送信側はネットワークの異変を感知し、送信手法を対応させている。しかし、対応を行なうまでに送信された映像データは受信側に届いていない。そのため、受信側の映像の乱れを引き起こす。従ってパケットロスが生じる前に送信側はネットワークの異変を検知し、対応させる必要がある。

第 2 章において、経路ネットワークにけるパケットの転送時間が変化するのは、ボトルネックリンクとなる中継ノードのキューイング処理, 経路変更が生じた場合に起きることを述べた。パケットロスが生じる状況の場合、キューの中は各コネクションのパケットで満たされてるため、大きなジッタの変化が前後で見られる [?]。そのため、ネットワークの状況を早く正確に把握するためには、パケットロス率だけでなく、ジッタの計測が重要になる。

ネットワークの状況をいち早く検知するためには、実際にボトルネックとなる中継ノードが通知を行なう ECN が適切である。しかし、これは中継ノード全てがネットワークの輻輳状態に応じたパケットの TOS フィールドが必要である。そのため、汎用性に欠ける。　しかし、RTP/RTCP を用いてジッタを計測するには、以下の問題点がある。

- 計測時におけるトラフィック量の増加

  RFC1889[7] では、RTCP パケットによるバーストトラフィックを防止するために RTCP パケットの送信間隔は、コントロールトラフィックは小さく、かつ既存のセッションバンド幅の一部程度に制御すべきであると述べている。ネットワークの帯域幅は常に変化していくため、ネットワーク状況を正確に把握するためには、粒度の細かい間隔でジッタを観測する必要がある。しかし、ジッタを毎回計測するために、RTCP SR パケットを定常的にネットワークに送出するのは効率的とは言えない。RTCP SR を利用しない手法として、RTP パケットのタイムスタンプの利用がある。しかし、時間粒度が荒いため、時間粒度の細かい単位でジッタを計測できない。

- アプリケーション層での遅延測定誤差

  RTP パケットのタイムスタンプ、もしくは、RTCP パケットを利用し、ジッタを計測する場合、アプリケーションがその処理を実行する。しかし、割り込み処理などにより、実際にパケットが NIC (Network Interface Card) に到達した時間とアプリケーションに到達する時間は大きな遅延時間が発生する場合がある。そのため、ネットワークの状況を正確に把握できない場合がある。

3.3.2 時間粒度の細かいフロー制御の必要性

前節において、ネットワークの状況把握にはジッタの計測が重要であることを述べた。しかし、現在におけるリアルタイム映像・音声配信のネットワーク状況把握の指標はパケットロスを用いたものが多い。例として TCP-Friendly[15] を用いた UDP フローが挙げられる。

パケットロスを指標として、映像品質を落すことで転送レートを制御するといったトリガーは、輻輳状態に陥っている場合は適切である。輻輳状態においては、極端に実行帯域幅が下がるためである。しかし、前節に述べたようにパケットロスは瞬間的なものが多い。フレームを間引くといった手法は極端に転送レートを下げるため、ネットワークの帯域を有効活用できない。そのため、粒度が荒い手法といえる。また、ジッタの計測によってフレームを間引くといったトリガーを使用した場合もネットワークの帯域を活用できない。図 3.3にその挙動を示す。

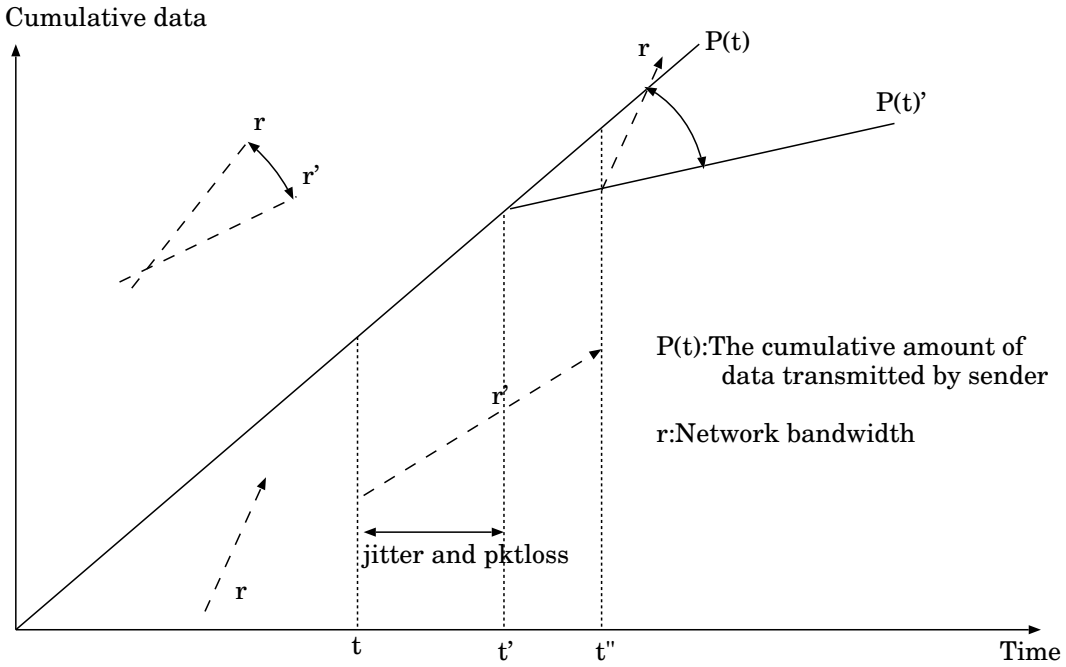

図 3.3: フロー制御と実行帯域幅の関係

時間 t に実行帯域幅は r から r' に変化している。時間 t' に実行帯域幅は r' から r に戻ったとする。送信側が時間 t までに転送したデータ量の合計を P(t) とすると、関数 P(t) の傾きが送信側の転送レートになる。時間 t' に受信側と協調し、送信側が映像品質を下げ、転送レートを制御した場合、直線 P(t)' になる。しかし、実行帯域幅 r' よりも極端に下げ過ぎている。また、直後に実行帯域幅が r に戻っているが、転送レートはそのままのため、ネットワーク資源を有効に活用できていない。

ジッタによって転送レートを調節するには、フレームを間引くといった極端に転送レートを下げる手法は適さない。そのため、フレームを間引きと比較して、転送レー

トを落さない時間粒度の細かいフロー制御が必要である。

また、時間粒度の細かいフロー制御は、計算機資源の限られた受信者を対象とした場合も必要である。計算機性能、バッファリング量が制限されているこのような組み込み機器は、受信するフローパターンが再生品質に大きく関与する。ある一定までのジッタは吸収できるが、それ以上のジッタは吸収できず、映像データは破棄される。

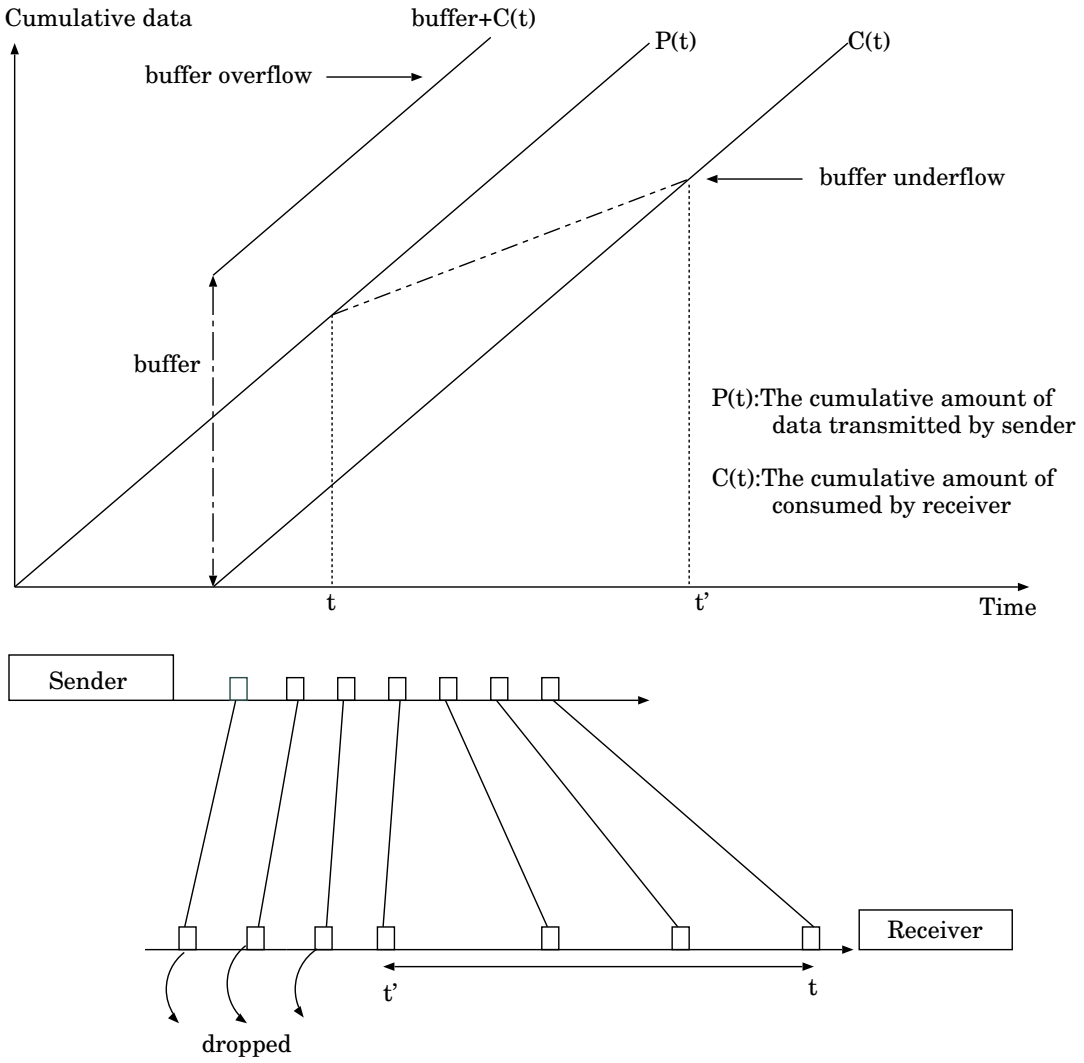

図 3.4: バッファアンダーランの様子

図 3.4では、時間 t に帯域幅減少によるパケット到達ジッタが発生している。パケット到達ジッタの発生により、受信側での再生消費量が減少し、バッファアンダーランが起きている。バッファアンダーランを起こした場合、デッドラインに間に合わなかったパケットは破棄されてしまう。

これらの機器を対象として配信する場合、 送信側は受信側の限られたバッファ内でフロー制御を行なう必要がある。また、ネットワークの状況に応じてはジッタが発生

するため、時間粒度の細かいフロー制御を行ない、受信タイミングを調整する必要がある。

3.4 本章のまとめ

本章では、IP ネットワークを介した映像・音声配信のフロー制御手法、フロー制御を行なう際の帯域推測手法について述べた。また、その手法を用いた場合の問題点を述べた。

次節では、問題点を解決するための本研究のアプローチを述べる。

第4章　パケットフロー特性に基づく映像・音声配信機構の提案

第３章では、リアルタイム高品質映像・音声配信における問題点をまとめた。本節では, それらの問題を解決するため、パケットペアを応用した映像・音声配信システムの提案をする。

4.1　パケットフローの観察

受信側は送信側から転送されたストリームデータパケットの間隔と受信時の間隔を比較することにより、ネットワークの状況把握を行なう。図 4.1にその様子を示す。

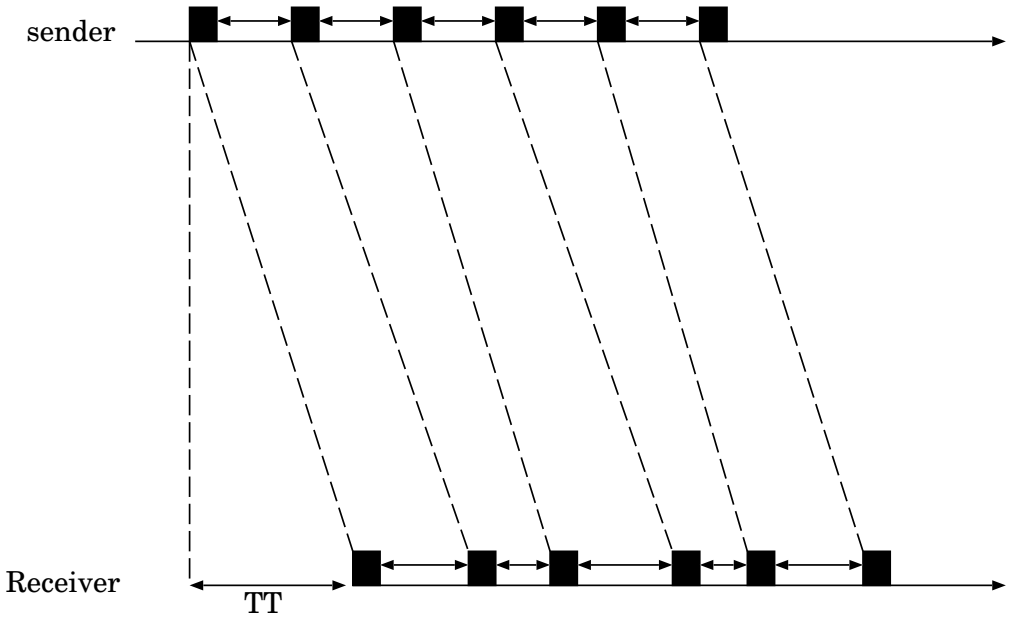

図 4.1: パケットフローの観察

ストリームデータを観察するため、送信側は計測パケットを送信する必要性がない。そのため、時間間隔を細かくしてジッタ計測を行なえる。ジッタ計測によるネットワーク状況把握には、パケットペア理論を用いる。

4.1.1 パケットペア理論の概要と検証

パケットペアの概要

パケットペア [16] はボトルネックリンクの帯域推測の手法である。中継ノードが各コネクションに対して均一な帯域を割り振る手法である。公平待ち行列を利用を前提とする。　パケットペアでは、データの送信側はペアとなる２つのパケットを連続的に転送する。公平待ち行列法では、各コネクションごとにキューが割り当てられるので、ペアとなっている２つのパケットは一緒にキューイングされる。キューイングされたパケットはボトルネックリンクを通過する時にパケットの間隔が拡大される。受信側はパケットを受信すると同時に送信側に ack を返す。確認応答パケットの大きさは、データパケットに対して十分小さいためにボトルネックリンクでの拡大は生じない。送信側は受信側から受けとった ack の RTT（Round Trip Time) の差を計測し、ボトルネックリンクの帯域を推測することができる。図 4.2にパケットペアによる帯域推測概念図を示す。

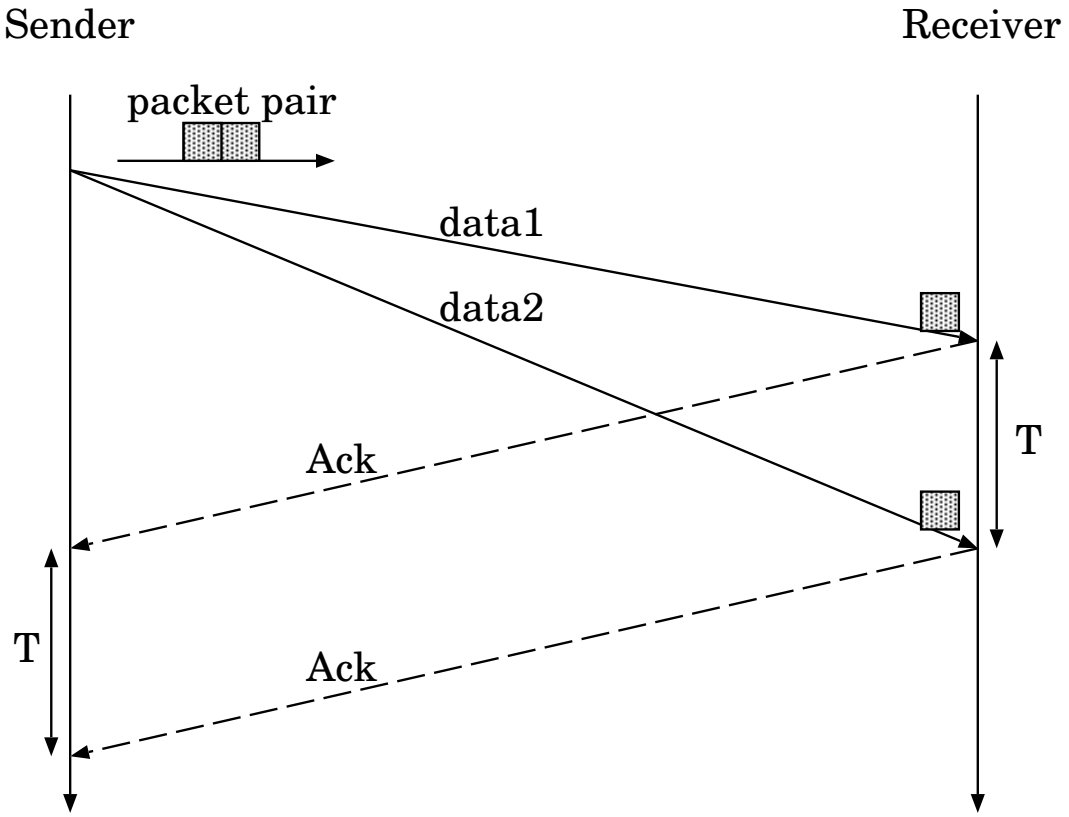

図 4.2: パケットペア方式の概念図

送信側が送ったパケットの大きさ、RTT の時間差 T から経路ネットワークにおけるボトルネックリンクの帯域を次の式で求めることができる。

$$\text{ボトルネックリンクの帯域}\,(byte/sec) = \frac{\text{データパケットサイズ}\,(byte)}{T(sec)}$$

パケットペア理論を用いる正当性の検証

パケットペアは中継ノードが公平待ち行列法を用いる場合に機能することを述べた。しかし、インターネット中の中継ノードのキューイング機構には、公平待ち行列法を採用している実装は少ない。多くの場合はFIFO(First In First Out)型のキューイング機構である。従ってペアとなる2つのパケットが連続的にキューイングされず、他のトラフィックのパケットが二つのパケット間にキューイングされてしまう可能性がある。この場合、計測結果には誤差が生じてしまう。しかし、中継ノードが公平待ち行列を採用していなくても、パケットが連続的にキューイングされる場合がある。このような考えから、パケットペアを用いて帯域推測を行なうアプリケーションとして、ボトルネックの許容帯域の推測を行なうbprobe[17]というアプリケーションがある。

上記にあげたアプリケーションはいくつかの対策を施すことで、パケットペアを用いる問題点を解決している。以下にそれらの問題点をまとめる。

- データパケットのサイズが小さい場合、ボトルネックリンクでパケットの間隔が拡大されない
- 他のトラフィックによるパケット間隔の拡大
- 計測パケットの喪失、破損
- 受信側からの応答パケット側の通信経路の輻輳

bprobeでは上記にあげた問題点に対処するため、1）転送するパケットサイズを変化させる、2）連続的に転送するパケット数を増やす、3）複数回計測を行なう、という実装を行なった。その解析結果によると、広域でボトルネックリンクの帯域が大きな場合に若干精度が低下したが、主に±20 %の範囲に80 %から90 %程度の計測値が存在することが報告されている[17]。以上のことから、中継ノードが公平待ち行列を採用してない場合でも、パケットペアによる帯域推測が有効であることが分かる。

前節で述べたフロー観察手法の場合、1) 計測回数の多さ、2) パケットサイズの大きさ、3) 受信側での計測により、いくつかの問題点は解消している。しかし、送信側は連続した2つのパケットを転送するとは限らないため、他のトラフィックによるパケット間隔の拡大は、bprobeよりも大きくなる。そのため、受信側では2つのパケット間隔の開きによってネットワークの状況を推測するのではなく、統計的に推測を行なう。送信側のパケット送出タイミングの理論値を基に、ネットワークの帯域を割出し、そ

の情報を基に送信側はパケットの送出タイミングを変化させる。送出タイミングをボトルネックリンクの許容帯域に合わせることで、パケット到達ジッタを抑える。

4.1.2 転送レート制御

本システムでは、時間粒度の細かいフロー制御を行なう。リアルタイム映像・音声配信において、伝送遅延時間は一定であると仮定すると、ある時間軸を基準にした場合、送信側から送出された単位時間辺りのデータ量は、受信側の単位時間あたりの消費量と一致する。フレーム間圧縮方式の映像データを用いた場合、動きが多いシーンのデータ量は多くなるため、受信側の再生状況を考慮した時間粒度の細かいフロー制御を行なう際に複雑な処理を必要とする。また、一つのフレームデータが他のフレームデータに影響を与えるため、何らかの原因によるパケットロスが再生品質に与える悪影響は大きい。

以上の理由から本システムでは、フレーム内圧縮方式を対象とする。フレーム圧縮の場合、固定ビットレートのため、フロー制御が用意に行なえ、また一部のデータ欠損が複数フレームに渡って連続的に影響を与えることはない。

以下に本手法が提案するフロー制御の要素を述べる。

- パケットの送出タイミングの変更
  送信側はトラフィックをシェーピングして送出する。これにより、1) バーストトラフィックによる経路ネットワーク上でのパケットロスを防ぐ、2) 受信側でパケットフローを正確に観察することが可能となる。また、受信側のパケットフロー観察の情報を基に、パケットの送出タイミングを変更する。これは、1) ネットワークの帯域幅に適した転送レートで送る、2) 瞬間的な輻輳状態によるパケットロスを防ぎ、映像品質を下げることで転送レートを下げ過ぎるのを防ぐためである。

- MTU(Maximun Transmision Unit) の変更
  MTU を変更して送信側はデータパケットを転送できるようにする。これは、ジッタは安定しているが、開きが出ているため、受信側でバッファアンダーランを予測した場合である。PassMTU を変更してデータパケットを転送することで、バッファアンダーランを未然に防ぐ。

4.2 まとめ：パケットペアを応用したフロー制御

本章では、パケットペア理論を用いた映像・音声配信を提案した。初めに現状のネットワーク推測手法、フロー制御における問題点を述べた。ネットワーク状況取得には、パケットロスだけでなく、ジッタの計測が重要である。しかし、計測パケットを用いるのは効率的ではない。フロー制御では、映像品質を落すことによる転送レート変更では、ネットワークの有効活用ができなことを述べた。そこで、ストリームデータの間

隔を受信側が比較し、時間粒度の細かい転送レート制御を送信側が行なうことで、ネットワーク資源の有効活用、受信側でのデッドラインを守る配信システムを述べた。

次章では、本手法の設計について述べる。

第5章　設計

本章では、初めに第４章で述べたパケットペア理論を用いたリアルタイム映像・音声配信を実現するための要件を述べ、設計を行なう。

5.1　設計要件

第４章で述べたパケットペア理論を用いた映像・音声配信システムの設計を行なう。その際に必要となる機能と、それを満たすための用件を以下に述べる。

- 受信側でのパケットフローの観察

  - 受信側でのパケット到達ジッタの計測は、ネットワーク状況を正確に把握できる時間精度でなければならない。MTU を 1500byte とした場合の理論値 100Mbps のネットワークを通過するのに必要な時間は、

$$Transfertime = \frac{packetsize}{Bandwidth}$$

  より、120u 秒と求められる。そのため、最低でも u 秒単位で計測を行なう必要がある。

  - パケット到達ジッタを計測する際、送信されたデータパケット転送間隔を受信側は把握しなければならない。

  - アプリケーションのバッファリング量を取得し、パケット到達ジッタの計測値を基にバッファアンダーランを防ぐ。

- 送信側のフロー制御

  - 受信側で計測されたパケット到達ジッタを基にパケットの送出タイミングを変更する。パケット到達ジッタの値は、送信側で用いる時間単位に合わせなければならない。

  - データパケットサイズを変更することにより、ボトルネットリングでの伝送遅延の間隔を狭め、映像データを受信側のデッドラインに間に合わせる。

5.2 設計概要

前節で述べた機能を満たすための、本研究の設計概要を図 5.1に示す。

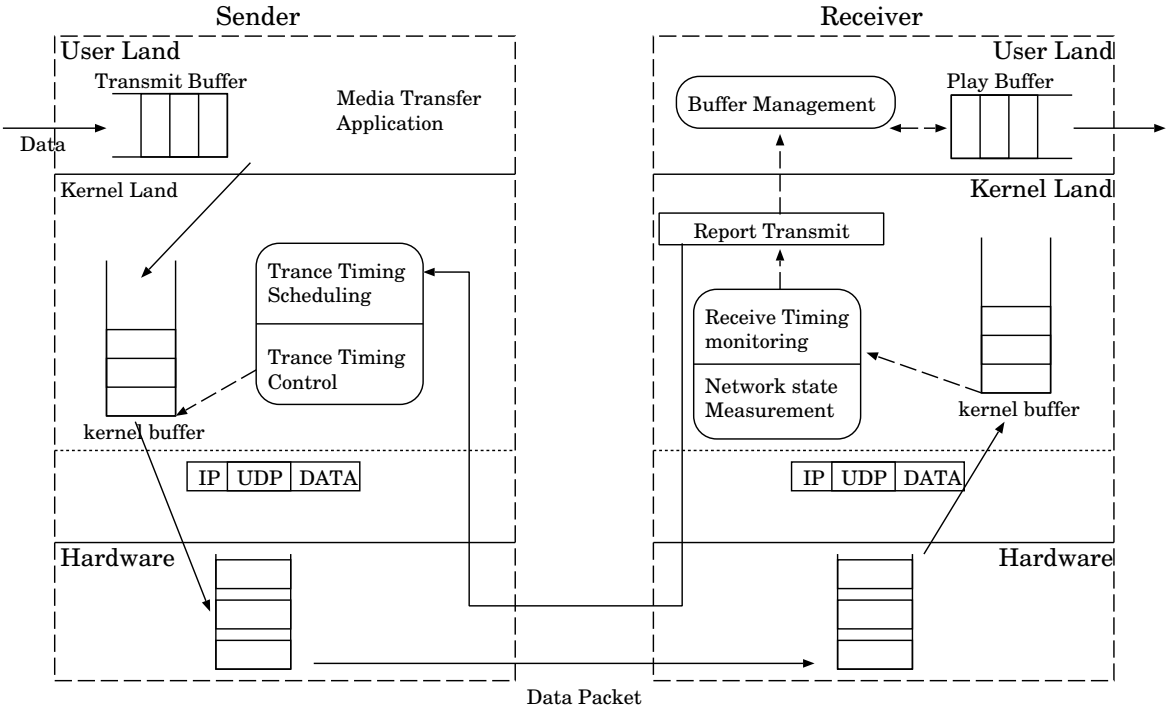

図 5.1: 設計概要図

●映像・音声の配信の流れは以下の手順によって行なわれる。

1.
送信側: CPU 能力の差異を考慮したパケット送出タイミング尺度の決定受信側: ネットワーク状況把握、再生バッファ監視に用いる尺度を RTT によって決定。2.
送信側: 一定の送出タイミングでトラフィックをシェーピングしてデータ転送
3.
受信側： パケット到達ジッタ計測
4.

受信側: パケット到達ジッタ計測によるネットワーク状況推測、再生バッファの管理
5.
受信側： 4 の情報を基に適切な送信側の送出タイミング決定
6.
受信側： 送信側に転送レート変更要求を通知
7.
送信側： 受信側の要求に基づいた送出タイミング変更

5.3 Kernel 内へのモジュール組み込み

本設計においては、送信側の転送スケジューリングモジュールと出力制御モジュール、受信側の受信タイミング把握とネットワーク状況把握部、レポート送出部はカーネル内で実行する。

5.3.1 送信側の出力制御モジュール

アプリケーション側からパケットの送出タイミングを変更する場合、ユーザランドから Socket インターフェースに映像データを渡すタイミングを調整することで可能である。しかし、ユーザランドからカーネルランドにデータが渡される際、割り込み処理が発生することがあるため、予測できない通信遅延が発生する。そのため、実際にネットワークインターフェースから出力されるタイミングは、アプリケーション側が予定したものとは大きく異なる可能性がある。その結果、受信側のパケット到達ジッタ計測に大きな誤差が生じ、ネットワークの状況を正確に把握できない。　以上の理由から、本設計では UDP 層に出力制御モジュールを配置する。UDP 層から IP 層にデータグラムが渡るタイミングを制御することで、誤差を最小限にとどめる。

5.3.2 スケジューリング決定モジュール

スケジューリング決定部には、後述する送受信側の CPU 性能の差異を吸収し、受信側の通知に対して適切な送出タイミングを出力制御部に伝える。また、映像転送を始めに行なう際に、アプリケーションに対して最適な送出タイミング値を保持する。

5.3.3 受信タイミング監視モジュール

上記に述べた理由と同じで、UDP 層に受信タイミング把握モジュールを配置する。IP 層から渡ってきたパケットを UDP 層が受け取るタイミングを毎回計測することで、パケット到達ジッタを計測する。ネットワーク状況、もしくはバッファリング状況に異変を検知した場合、できるだけ送信側は迅速にトリガーアクションを起こす必要性

があることを第３章で述べた。そのため、ネットワーク状況把握部、レポート送出部共に同等の理由からカーネル内に組み込む。

5.3.4 時間処理

送信側のパケット出力タイミング調整、受信側のパケット到達ジッタ把握をする際、何らかのタイマー機構が必要である。本機構では、時間粒度を細かく調整する必要があるため、タイマーの時間精度は重要である。しかし、カーネル内で用いるタイマの粒度は一般的に粗い。そのため、本機構では、時間処理には CPU カウンタを用いる。CPU カウンタとは、CPU 稼働中１クロックサイクル毎に１ずつインクリメントされる経過クロックサイクル数である。1GHz のクロック周波数を持った CPU の１クロックサイクルは１ n 秒であり、カーネル内のタイマ機構よりも時間粒度は細かい。CPU カウンタを基に送受信側共に時間処理を行なう。

送信側と受信側の CPU 能力には違いが生じるため、どちらも独自の CPU カウンタを用いると尺度が異なってしまう。そこで、統一した時間処理を行なうため、予めある経過数を CPU カウンタの増加数で割ることによって、CPU 能力の違いの比率を求め、送信側がある一定時間に増加する CPU カウンタ数を受信側の CPU カウンタ増加数に合わせる。

5.3.5 ネットワーク状況把握モジュール

送信側のパケット送出間隔は、始めはアプリケーションに適したある一定値である。そのため、受信側はその間隔を基準値とし、受信側はパケット到達ジッタを把握する。ネットワーク状況把握部は、ネットワークの帯域が安定、減少しているかの２点を判断する。

パケット到達ジッタの計測には、安定状態からの触れ幅の平均値、一回の計測による間隔値を共に用いる。両値は限度値が設定してあり、限度値を超えた場合、ネットワークの帯域減少を把握し、レポート送出部に通知を行なう。

5.3.6 バッファリング監視モジュール

バッファリング監視部は、アプリケーションの再生バッファを参照するため、ユーザランドに存在する。バッファの状態は、常にレポート送出部が管理できる。バッファのキューの長さがある一定値を超える、もしくは下回った場合、レポート送出モジュールは送信側に適切な要求を出す。

5.4 レポート送出部

レポート送出部は、ネットワーク状況把握部とバッファリング把握部の状態通知によって、両者の状態を把握し、送信側にパケット送出タイミング変更を要求する。

転送レート変更要求のアルゴリズム

転送レート変更要求は、ネットワーク状況把握部、バッファリング監視部から通知を受信した場合に行なう。図 5.2以下に処理の流れを示す。

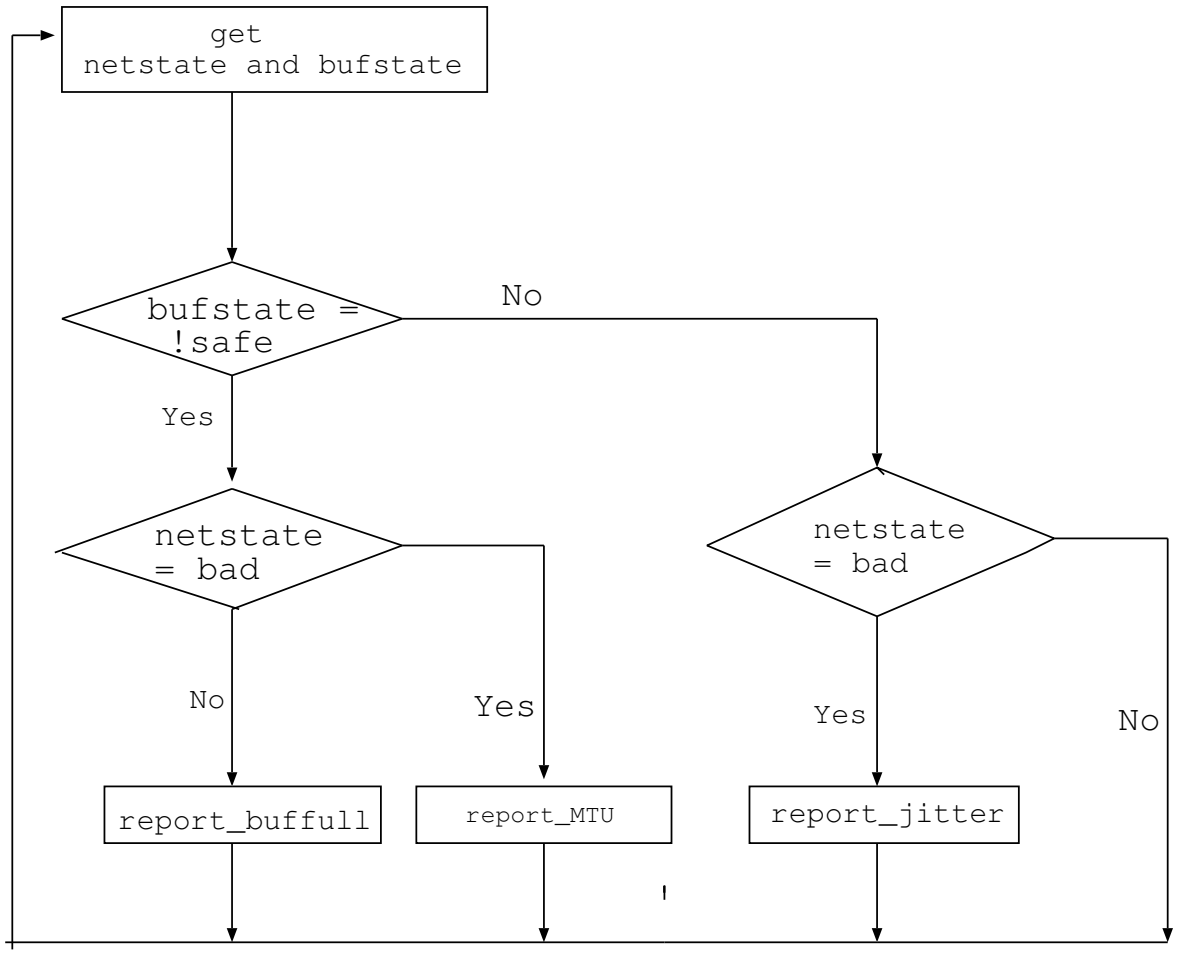

図 5.2: レポート送出モジュールの処理

5.4.1 ネットワーク状況把握部・バッファリング監視部からの通知

上述したジッタ計測により、ネットワークの帯域資源が減少した情報を受信した場合、受信情報に基づいたパケット転送送出タイミングを送信側に要求する。その際、

バッファリング状況が安定状態になっているか確認を行なう。バッファが不足しており、パケット到達ジッタの間隔が広い場合は MTU 変更を要求する。これは、パケットのサイズを小さくすることで、パケットのキューイング処理を軽減させ、受信側に早くパケットを到達させるためである。バッファの状態が安定しており、パケット到達ジッタの間隔が広い場合は、そのジッタ情報を基にパケットの送出間隔を開ける要求を行なう。

5.4.2 RTT の考慮

エンド間における RTT が小さければ小さいほど、送信側は受信側の要求に基づいたトリガーアクションを早く行なえる。逆に RTT が大きければ、送信側の対応が遅くなってしまう。そのため、通信を行なう前に事前に RTT の計測を行なう。計測を行なった RTT に基づいて、ネットワーク状況把握部、バッファリング監視部で用いるパラメータを変更し、早めに異変を検知することで対応を行なう。

5.5 本設計の送信例

図 5.3に本システムの送信例を示す。

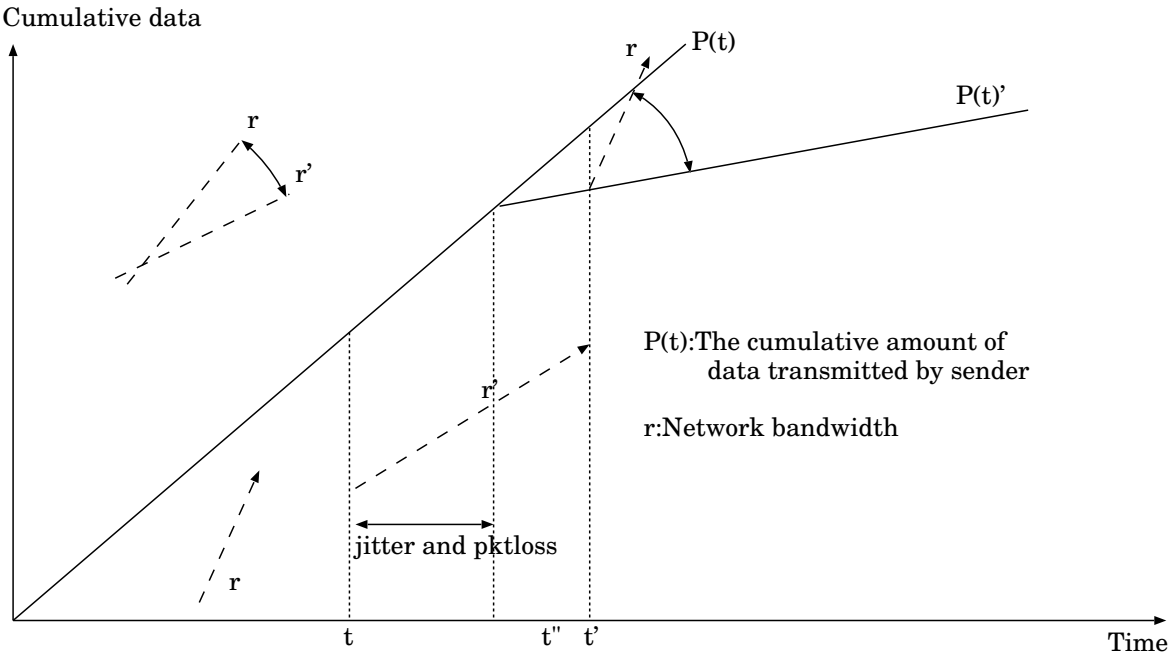

図 5.3: 本システムの送信例

この例では、帯域幅が時間 t から t' にかけて r から r' に変化している。帯域幅 r' は

リアルタイム映像配信アプリケーションが必要とする最低限の dP(t)/dt よりも足りないため、受信側のパケット到達ジッタに影響が出る。送信側は受信側と協調し、時間 t' にパケットの送出間隔を開けて送信する。時間 (t“ - t‘) では使用する帯域幅が r‘以下になっており、受信側の到達パケットはほぼ一定で到達する。

送信側がパケットの送出間隔を開けた後、受信側のバッファは枯渇していく。時間 A では、受信側は送信側に再生バッファ状態を通知し、送信側は時間 B でパケットの送出間隔を狭め、受信側の再生バッファを満たしていく。送受信側が協調し、送信側が時間粒度の細かいフロー制御を行なうことで受信側は適切なタイミングでパケットを受信できる。

5.6 まとめ

本章では第 4 章で提案したパケットペア理論を用いた映像・音声配信の設計概要・用件について述べた。次章では、本設計に基づいて DVTS を用いた場合の実装について述べる。

第6章　実装

本章では、第５章にて行なった設計を基に、DVTS に適用した場合の実装概要について述べる。6.1 節では、実装概要と実装を行なった環境について述べる。6.2 節以降では、本機構の各部分における実装について述べる。

6.1　実装概要

インターネット上で DV ストリームの送受信を行なう DVTS を用いて、第４章で行なった設計を実装した。

subsectionDVTS　DVTS は DV 機器から IEEE1394[8] インターフェースを用いて DV データのフォーマットを読みだし、IP パケット化を行ない、転送を行なうリアルタイム伝送を目的としたシステムである。以下に DVTS の概要について述べる。

- IEEE1394 からの映像・音声データの入出力に Isochronous 転送を用いる。
- 音声は IEEE1394 からの入力を非圧縮で転送する
- IPv4/IPv6 に対応する
- RTP を利用した通信を行なう
- Unix 用ソフトウェアとして、Open Source で提供されている。

受信側のパケット到達ジッタの計測には、CPU カウンタを用いるため、使用する機器の CPU は、Pentium 以降の x86 シリーズでなければならない。

実装に用いたソフトウェア環境を表 6.1に示す。

表 6.1: 実装ソフトウェア環境

OS	FreeBSD 5.2 RELEASE
プログラミング言語	C 言語
コンパイラ	gcc 3.2

本機構の実装を行なった機器環境を表 6.2に示す。

表 6.2: 機器環境

CPU	Pentimun III 1GHz
メモリ	512MB
ハードディスク	80GB
NIC	100Base-TX

6.2 送信部

DVTS は DIF ブロックと呼ばれる 80byte のデータを基本単位として構成される。映像方式が NTSC の場合、フレームレート 29.97fps であり、その 1 枚のフレームは 1500 個の DIF ブロックで構成される。

MTU を Ethernet の 1500byte とした場合、一つの IP パケットに内包することができる DIF ブロックの数は最大 18 となる。この条件で最適な送信タイミングを計算すると、$(80 \times 18)/(29.97 \times 120000)$=400u 秒となる。出力制御モジュールでは、この値を基本値として扱う。

6.2.1 スケジューリング決定モジュール

スケジューリング決定部は、受信側のパケット到達ジッタ情報、バッファリング状況を基にパケット転送タイミングを決定させる。独自のシステムコール *dv_send()* は、UDP データグラムを生成する *udp_dv_output()* を通じて IP パケットを生成する。スケジューリング決定部は, 受信側の通知を受け取った場合、udp_dv_output() で呼ばれる。転送タイミングを制御する際のパラメータには、受信側が通知するパケット到達ジッタ間隔値、バッファリング量がある。図 6.2にスケジューリングモジュールの実装を示す。

関数 *timing()* は、受信側が使用する CPU カウンタの性能の差異を吸収するための関数である。システムコール textitdv_send() は,u 秒単位でインクリメントされる CPU カウンタの値と、受信側から通信の始めに通知される同単位の CPU カウンタを比較する。比較値を基に受信側から通知されるジッタ情報を自身の CPU カウンタ性能に合わせる。

6.2.2 出力制御モジュール

関数 *udp_dv_output()* で呼ばれる出力制御モジュールは、通常時、400u 秒単位でパケットを送出するように制御を行なう。関数 *ip_output()* に渡すタイミングを CPU カウンタで毎回計測を行なう。前回のカウンタ数よりもインクリメント数が少なければ、タイミングが早いと判断し、インターバルを挿入する。インターバルは受信側のフィードバック情報によって可変である。インターバルの調整には、CPU カウンタ数で指定された時間だけ処理を中止させる関数 textittimebal_cpu() を用いる。

```
long long
schdule(){
  (omit)
/* scheduling transtiming */
if(bufstate){
  go out; /* normal send */
}
else{
  timing = refresh(jitter) /* regulation of
}                              CPU counter differencu */                    (c
  retrun timing;
}
```

図 6.1: スケジューリング決定モジュール

6.3 受信部

受信部は、UDP 層にあるモジュールが主な役割を果たす。IP 層で識別された DVstream パケットは独自の関数 *udp_dv_input()* に渡される。カーネル内のモジュールはこの関数内に実装されている。

udp_dv_input() において、パケットの受信タイミング計測によるネットワーク状況把握が終了した後、DV データは独自のシステムコール dv_from() によって読み込まれる。dv_from システムコールは、アプリケーション層のバッファリング状況を udp_dv_input 関数に通知できるように実装した。以下、各モジュールにおける関数の詳細を述べる。

6.3.1 受信タイミング把握部

受信タイミングの計測は、IP 層から UDP 層に dv データが渡された瞬間に計測を行なう。図 6.2に CPU カウンタでパケット到達の差分を得る実装を示す。

6.3.2 ネットワーク把握部

udp_dv_input() は、得られたジッタ情報を基にネットワークの状況を把握する。把握した情報はカーネル内のモジュールが利用する kern_dv_from 構造体に格納する。図 6.3に示す。

```
/* include file */
long long rdtsc_count(void);
__asm__("rdtsc_count: ");
__asm__(" rdtsc ");
__asm__(" ret ");

static long long
get_jitter()
{
  long long now_count;
  long long rdtsc_jitter;
  static long long pre_count = 0;

  /* input now_rdtsc*/
  now_count = rdtsc_count();

 if(pre_count){
  rdtsc_jitter = now_count - pre_count;
  pre_count = now_count ;
 }else{
  pre_count = now_count;
  return 0;
 }
   return rdtsc_jitter;
 };
```

図 6.2: パケット受信タイミングの取得

6.3.3 レポート送出部

レポート送出部の実装を図 6.4に示す。

6.4 本章のまとめ

本章では、５章の設計に基づき実装を行なった概要、ならびに以下に示す各部の動作の詳細について述べた。第７章では、本機構の動作について検証し、本手法を用いることの有用性について評価を行なう。

```
struct kern_dv_from{
long long net_state;      /* network state */
int usr_bufstate;         /* user Buffuring State */
struct sockaddr *send;    /* args of dv_from */
struct socket  *so        /* socket for reporting */
};
```

図 6.3: カーネルモジュールが利用する構造体

```
void
report(long long rdtsc)
{
  (omit)
  /* report in or out */
if(kern_dvi_from->net_state && (kern_dv_from->bufstate = empty){
  report_send(0,usr_bufstate); /* request for full buffer  */
}
else if(kern_dvi_from->net_state && (ken_dv_from->bufstate = SAFE)}
  report_send(0,usr_bufstate);  /* request for changing mtu */
else if(!kern_dvi_from->net_state){
  report_send(jitter,usr_bufstate);/* request for
}else if}                                    decreasing transrate */
 go out;
}
 (omit)
}
```

図 6.4: レポート送出部

第7章　評価

本章において本実装の評価に関して述べる.

7.1　評価概要

本評価は、本研究の提案するフロー制御手法の有効性の検証を目的とする。訂正評価、ならびに定量評価は以下の項目について行なう。

- 定性評価：実装により実現した機能
- 定量評価：フロー制御を行なった際の送受信側の挙動

7.2　本システムの実現した機能

本機構は、第４章で述べた機能を DVTS を用いて満たし、End to End モデルを前提としたパケットのフロータイプ制御機能を実現した。

- CPU カウンタを利用した時間精度の高いパケット到達ジッタ計測
- 通信ノード間でのパケットフロータイプの情報交換
- 受信側でのパケット到達ジッタに基づくパケット送出間隔変更機能

7.3　定量評価

本機構を利用した場合の DVTS の挙動を確認し、パケットのフロータイプ制御が実現できたかを評価する。評価する際に当たっては、以下に挙げる実験を行ない、本機構を利用した場合の DVTS の有効性を考察する。

- 送信ホスト OS に負荷をかけた場合の挙動

  ホスト OS に負荷を欠けた場合に、パケットの送出フロータイプがどのようになっているかを評価する。評価の指標は、パケット送出間隔の最小値、最大値、平均および標準偏差を用い、考察を行なう。

- 帯域制御

  瞬間的にバーストトラフィックを発生させ、帯域制御を行なった場合の受信側のパケット到達ジッタの挙動を評価する。また連続的に帯域幅を上下させた場合の挙動を評価する。。評価の指標は、パケット到達ジッタの最小値、最大値、平均および標準偏差を用い、考察を行なう。

- 遅延時間を変化させた場合の挙動

  パケット伝送にかかる時間を増加させた場合に、帯域制御を行なった際の受信側のパケット到達ジッタの挙動を評価する。

第8章　結論

8.1　まとめ

IP ネットワークを通信路とし、リアルタイム映像配信システムでは、ネットワークの帯域減少によって起きるパケットロス、伝送ジッタが受信側の再生品質に悪影響を与える。特に組み込み機器や携帯端末においては、計算機資源が制限されているため、十分なバッファリングを行なうことが難しい。そのため、受信側でのパケット到達ジッタが大きく再生品質に影響する。再生品質を維持するためには、ネットワークの状況を迅速、かつ正確に把握し、受信側の状況を考慮した効率的な配信をしなければならない。

End to End モデルでネットワークの状況を把握するためには、パケットの伝送遅延を考慮する必要がある。しかし、既存の技術では主にパケットロスを指標にネットワークの状況把握を行ない、転送レートをさげることで使用帯域を削減し、再生品質を維持する。この問題点には、一時的なパケットロスによる不必要な転送レート制御、パケット到達ジッタによる再生品質の劣化を考慮していない問題が挙げられる。

本研究では、ネットワークの帯域変化が受信側でのパケット到達ジッタに現れることに着目した。受信側では、送信側のパケット送信タイミングと受信タイミングを比較することで、パケットペア理論を用いたネットワークの状況把握を行なう。送受信側は協調を行ない、送信側はネットワーク状況、受信側のバッファリング状況を考慮し、パケットのフロータイプを変化させる。本手法を DVTS に実装した。

評価に当たっては、本手法を用いた場合に、受信側でのパケット到達ジッタ、パケットロス値を既存のものと比較を行なった。

本手法を用いることで、ネットワーク帯域資源を有効活用し、パケット到達ジッタを減少させた効率的なリアルタイム映像配信が可能となった。本研究により、リアルタイム映像配信の信頼性の向上が可能となった。

謝辞

本研究を進めるにあたり，御指導を頂きました，慶應義塾大学環境情報学部教授の村井純博士，徳田英幸博士，同学部助教授の楠本博之博士，中村修博士，同大学環境情報学部専任講師の南政樹氏，重近範行氏に感謝致します．絶えず暖かい目で指導していただいた独立行政法人通信総合研究所の杉浦一徳氏，に感謝します。多大な御迷惑をおかけし、いつも助言を与えてくれた久松氏に感謝します。同研究軍団の修士課程、三島氏、堀場氏に多大な感謝をします。最後に長き日々をともにした、stream 軍団に感謝します。

参考文献

[1] SOI(School of Internet. *http://standards.ieee.org/reading/ieee/std/lanman/restricted/802.3u-1995.pdf.*

[2] DVTS Consortium. *http://www.dvts.jp.*

[3] J. Postel. RFC768 User Datagram Protocol. *http://www.ietf.org/rfc/rfc0768.txt*, pages 1–3, August 1980.

[4] Windows Media 10 series. *http://www.microsotf.com/japan/windowsmedia.*

[5] Comet Project. *http://www.comet-can.jp.*

[6] J. Postel. RFC793 Transmission Control Protocol. *http://www.ietf.org/rfc/rfc0793.txt*, pages 1–85, September 1981.

[7] Audio-Video Transport Working Group. RFC1889 RTP: A Transport Protocol for Real-Time Applications. *http://www.ietf.org/rfc/rfc1889.txt*, pages 1–75, January 1996.

[8] Institute of Electrical and Electronics Engineeers, Inc. IEEE Std 1394b-2002, Amendment to IEEE Std 1394-1995. *http://standards.ieee.org/reading/ieee/std/busarch/1394b-2002.pdf*, Dec 2002.

[9] Sure Stream *http://www.service.jp.real.com.*

[10] David Taubman,Member, IEEE High Performance Scalable Image Compression with EBCOT July,2000

[11] Akimich ogawa DVTS(Digital video Transport System). *http://www.sfc.wide.ad.jp/DVTS/.*

[12] K.Ramakrishnan and S. Floyd A Proposal to add Explicit Congestion Notification(ECN) to IP. january 1999. RFC2481.

[13] Jean-Chrysostome Bolot, Hugues Crepin, Andres Vega Garcia Analysis of Audio Packet Loss in the Internet. 1995.

[14] Kazuhiro Mishima 伝送特性に応じた適応型映像・音声配信機構の設計と実装 jan,2003.

[15] S.Floyd and K.Fall. Promoting the use of end-to-end congestion control in the internet IEEE/ACM Transactions on Networking, August 1998.x

[16] Keshav,S. A Control-Theoretic Approach to Flow Control. Proceedings of ACM SIGCOMM'91. September 1991.

[17] Carter, R., Crovella, M. Measuring Bottleneck Link Speed in Packet-Switched Net works. Technical Report BU-CS-96-006. Computer Science Department,Boston University. March 1996.