

# P2P ストリーミングにおけるジッタ性能評価

2007MI024 福島 悠太 2007MI111 近藤 大樹

指導教員 河野 浩之

## 1 はじめに

P2P 技術の発達につれ、P2P ストリーミングの利用者が増加している。しかし従来のクライアント・サーバ方式と同様に、ジッタによる動画再生への悪影響の問題を持つ。このジッタを発生しにくくすることで、ストリーミングサービスの品質向上を図ることができる。本研究では、ストリーミングサービスの品質向上をどのように実現すればいいのかを提案する。実験では P2P ネットワークを構築し、パケットキャプチャソフトでパケットの計測を行い、ジッタ発生率の解析を行う。ジッタの発生を極限まで減らしつつ、動画再生品質を出来るだけ向上させる方法を模索する。

## 2 P2P ストリーミングのクオリティ向上への提案

### 2.1 クライアント・サーバ方式との違い

Masahiro Sasabe らの文献 [1] では、高品質かつ低遅延なビデオストリーミングサービスを提供する為のプロキシ貯蔵メカニズムを提案している。このメカニズムはクライアント・サーバ方式のストリーミングに対する提供であり、本研究で取り扱う P2P ストリーミングとはネットワークの形式が異なる。文献 [1] に限らず、クライアント・サーバ方式のストリーミングのクオリティを向上させるには、その大黒柱である配信サーバの安定性や処理速度に焦点を当てる必要がある。それに対して P2P ストリーミングのクオリティを向上させるには、配信者だけでなく、中継するピアの環境も重要な要素となる。本研究では P2P ストリーミングについての測定を行うので、サーバ周辺だけでなく、各ピア間の送受信に焦点を当てている。

### 2.2 Xiaosong Lou らによる類似研究

Xiaosong Lou らの文献 [2] では、P2P を利用した IPTV<sup>\*1</sup> の映像再生中のジッタによる動画の不具合を解決する提案を行っている。シミュレーションを用いて P2P を用いた IPTV サービスを構成し、ピアのダウンロード待ち時間をどうすれば分配できるかの模索をする事で、P2P ストリーミングのジッタを軽減させようとしていた。文献 [2] より、 $q(t)$  をタイムスロット  $t$  内でジッタが発生したピアの割合、 $L(t)$  をクライアントの

前方 (再生前) バッファのうち使用可能なセグメントの数、 $c(t)$  を同タイムスロット中にダウンロードしたデータチャンク量、そして  $C$  は 1 タイムスロットあたりの総チャンク量とすると、式 (1) が成立する。この式は文献 [2] 内で証明が行われている。

$$q(t+1) \leq Pr[c(t) < C | L(t) = 1] \times Pr[L(t) = 1] (1)$$

文献 [2] では、5 つのノードと 2000 のクライアントを設定したシミュレーションによってピアのダウンロード待ち時間をうまく分散させる手段を模索していた。

本研究では、シミュレーションではなく実際にハードウェアを用いて P2P 環境を構成する。しかし文献 [2] のように 2000 以上のピアを用意する事は実環境ではコスト面から見て不可能なので、各ピアの中継にリピータハブを使うことで、ネットワーク内のパケットを増やしてジッタの変化を推測する。

文献 [2] では式 (1) によって一定時間毎のジッタ発生率を主に計測している。本研究では、TCP が定期的に算出する RTT 値と、連続するパケットの到着間隔の変動からジッタの変化を推測する。

表 1 研究内容の比較

	文献 [2]	本研究
ピア数	2005	12
環境	仮想環境	実環境
過負荷方法	膨大なピア数	リピータハブ
ジッタ算出	式 1	RTT&到着間隔

## 3 ジッタ測定に必要なツールなどの提案

### 3.1 ジッタの計算方法

TCP プロトコルには、RFC 3550 RTP のようなパケットの到着時間の差を直接参照できる要素は存在しない為、別の要素でジッタの変化を測定する必要がある。TCP プロトコルでは、帯域測定の為に往復遅延時間 Round Trip Time(RTT) を定期的に計測する。双方向通信の TCP では、パケットを送信してから、そのパケットの ACK を受け取るまでの時間差を計ることができる。

### 3.2 ソフトウェア構成と使用するツール

ジッタ評価を行う為に P2P ネットワークを構成し、そのネットワークに意図的に負荷をかけつつストリーミ

\*1 IPTV(Internet Protocol TeleVision)、インターネットを利用してテレビ放送を配信するサービス

ングを行い、ジッタの変化がどのような環境に依存するかを測定する。本研究では、WireShark, PeerCast を用いて、ストリーミングと。小節 3.2.1 以降は、これらツールの特性を記述する。

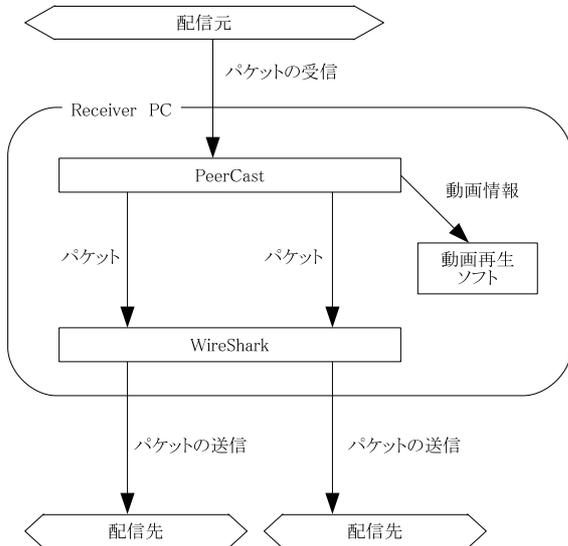


図1 ソフトウェア構成図

これらのツールは、各 Receiver 内で動画データの流れが図 3.2 のようになるよう構成する。図中の PeerCast は他 PC とのパケットのやりとりを中継しつつ、動画再生ソフトへデータを動画再生できる形式にして渡す。WireShark は配信先へ送るパケットの解析を行う。ここで得られるデータから RTT 値とジッタの変動について考察する。

### 3.2.1 パケットキャプチャ

RTT 値を得る為のキャプチャには、WireShark<sup>\*2</sup> を使用する。WireShark はパケットキャプチャソフト Ethernet の後続のフリーウェアである。通信しているプロトコルを解析する機能を持ち、パケットからジッタや遅延がどの程度発生しているかを検出することが出来る。本研究では、WireShark を一部のピアに導入してジッタの測定を行う。WireShark は TCP 通信での RTT 値およびパケットの到着時間間隔の変化をグラフ化して出力することができる。本研究ではこのグラフ化の機能を用いて、RTT 値とパケット到着時間の変化を考察する。

### 3.2.2 ストリーミングソフトウェア

視聴が個人で自由に可能な P2P ストリーミングサービスの候補を表 2 に記す。この候補の内、

\*2 WireShark, <http://www.wireshark.org/>

表 2 既存の P2P ストリーミングサービスの性能表

サービス名	個人での配信	ビットレート
LiveStation	不可	—
KeyHoleTV	可	100/200/500(Kbps)
PeerCast	可	任意

LiveStation<sup>\*3</sup> は個人で配信が不可能であり、ストリーミングを自由な形式でキャプチャする事ができず、測定には向かない。KeyHoleTV<sup>\*4</sup> は 500Kbps を超える高ビットレートでの配信ができないので、今回の実験で想定している高ビットレートの動画とジッタ変化量の関係の測定が困難になってしまう。よって、PeerCast<sup>\*5</sup> や Livetube<sup>\*6</sup> のような、個人での配信が可能で、動画の形式を自由に指定できるストリーミングサービスがこの測定に適していると言える。

### 3.3 ローカルネットワーク内で負荷を与える手段

PeerCast は節 3.2.2 でも述べたように、ピア同士の最初の接続は YellowPages を介して行うが、一度接続してしまえばそれ以降はピア同士で直接パケットの送受信を行う。ストリーミングに参加するピアが全て同じネットワーク内にいれば、最初の接続以外の全ての通信をローカルネットワーク内で行う事になる。近距離の通信なので帯域への負荷が少なく、長距離通信を前提とした計測を行うには不向きである。同一ネットワーク内にあるコンピュータで長距離通信時のような負荷を再現したい場合、ソフトウェアで意図的に負荷を与えるか、帯域の狭いハードウェアを中継させれば、長距離通信でなくても、帯域に負荷を与えることができる。本研究では、測定時にリピータハブを用いて各 PC を接続する。そうすることでパケットがスムーズに到着せず、ネットワークに負荷をかけることができる。

### 3.4 配信に使用する動画

本来のストリーミングは、低帯域回線のピアが中継に参加しても当たり障りのないようにと、配信する動画のデータ量はできるだけ小さくするべきである。データ量を小さくし出来るだけ回線に負荷を与えないようにする事で、ジッタによるストリーミングへの悪影響も抑えることが出来る。しかしこの研究では動画がストリーミングへ与える影響を測定するので、できるだけ回線に負荷を与える動画を配信する。出来るだけ動きの激しい動画を高ビットレートで配信すれば、動画の持つデータ量が増加し、動画がストリーミングに与える影響も大きく

\*3 LiveStation, <http://www.livestation.com/>

\*4 KeyHoleTV, <http://www.v2p.jp/video/>

\*5 PeerCast, <http://www.peercast.org/download.php>

\*6 Livetube, <http://livetube.cc/>

なる。

## 4 ジッタ測定システムの実装

### 4.1 測定環境

#### 4.1.1 ハードウェアの構成

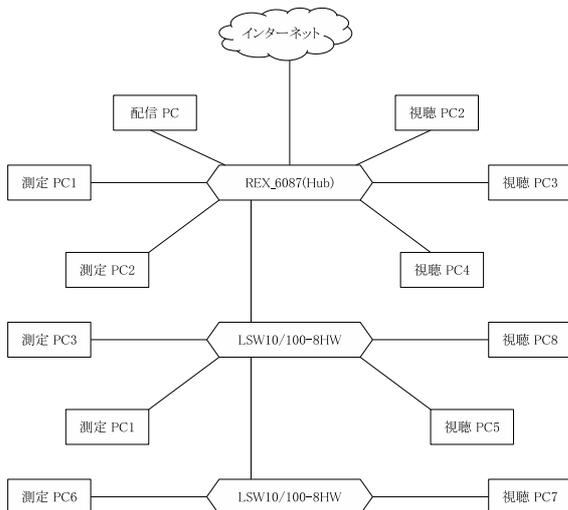


図2 ハードウェア構成図

P2P ストリーミングを実現するためのハードウェア構成を図 4.1.1 に示す。動画配信を行う SenderPC は、リポート Hub である RATOX REX-0687 を経由して Receiver にストリーミングを行う。PC1 は REX-0687 の直下に接続し、その他の PC は 100BASE-TXHub の直下に接続する。こうすることで、100BASE-TXHub に接続されている PC がネットワークへ接続する際に、REX-0687 の帯域制限に引っ掛かりやすくなり、パケット転送がスムーズに行われなくなるようにする。ReceiverPC は RATOX REX-0687 を経由して Sender からパケットを受け取る。図 4.1.1 のようにストリーミングに参加する PC は 100BASE-TX 対応の Buffalo LSW10/100-8HW を経由して接続する。この Hub はジャンプスイッチでポート毎に転送速度を指定可能で、さらに全二重転送か半二重転送のどちらで接続するか指定可能な Hub である。この Hub を接続し、全てのポートを 10Mbps、半二重転送モードで接続した。

#### 4.1.2 測定に使用する動画とストリーミングの条件

今回の実験で用意する動画は、ゲームのリプレイ画面をソフトウェア DirectShowFilter の一つ、SCHF を用いてキャプチャし、WindowsMediaEncoder9 でストリーミングを行う。フレームレートは 29.97fps、解像度は VGA サイズ固定とする。

### 4.2 実験内容

この節では、前節 4.1 で構築した環境下で小節 4.2.1 から小節 4.2.2 までで記す 3 種類の実験を行う (それぞれを実験 A, B, C とする)。それぞれの実験の大まかな特徴は表 3 に記述する。

表 3 ジッタの測定を行う 3 種類の実験

	検証する要素	ビットレート	ピア数
実験 A	ビットレート	変動	12
実験 B	ピアの数	—	変動
実験 C	通信帯域	—	12

キャプチャしたパケットには、ストリーミングには無関係のパケットも含まれている可能性があるため、グラフ化の際には WireShark 内で TCP ポート 7144 のみを抽出するフィルタリングを行う。PeerCast は全ての通信を TCP の 7144 ポートを通して行うので、このポートを通るパケットに限定すれば、ほぼ全てのキャプチャしたパケットがストリーミングに関するパケットとなる。フィルタリングのコマンドは `((tcp.port == 7144) and (ip.src == "src.address") and (ip.dst == "dist address"))` と記述する。

#### 4.2.1 実験 A ビットレートとジッタの関係

ストリーミングのビットレートを変化させた時に、ジッタにどのような影響があるのかを解析する。動画のビットレートは、500Kbps, 700Kbps, 1Mbps の三種類を用意し、ストリーミング中の参加や退出は無いものとする。この状態でキャプチャを行う。

#### 4.2.2 実験 B ピア数とジッタの関係

この節では動画のビットレートは 700Kbps で統一する。実験 A から得られた最大ビットレートの動画を流し、開始時のピア数は実験 1 よりも少ない状態から開始する。ストリーミング開始後から参加させるピア数を徐々に増やしていく。この節の実験は 3 通り行う。

##### B-1 参加ピアの増加

ノードとピア 7 台の計 8 台を P2P に参加させた状態からストリーミングを開始し、パケットキャプチャを延べ 2 分行う。測定開始より経過 30 秒から、10 秒おきにピアを 1 台ずつストリーミングに参加させ、ネットワークへの影響をキャプチャ結果から推測する。

##### B-2 先端ノードから参加ピアの減少

参加しているピア数をノードの先端にあるピアから順に少しずつ減少させ、その影響を検証する。実験 B-1 で途中から参加させたピアを、実験 B-1 とは逆の順番でストリーミングから退席させていく。測定開始より経過 30 秒から、10 秒おきにピアを 1 台ずつストリーミングに参加させ、ネットワークへの影響をキャプチャ結果から

推測する。

## 5 実験結果と考察

説 4.2 で記した内容でストリーミングのキャプチャを行い、それぞれの RTT とジッタ到着遅延時間のグラフを得ることができた。

### 5.1 動画のビットレートごとの RTT 値の評価

図 3 と図 4 は、実験 A でビットレートが 500Kbps の時に出力したグラフである。他の出力グラフと比較して、ビットレートの上昇につれて RTT の最大値も上昇し、RTT パケットの量が増えている事が分かる。ネットワークの余剰帯域が狭まり、パケットが潤滑に送受信されていないことが原因であると言える。また I/O Graph でスループットをグラフ化した場合、今回構成したネットワークの有効帯域は 1Mbps 以下であることが判明したので、実験 B で行うビットレートを 700kbps に固定する事にした。このネットワーク環境のまま配信ビットレートを 1.5Mbps に引き上げた場合、ストリーミングが強制終了する場面が何度があった。この要因として、ネットワークの帯域不足が深刻になり正常なストリーミングが受信できなかったと言える。

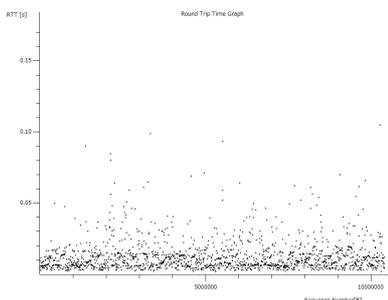


図 3 実験 A の直下ピアへの RTT グラフ ビットレートが 500Kbps の場合

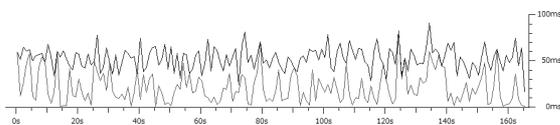


図 4 実験 A の到着時間の I/OGraph ビットレートが 500Kbps の場合

RTT 値が上昇するということは、パケット送信側から受信側にパケットが到達して、そのパケットが正常に受信できたことを確認する ACK の到達時間が伸びているということである。また、時間が経過するにつれ RTT 値の最大値が大きくなっているのは、 $n$  番目の RTT が確認され、 $n+1$  番目の RTT を確認するまでに、ネットワーク内でのデータの滞留が  $n$  番目の RTT を受信した時よりも大きくなり、データ転送に支障が生じ

ているからである。この事は、パケットキャプチャを行わなかった PC で、ストリーミングが途中で切れてしまったことから確認が取れる。スループットのグラフから、今回構成したネットワークでは 1Mbps 以上の帯域を安定して得るのが難しいネットワークだというのがわかる。ビットレートを上げすぎた場合、帯域不足で接続が切れてしまうということが推測される。同様に、PeerCast で組んだ P2P が、帯域不足の為疑似的なパケットリレーになったのもこの為だと言える。

## 6 おわりに

本研究では、P2P ストリーミングにおけるジッタの変動の特性を、シミュレーションではなく実際にネットワークを構築して検証した。先行研究ではネットワークシミュレーターを用いて大規模な仮想ネットワークを構築し測定したが、本研究では小規模ながらも実ネットワーク環境を用いて測定した。それにより、仮想ネットワークでは仮定を元にネットワークを構築する為、実際の環境で同じ環境を再現した時に、無視できない誤差が生じるというリスクを常に意識しなければならない。しかし、本研究は実環境で測定したので、そのような誤差は出ないと言える。反省点としては、今回の研究ではジッタを直接測定する事が出来なかった。RTT 値からジッタを推測する事に留まったので、今後はジッタを直接測定出来るような実験を行いたい。

## 参考文献

- [1] Masahiro Sasabe, Yoshiaki Taniguchi, Masayuki Murata, and Hideo Miyahara, "Proxy caching mechanisms with quality adjustment for video streaming services," IEICE Transactions on Communications Special Issue on Content Delivery Networks, Vol.E86-B, pp.1849-1858, 2003.
- [2] Xiaosong Lou, Kai Hwang, Gaogang Xie, "Quality of Service in Peer-to-Peer IPTV Networks," 2009 15th International Conference on Parallel and Distributed Systems, pp.9-17, 2009.
- [3] 川上 貴宏, 岸田 崇志, 河野 英太郎, 前田 香織, "高品質動画伝送を想定したネットワーク性能評価ツールの開発," 情報処理学会研究報告.DSM[分散システム/インターネット運用技術], Vol.38, pp.1-6, 2003.
- [4] Xiaojun Hei, Yong Liu, Keith W. Ross "IPTV over P2P Streaming Networks: the Mesh-pull Approach," IEEE Communications Magazine, Vol.46, pp.86-92, 2008.