Vol.2009-EVA-29 No.1
2009/8/5

端末伝送型インターネット放送における再生中断に関する評価

義久智樹[†1]　西尾　章治郎[†2]

近年のインターネット放送の普及にともない端末伝送型のインターネット放送が普及している．端末伝送型インターネット放送では，音楽や映像といったストリーミングデータを再生端末間で送受信する．再生開始時刻までに必要なデータの受信が間に合わない場合，再生に中断が発生するため，再生中断時間を削減する手法が提案されている．これらのほとんどは，受信先となる再生端末が長時間変わらないため，受信先の再生端末が必要なデータを持っていなければ再生中断時間が長くなるという問題があった．本研究では，受信先となる再生端末を頻繁に変えて再生中断時間を短縮する手法を提案し，評価を行う．既存手法と比較した結果，提案手法を用いることでさらに再生中断時間を短縮できることを確認した．

Evaluation of Play Interruption for Node Relay Based Webcast

Tomoki Yoshihisa[†1] and Shojiro Nishio[†2]

Due to the recent development of webcasts, node relay based webcast has been attracted great attention. In node relay based webcast, streaming data such as music and movies are transmitted between nodes. The continuity of the play is interrupted if nodes do not receive the necessary data until the time to start playing it. To reduce the interruption time, several methods are proposed. However, most of them have a problem that the interruption time lengthens when the received node has no necessary data since the receiving node is not reselected for a long time. In this paper, we propose a method to reduce interruption time by selecting the receiving node frequently. Our comparison with a previous method presents that out proposed method can give a shorter interruption time.

1. はじめに

近年のインターネットの普及にともない，音楽や映像といったストリーミングデータをインターネットで配信するインターネット放送が普及している．インターネット放送では，ストリーミングデータを再生する端末（再生端末）の数が非常に多く，従来のサーバクライアント型の動画配信では，再生端末にデータを配信する配信サーバの負担が大きくなる．このため，端末伝送型のインターネット放送が注目されている．端末伝送型インターネット放送では，配信サーバだけでなく再生端末間でも送受信しながら再生することで，配信サーバだけが各再生端末にデータを送信する場合と比べて配信サーバの負荷を軽減できる．本研究では，ライブ配信ではなくオンデマンド配信を端末伝送型インターネット放送で行う場合を対象とする．例えば，ストリーミングデータの初めの 5 分のデータを受信完了して再生している再生端末は，他の再生端末に初めの 5 分のデータを送信できる．

端末伝送型インターネット放送では，一般に，ストリーミングデータをデータサイズの等しい幾つかの部分に分割して送受信する．分割されたデータを**ピース**と呼び，ピースを配信する配信サーバや再生端末を**シーダ**と呼ぶ．再生端末は，各ピースの再生開始時刻までに受信を完了することでストリーミングデータを最初から最後まで途切れずに再生できる．しかし，送受信のための帯域が十分確保できない場合，再生開始時刻までにピースの受信を完了できず，データの再生が中断されることがある．

このため，再生中断時間を削減する幾つかの手法が提案されている．再生中断時間とは，再生開始までの待ち時間も含め再生が中断されている時間を指す．これらの手法のほとんどは，再生開始時にシーダを決定すると，シーダがネットワークから切断されるまで，同じシーダからピースを受信している．これは，必要なピースをもつシーダに接続しなおしてピースを受信すると，ピースを多くもつシーダに負荷が集中するためである．しかし，シーダが必要なピースをもっていない場合には，シーダがそのピース受信するまで待つことになり，再生中断時間が長くなるという問題がある．

そこで本研究では，受信先となる再生端末を頻繁に変えて再生中断時間を短縮する手法を提案する．提案手法では，各ピースの受信を開始する際，そのピースの受信にかかる時間を

†1 大阪大学 サイバーメディアセンター
Cybermedia Center, Osaka University
†2 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University

Vol.2009-EVA-29 No.1
2009/8/5

すべてのシーダに対して推測し，早く受信できるシーダからピースを受信する．ピースを早く受信できるため，再生中断時間を短縮できる．既存手法と比較した結果，再生端末の数が少なく，ネットワーク内に存在するピースの数が少ない場合にはシーダに負荷が集中して再生中断時間が長くなるが，再生端末の数が多くなると，提案手法の方が再生中断時間を短縮できていることを確認した．

以下，2 章で端末伝送型インターネット放送を説明する．3 章で提案手法を説明し，4 章で提案手法の評価を行う．6 章で関連研究を紹介し，最後に，6 章で本論文をまとめる．

2. 端末伝送型インターネット放送

端末伝送型インターネット放送の概要について述べる．文献[1] と同様の説明ではあるが，理解を深めるためここで説明する．

2.1 概　　要

端末伝送型インターネット放送では，一般に，ストリーミングデータを幾つかのピースに分割して送受信する．例えば，768Kbps の MPEG4 で符号化された 30 分の映像を配信する場合，ピースのデータサイズを 256K バイトとすると，675 個のピースに分割される．ピースは受信完了と同時に再生できる．ピースを受信している再生端末を**リーチャ**と呼び，ピースをもちリーチャにピースを送信できる再生端末をシーダと呼ぶ．リーチャは同時にシーダになり得るが，リーチャがすべてのピースを受信するとシーダとしてのみ機能する．常にネットワークにつながっておりすべてのピースをもつシーダを**スーパーシーダ**と呼ぶ．リーチャやシーダ，スーパーシーダは複数存在する．

端末伝送型インターネット放送では，**トラッカ**と呼ばれるサーバによってネットワークに参加している再生端末が管理されている（**図 1**）．再生端末がストリーミングデータを受信する際には，まずトラッカに問い合わせてシーダの宛先を取得する．トラッカの宛先は Web ページ等で確認する．取得したシーダの中から適切なシーダを選択してピースを受信する．現在普及しているダウンロード型のインターネット放送システム BitTorrent[2] は上記のような端末伝送型インターネット放送システムを用いており，実際にサービスが開始されているシステムである．

2.2 再生中断時間を短縮する意義

ストリーミングデータの視聴方法には 2 通りある．本論文では，ピースの受信が再生に間に合わなかった場合，ピースを受信完了するまで待ってから再生を続けるとしている．一方，受信が間に合わなかったピースに含まれる部分の再生を諦めて，飛ばして再生することも考えられる．前者の場合，再生中断時間が発生するが，視聴者はすべてのデータを視聴できる．後者の場合，データの再生が途切れるため再生中断自体は発生するが，中断が継続されることはなく，再生中断時間は発生しない．しかし，データを飛ばして再生するため，視聴者はすべてのデータを視聴できない．例えば，映画などのようにじっくりと内容を鑑賞したい場合には前者が適しており，ニュースのように大まかな内容だけ把握できればよい場合には後者が適していると考える．

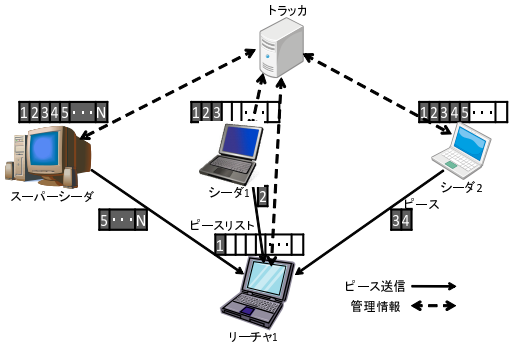

図 1　端末伝送型インターネット放送システム
Fig. 1　A node relay based webcast system

前者では，再生中断時間が長くなるほど視聴者は再生にストレスを感じる．どれほどの再生中断時間が現実的であるかは主観的になるが，再生開始までデータと同じ長さだけ待つ等長すぎると視聴者は視聴を諦めると考えられ，再生中断時間を短くすることが望ましい．

2.3 提案手法のアプローチ

端末伝送型インターネット放送において再生中断時間を短縮するためには，多くのピースを再生開始時刻までに受信完了できるようにシーダからピースを受信する必要がある．提案手法では，多くのピースを受信完了できるように早くピースを受信できるシーダからピースを受信する．各再生端末は前の方のピースから再生しているため，途中のピースを飛ばして受信することはない．このため，リーチャは，早くピースを受信できるシーダからピースを受信すると同時に，自身の再生位置に近いピースを受信することになり，前の方のピースを受信することになる．

しかし，ネットワーク内に存在するピースが少ない場合には，多くのピースをもつシーダに負荷が集中し，通信帯域が分断され，ピースの受信にかかる時間が長くなるという問題が

発生する．そこで，あるシーダが自分よりも再生開始時刻の早いリーチャにピースを送信している場合には，そのシーダから受信しないようにする．こうすることで，シーダの負荷を抑えた上でピースを送信できる．

3. 提案手法

端末伝送型インターネット放送における再生中断時間短縮手法 DTCast (Download Time based webCast，ディーティーキャスト) を提案する．本章では，まず DTCast の概要について述べ，次に想定環境，DTCast の詳細を順番に説明する．

3.1 概　　要

DTCast では，要求するピースの受信にかかる時間を推測し，最も早く受信できるシーダからピースを受信する．各再生端末は前の方からピースを受信するため，飛ばしてピースを受信することはなく，リーチャは，再生位置に近いピースを受信できる．

3.2 想定環境

本研究では以下の環境を想定する．2.1 節で説明した言葉を用いている．

- トラッカは再生端末の接続状況を管理している．
- 再生端末は 1 対 1 の通信を行う．
- 再生端末は受信したすべてのピースを保存できる．
- 再生端末はピースを受信完了と同時に再生できる．
- 再生端末は各ピースを受信完了してから他の再生端末に送信できる．
- 再生端末はストリーミングデータを最初から最後まで通常速度で再生する．
- 再生端末は再生終了を終了するとネットワークから切断する．

トラッカが管理する接続状況には，接続している再生端末や，その再生端末と他の再生端末との帯域幅が含まれる．自身と他の再生端末との間の帯域幅の取得方法として，ストリーミングデータの受信要求を出す前に調べておくことや，受信要求を出す時点に調べることで取得できる．他の再生端末間の帯域幅は，各再生端末が帯域幅の情報をトラッカに送信して他の再生端末がストリーミングデータの受信を開始する時点で取得することや，各再生端末が余裕時間を計算する際に必要な帯域幅の情報を取得することが考えられる．

3.3 シーダの決定方法

DTCast においてシーダを決定するアルゴリズムを**図 2** に示す．リーチャ i はピースの受信を終了するたびに，すべてのシーダに対してピースの受信にかかる時間を推測し，最も早く受信できるシーダからピースを受信する．ピースを受信する候補となるシーダをシーダ j とする．まず，リーチャ i がもっておらずシーダ j がもっている初めの方のピース p を探す．ピース p がなければ，リーチャ i は一定時間後に再度シーダの探索を行う．この一定時間はシステムのパラメタとして設定される．ピース p があれば，そのピースの受信にかかる時間 DownloadTime を計算する．DownloadTime は 3 個の待ち時間の合計となる．初めに，現在シーダ j が送信中のピースの送信完了にかかる時間が挙げられる．この待ち時間はシーダ j に問い合わせることで取得できるが，帯域幅が一定とは限らないため，実際の待ち時間とは異なり，推測値となる．アルゴリズム中では，GetFinishSendingTime() 関数で取得している．次に，シーダ j からの送信を待っているリーチャへピースを送信するのにかかる時間が挙げられる．幾つかのリーチャが待っている可能性があるため，待っているすべてのリーチャへの送信にかかる時間を計算する必要がある．この待ち時間は CalculateDownloadTime() 関数で取得している．最後に，リーチャ i がシーダ j からピース受信するのにかかる時間である．これらの合計がこれまでに候補としたシーダからピースを受信するのにかかる時間

```
When reacher i finishes a piece reception:
input: i          output: TargetSeeder
for j ∈ seeders do //すべてのシーダについて
   p=FindPiece(i,j) //リーチャi がもっておらず,
                                    シーダ j がもっている初めの方のピース p を取得
   if p is not null then
      if currently sending node of j is not i then
         //j が現在送信中のピースの送信完了までの時間
         DownloadTime=GetFinishSendingTime(j)
         //j からの受信を待っているリーチャへの送信にかかる時間
         while k ∈ waiting list of j do
            DownloadTime+=CalculateDownloadTime(k,j);
         end while
         //i が j からピースを受信するのにかかる時間
         DownloadTime+=CalculateDownloadTime(i,j)
         if DownloadTime<MinDownloadTime then
            MinDownloadTime=DownloadTime;
            TargetSeeder=j;
         end if
      end if
   end if
end for
if TargetSeeder is null then
   //一定時間後にリトライ
   Retry();
end if
```

図 2 DTCast のリーチャの接続アルゴリズム
Fig. 2 The algorithm of reacher's connection under DTCast

Vol.2009-EVA-29 No.1
2009/8/5

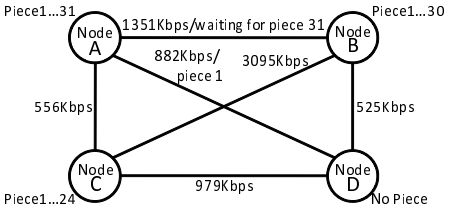

図 3 DTCast の具体例
Fig. 3 An example of DT Cast

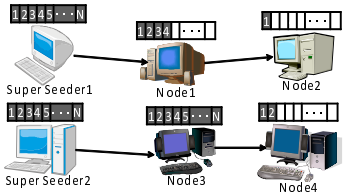

図 4 XebCast のピースの送受信のイメージ
Fig. 4 An image of transmitting peaces under XebCast

表 1 評価に用いたパラメタ
Table 1 Parameterf for evaluations

パラメタ	値
シミュレートした時間	4 時間
データの再生速度	512Kbps
データの再生時間	30 分
ピースのデータサイズ	256K バイト
スーパーシーダの数	1
受信要求の平均到着間隔	30 秒
平均帯域幅	1072Kbps
帯域幅の分散	1.01^2 $Kbps^2$
帯域幅の揺れ	5%
通信オーバヘッド	20 バイト

MinDownloadTime より小さい場合，MinDownloadTime を更新し，シーダ j からピースを受信することとする．

3.4 具 体 例

図 3 を例に具体的なシーダの決定方法を説明する．再生端末 A，B，C，D が順に再生を開始して現在，A はピース 1 から 31，B はピース 1 から 30，C はピース 1 から 24，D はピースをもっていない状況を考える．AB 間の帯域幅は 1351Kbps，AC 間は 556Kbps，AD 間は 882Kbps であり，BC 間は 3095Kbps，BD 間は 525Kbps，CD 間は 979Kbps である．ピースのデータサイズは 256K バイトとする．A は D にピース 1 を送信しており，B は A からピース 31 を受信するのを待っており，C がこれからピース 25 を受信するシーダを決定することを考える．まず，A を候補として考える．A はピース 31 までもっているため，$p = 25$ となる．A は D にピースを送信中であり，その送信完了までの時間は，平均して $256K \times 8/882K/2 = 1.16$ 秒になる．さらに B が送信を待っており，A が B にピースを送信するのに $256K \times 8/1351K = 1.52$ 秒となる．最後に C が A からピース 25 を受信すると，受信に $256K \times 8/556K = 3.69$ 秒かかることになり，DownloadTime=1.16+1.52+3.69=6.37 秒になる．次に，B を候補として考える．B はピース 30 までもっているため，$p = 25$ となる．B はピースを送信しておらず，送信待ちの再生端末もいないため，DownloadTime は，C が B からピース 25 を受信する時間となり，$256K \times 8/3095K = 0.66$ 秒になる．D は C がもっていないピースをもっていないため，候補とならない．結局 C は B からピースを受信する場合が最も早くピースを受信完了できるため，C は B からピース 25 を受信することになる．

4. 評　　価

本章では，DTCast の有効性を示すため，受信先となるシーダが長時間変わらない手法の中で再生中断時間を最も短縮できる XebCast と比較評価を行う．

4.1 XebCast

Xebcat では，列構造の送信経路を作成し，この経路に従ってピースを送信する．各リーチャは，必要とするピースを接続先のシーダがもっていない場合，受信するまで待つことになる．XebCast では，列構造の送信経路作成のために余裕時間と呼ぶ，再生に途切れが発生するまでの余裕を指標として，余裕時間が長くなるように列構造を作成している．XebCast のピース送受信のイメージを**図 4** に示す．このイメージではスーパーシーダが 2 個あり，スーパーシーダ 1 は再生端末 1 にピースを送信し，再生端末 1 は再生端末 2 にピースを送信する．スーパーシーダ 2 は再生端末 3 にピースを送信し，再生端末 3 は再生端末 4 にピースを送信している．

4.2 評 価 環 境

本章では，**表 1** の値を用いる． 時刻 0 で初めのリーチャが受信要求を出して接続し，順番に各リーチャが受信要求を出してデータの受信を始める．4 時間後までシミュレートした．データの再生速度は 512Kbps，ピースのデータサイズは 256K バイトで BitTorrent のデフォルトの値である．再生端末の再生要求の平均到着間隔はポアソン分布で与えた．文献[1] と同じく，平均帯域幅が 1072Kbps，分散が 1.01^2 $Kbps^2$ の対数正規分布[4),5)] で再生端末間の帯域幅を与えた．帯域幅の揺れとは，DTCast において受信時間の計算や XebCast の余裕時間の計算で用いる帯域幅と，実際にデータの配信で用いられる帯域幅との誤差を示す．再生端末が通信を開始する際，TCP のヘッダーサイズと同じ 20 バイトのオーバヘッド

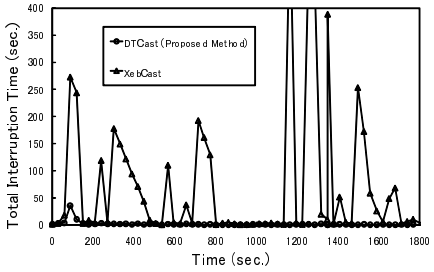

図 5 再生端末の受信開始時刻と再生中断時間
Fig. 5 The time for starting receiving the data and the total interruption time

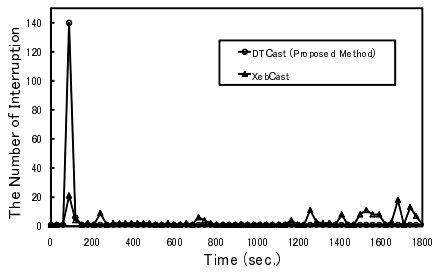

図 6 再生端末の受信開始時刻と再生中断回数
Fig. 6 The time for starting receiving the data and the number of interruptions

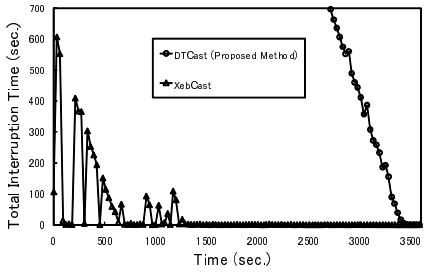

図 7 再生端末の受信開始時刻と再生中断時間（帯域幅が狭い場合）
Fig. 7 The time for starting receiving the data and the total interruption time (low bandwidth)

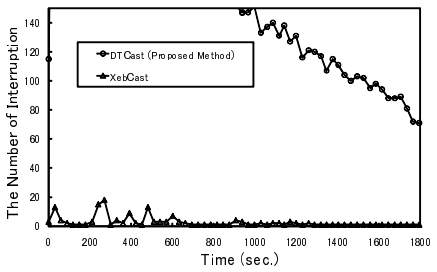

図 8 再生端末の受信開始時刻と再生中断回数（帯域幅が狭い場合）
Fig. 8 The time for starting receiving the data and the number of interruptions (low bandwidth)

が発生するものとしている．ストリーミングデータを最後まで再生すると再生端末がネットワークから切断されるとした．

4.3 再生端末の再生中断

各手法における再生端末の再生中断時間を**図 5** に示す．横軸は再生端末の受信開始時刻であり，縦軸は各再生端末の再生中断時間合計である．DTCast は提案手法，XebCast は比較手法である．図を見やすくするため，シミュレーションした最初の 30 分のみ示している．このグラフより，DTCast の合計再生中断時間が XebCast より短くなっていることが分かる．これは，DTCast において，出来る限り早くピースを受信できるようにシーダを選択することが効率的に働いているためである．また，データの配信開始後の 87 秒に受信を開始した再生端末の合計再生中断時間が 36 秒になっており，他の再生端末よりも比較的長くなっていることが分かる．開始直後は，ネットワーク内のピースの分布状況が十分でなく，各再生端末はスーパーシーダからピースを受信する場合が多くなる．このため，再生端末とスーパーシーダの通信帯域が小さくなってしまい，再生開始時刻に間に合わないピースが多くなるためである．その後は，ほぼ安定しており，合計再生中断時間は数秒になっている．

また，再生中断回数を**図 6** に示す． 縦軸が各再生端末の再生中断回数である．多くの場合 DTCast の再生中断回数が XebCast に比べて少ないことが分かる．しかし，合計再生中断時間の場合と同じく，配信開始から 87 秒後に受信を開始した再生端末の再生中断回数が比較的多くなっている．上述したように，ピースの受信が再生に間に合わなかったためである．特に通信帯域が再生レートより小さく，頻繁に再生中断が発生している．

一方，配信が始まったばかりの段階で帯域幅の狭い再生端末がデータを受信していると，スーパーシーダからのピースの配信が遅くなり，待ち時間が非常になる問題が発生する．帯域幅が狭い場合の再生中断時間と再生中断回数を**図 7** と**図 8** に示す．図 5 や図 6 との違いは，発生させる乱数の基数を変更しただけである． 最初の結果では，DTCast が XebCast より再生中断時間および再生中断回数が少なくなっていたが，スーパーシーダ付近の帯域幅が狭い場合には，XebCast の方が良い性能を与えている．これは，データの配信開始直後の段階で帯域幅が小さい再生端末が多くなり，ネットワーク内のピースの分布が滞るためである．

このため，単にピースを早く受信するだけでなく，ネットワーク内のピースの配布状況を考慮しつつピースを受信するシーダを決定する必要があるといえる．

5. 関 連 研 究

端末伝送型インターネット放送において，再生中断時間を短縮する手法が幾つか提案されている．

SplitStream[6] や CoopNet[7] では，再生端末で木構造の送信経路を作成し，送信経路に沿ってピースを送信している．送信経路を効率よく作成することでピースを速く送信でき，中断時間を削減できる．木構造の送信経路では，再生が終了して再生端末がネットワークから切断した場合に木を再構築する必要があるため，メッシュ構造の送信経路を作成する Chainsaw[8] や PRIME[9] が提案されている．木構造やメッシュ構造では，複数の再生端末にピースを送信することになるため，リーチャあたりのシーダの帯域幅が少なくなり，ピー

スの受信完了までにかかる時間が長くなるという問題がある．

XebCast[1] では，再生端末で列構造の送信経路を作成している．送信経路を列構造にすることで，シーダは全可用帯域を用いてピースを送信できるため，リーチャに早くピースを送信できる．結果，木構造やメッシュ構造に比べて再生中断時間を短縮できることが示されている．

これらの手法では，送信経路が固定されており，各再生端末は再生に必要なピースが送信されるまで待つ必要がある．送信経路上にない再生端末が必要なピースをもっていても受信できないため，このような状況で再生中断時間を削減できない．

再生開始時刻までに受信完了できなかったピースの受信を諦め，そのピースを飛ばして再生する環境のもと，再生中断回数を削減する研究が行われている．CoolStreaming[3] では，再生端末はストリーミングデータの受信を開始する際，ピース毎に受信にかかる時間をすべてのシーダに対して計算して最も早く受信できるシーダからピースを受信できるように，ピースの受信スケジュールを作成する．どのシーダからピースを受信しても再生開始時刻までに間に合わない場合には，そのピースの受信を諦める．ピースの受信スケジュールに含まれるシーダに対してピースの送信を予約しているため，ほぼ計算通りにピースを受信でき，中断回数を削減できる．しかし，間に合わなかったピースを再生できないため，視聴者はデータをすべて視聴できない．

本論文の提案手法では，シーダを固定せず，ピースの受信を開始する度に選択しなおす点が新しく，再生に間に合わなかったピースを受信できるまで待つため，視聴者がデータをすべて視聴できないということもない．

6. ま と め

本論文では，端末伝送型インターネット放送における再生中断時間短縮手法 DTCast を提案した．DTCast では，各ピースの受信を開始する際，そのピースの受信にかかる時間をすべてのシーダに対して推測し，早く受信できるシーダからピースを受信する．ピースを速く受信できるため，再生中断時間を短縮できる．既存手法と比較した結果，再生端末の数が多くなると，提案手法の方が再生中断時間を短縮できることを確認した．

今後，端末伝送型インターネット放送で選択型コンテンツ[10] を配信する場合やランダムに広告を配信する場合の再生中断時間短縮手法を考えている．

謝辞 本研究は，科学研究費補助金基盤研究 (S)（課題番号：21220002）および科学研究費補助金若手研究 (B)（課題番号：21700108）によるものである．ここに記して謝意を表す．

参 考 文 献

1) 義久 智樹, 西尾 章治郎: “端末伝送型インターネット放送における再生中断時間短縮手法,” 情報処理学会論文誌, Vol.50, No.9 (2009, 採録決定).
2) B.Cohen: “Incentives build robustness in BitTorrent,” in Proc.Work.on Economics of Peer-to-Peer Systems (P2PEcon 2003) (2003).
3) X.Zhang, J.Liu, B.Li: “DONet/CoolStreaming: A Data-driven Overlay Network for Live Media Streaming,” in Proc.IEEE INFOCOM2005, Vol.3, pp.2102–2111 (2005).
4) W.Chuan, L.Baochun, Z.Shuqiao: “Characterizing Peer-to-Peer Streaming Flows,” in IEEE Journal on Selected Areas in Communications, Vol.25, No.9, pp.1612–1626 (2007).
5) S.Saroiu, K.P.Gummadi, S.D.Gribble: “A Measurement Study of Peer-to-Peer File Sharing Systems,” in Proc.of SPIE/ACM Multimedia Computing and Networking Conf. (MMCN 2002), pp.156–170 (2002).
6) M.Castro, P.Druschel, A.-M.Kermarrec, A.Nandi, A.Rowstron, A.Singh: “SplitStream: High-bandwidth Content Distribution in a Cooperative Environment, in Proc.Int’l Wwork.on Peer-To-Peer Systems (IPTPS 2003), pp. 298–313 (2003).
7) V. N. Padmanabhan, H. J. Wang, P. A. Chou, K. Sripanidkulchai: “Distributing streaming media content using cooperative networking,” in Proc. Int’l Work. on Network and Operating System Support for Digital Audio and Video (NOSSDAV 2002), pp.177–186 (2002).
8) V.Pai, K.Kumar, K.Tamilmani, V.Sambamurthy, E.E.Mohr: “Chainsaw: Eliminating Trees from Overlay Multicast,” in Proc.Int’l Wwork.on Peer-To-Peer Systems (IPTPS 2003),pp. 127–140 (2005).
9) N. Magharei, R. Rejaie: “PRIME: Peer-to-Peer Receiver-driven Mesh-based Streaming,” in Proc.IEEE INFOCOM2007 (2007).
10) 義久 智樹, 金澤 正憲: “選択型コンテンツの放送型配信におけるスケジューリング手法,” 情報処理学会論文誌, Vol.47, No.12, pp.3296–3307 (2006).