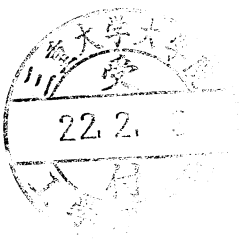
修士論文

データ駆動型
アプリケーション層マルチキャストの
配信手法の高速化について



平成 21 年度修了

三重大学大学院 工学研究科
博士前期課程 情報工学専攻

越賀 雅士

目次

はじめに

近年では，プロセッサの高速化，メモリの大容量化，ネットワークの広帯域化を背景として，多くのユーザが高画質の映像をインターネットを通して視聴できるようになった．しかし，従来のクライアント/サーバ型では，サーバへの負荷が集中することになり，大規模配信を行うには大規模サーバを必要としていた．これに対して，近年 Peer-to-Peer(P2P) 型での映像配信が注目されている．この映像配信の手法をアプリケーション層マルチキャスト (Application Layer Multicast, 以下 ALM) という．

マルチキャストの手法として一般的に「IP マルチキャスト」と「ALM」の 2 種類ある．IP マルチキャストは，データ複製の為マルチキャストをするノードの経路上のルータ全てがマルチキャスト対応のルータを用いる必要があるのでコストの面の問題で普及は難しい．ALM は IP マルチキャストで通常 IP 層で行われているマルチキャストの技術を，アプリケーション層で実現している手法である．データは ALM の各ノードで複製され，ネットワーク上の間ノードでユニキャストを用いてデータ転送が行われる．ALM ではマルチキャストを行うのに特別な装置を必要としないので，ALM によるマルチキャストが主流となってきている．従来，ALM ネットワークの構築及びデータ転送方法としてツリー型とメッシュ型の 2 種類が考えられていたが，近年ではデータ駆動型という新たなデータ転送方法の研究が盛んに行われている．

ツリー型 ALM はその名の通りデータ配信ノードを木の根として節ノード・葉ノードへとデータを転送していく手法である．メッシュ型 ALM ではツリー型で持っている親子の関係がなく，隣接するノードからデータを送受信することでネットワーク全体にデータを広めていく手法である．これに対して，データ駆動型 ALM はトポロジはメッシュ型と同じであるが，データを一方的に送る push 型の転送ではなくデータを要求する pull 型のデータ転送方式をとっている手法である．

そこで本研究では，ALM の転送方式の 1 つデータ駆動型に注目し，データ駆動型 ALM で制御パケット量を抑えつつ，ノードの上り帯域を効率良く使用することによって遅延を少なくする手法を提案する．

本論文では第 1 章において，P2P について述べる．第 2 章において，マルチキャストについて述べる．第 3 章において，ALM の転送方式について述べる．第 4 章において，関連研究について述べる．第 5 章において，データ駆動型 ALM の提案方式について述べる．第 6 章において，提案する手法の有効性を示す実験について述べる．

第 1 章

Peer-to-Peer(P2P)

本章では，P2P のネットワークトポロジについて述べる．1.1 において，P2P 型について述べる．1.2 において，クライアント/サーバ型について述べる．1.3 において，P2P 型の分類について述べる．

1.1　P2P 型

P2P とは Peer-to-Peer の略であり，Peer には「対等」と言う意味がある．つまり，「対等のもの同士が対等の立場で相互にやりとりを行う方式」である．クライアント/サーバ型ではアクセスが非常に多くなると，サーバがパンクすることがあるのに対して，P2P 型では元となるデータを一極集中していないのでアクセス数が多くなっても問題ない．P2P 型の通信例を図 1 に示す．

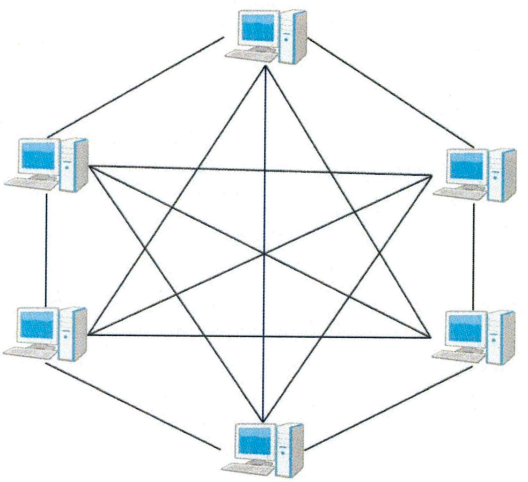


図 1　P2P 型

長所としては，処理が分散できるため，個々のコンピューターの負担が大きくなりにくい．また，個々で通信を行うため比較的秘密裏に処理を行えるため隠匿性が高いという点があげられる．また，短所として，処理が分散するために管理や監視がしにくいという点があげられる．

1.2　クライアント・サーバ型

P2P 型に対して従来からの通信形態としてクライアント・サーバ型がある．クライアント・サーバ型とは，各コンピュータの役割がサーバとクライアントに分かれているコンピュータネットワークのソフトウェアモデルである．サーバはデータの配信，蓄積，検索などのサービスを提供する役割を持っている．一方，クライアントはサーバに対してサービスを要求し，そのサービスを利用する役割である．このようにコンピュータの役割は固定され，変わることはない．クライアント・サーバ型の様子を図 2 に示す．

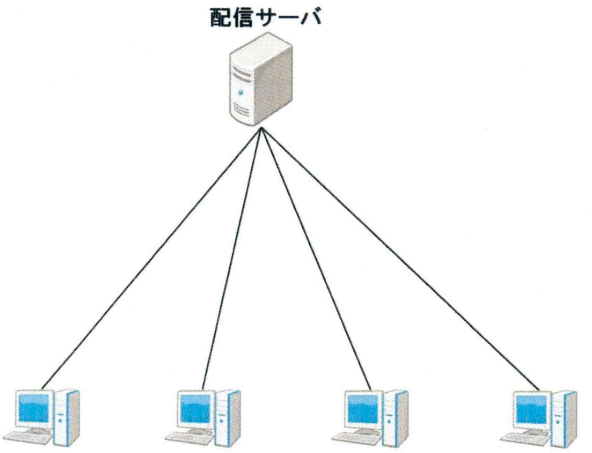


図 2　クライアント・サーバ型

長所としては，クライアントの処理が少なくてすむという点，システムの管理及び監視が行いやすいという点があげられる．また，短所として，サーバーの処理が非常に重くなるため，高価なコンピュータをそろえる為のコストが必要という点，サーバーに処理が集中するため，回線帯域を圧迫しやすくなるので大容量の回線が必要という点などがあげられる．

1.3　P2P 型の分類

P2P 型の構築方法を大別するとハイブリット P2P，ピュア P2P そしてスーパーノード型ハイブリット P2P の 3 種類ある．

1.3.1 ハイブリッド P2P

ハイブリッド P2P は，クライアント・サーバ型と P2P 型をハイブリッド（複合）した方式である．データの所在情報は固定的なサーバーに管理させ，各ノードはあらかじめ知っているサーバーに問い合わせることでデータ場所を知る．データの探索を行うサーバーのことを「インデックスサーバー」という．インデックスサーバーはデータの所在情報のみを扱う．データのやり取りはノード同士で行うという点がクライアント・サーバ型との違いである．ハイブリッド P2P の様子を図 3 に示す．

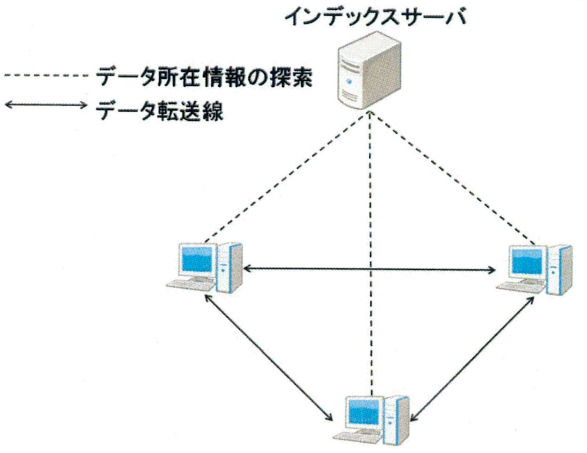


図 3　ハイブリッド P2P

長所としては，シンプルでありシステムを管理・制御しやすいという点があげられる．また，短所としては，所在情報だけではあるがシステムに中心を持つため，スケーラビリティや耐障害性が十分に発揮されないという点があげられる．

1.3.2 ピュア P2P

ピュア P2P は，ハイブリッド P2P とは異なりデータの所在情報を持つインデックスサーバーすら持たず，純粋にノード同士のみでシステムを形成する．探索方法としてはノード同士が助け合う形で実現している．通常ピュア P2P では，各ノードは近接のノードと定常的な接続をメッシュ状に張っている．探索には「フラッディング」と「DHT(Distributed Hash Table，分散ハッシュテーブル)」などがある．フラッディングとは，探索クエリ (探索要求のこと) を隣接ノードへバケツリレー式に伝搬し，見つかればその結果を再び伝搬することでデータの所在を知る．また DHT とは，ノードとデータにそれぞれ単一の ID が割り当てられる．ノードなら IP アドレス，データ

ならファイル名からハッシュ関数を使い ID を求め，データは ID が近いノードによって管理される．よって，欲しいデータの ID に近い ID を持つノードに問い合わせることによりデータの所在を知る事ができる．ピュア P2P の様子を図 4 に示す．

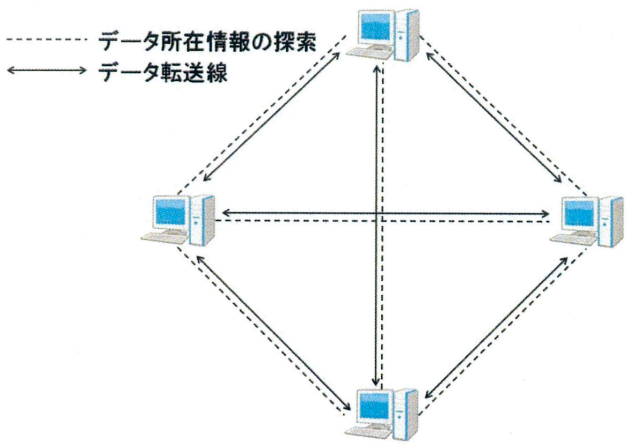


図 4　ピュア P2P

長所としては，純粋な P2P が実現できるためスケーラビリティや耐障害性などが高いという点があげられる．また，短所としては，実装が複雑になるという点，一度機能するとシステムの停止をすることが難しいという点などがあげられる．

1.3.3　スーパーノード型ハイブリット P2P

スーパーノード型ハイブリット P2P は，データ探索は一般ノードよりもスペックの高いスーパーノードと呼ばれるノードが行う．インデックスサーバーとの違いはスーパーノードは固定的ではなく，システム全体のノード数により自動的に数を調整し，スケーラビリティと冗長性を持っている点である．ピュア P2P では，全てのノードが自身と関係のない探索クエリの伝搬を常に行わなければならないので，スペックの低いノードの負担が大きくなることがある．しかし，スーパーノード型はスペックの低いノードがデータ探索処理をすることはないので負担はかからなくなる．スーパーノード型ハイブリット P2P の様子を図 5 に示す．

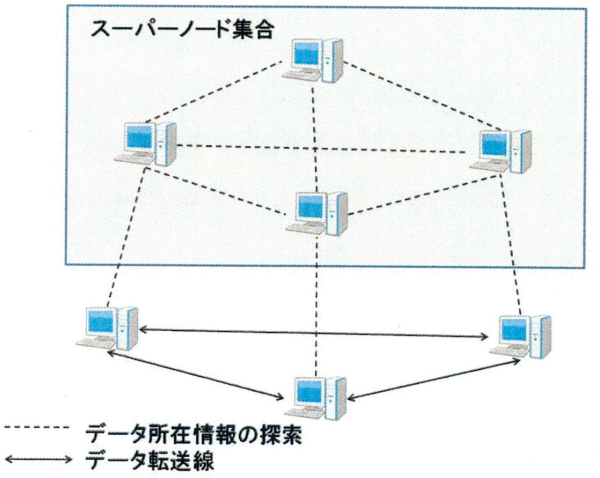


図 5　スーパーノード型ハイブリット P2P

長所としては，ハイブリッド P2P とピュア P2P の両方の良さを持つという点があげられる．また，短所としては，探索データの分散化などの実装が難しくなる．ノード数がある程度増えるまで安定しない可能性があるという点があげられる．

第 2 章
マルチキャスト

本章では，マルチキャストの概要について述べる．2.1 において，マルチキャストの概要について述べる．2.2 において，IP マルチキャストについて述べる．2.3 において，ALM について述べる．

2.1 概要

マルチキャストとは，ネットワーク内で複数の相手を指定して同じデータを送信するという通信である．これに対してネットワーク内の全員へ同じデータを送信することを「ブロードキャスト」と言い，特定の相手のみにデータを送信することを「ユニキャスト」という．マルチキャストの送信者はどこに居ても良く，受信者は複数存在するという状況が想定されている．マルチキャスト受信者は，「マルチキャストグループ」と呼ばれるグループに「join」することにより，データを受け取れるようになる．

マルチキャストには一般に 2 つの手法がある． 1 つは IP マルチキャストといい，OSI 参照モデルにおけるネットワーク層でマルチキャストを行う手法である．詳しくは 2.2 において説明をする．もう 1 つは ALM と言い，ネットワーク層ではなくアプリケーション層でマルチキャストを行う手法である．詳しくは 2.3 において説明をする．

2.2 IP マルチキャスト

IP マルチキャストとは，OSI 参照モデルにおけるネットワーク層を用いてマルチキャストを行う手法である．最小限のネットワーク帯域幅を使用して，送信元にも受信者にも負担をかけずに，アプリケーションのソーストラフィックを複数の受信者に配信する．マルチキャストは，グループという概念を基盤とする．マルチキャストグループとは，特定のデータストリームを受信したいという意向を表明した任意の受信者グループを表す．このグループには物理的または地理的な制約がなく，インターネットまたはプライベートネットワーク上のどの場所にホストが存在しても良い．IP マルチキャストを行うには，マルチキャストアドレスと呼ばれる特殊な IP アドレス (クラス D のアドレス，224.0.0.0〜239.255.255.255) を用いてデータが送信される．各マルチキャストグループ宛に送信されたパケットは，マルチキャストルーティングプロトコルにより設定された送信ノードを根，中継のルータを節，多数の受信ノードを葉とするツリー上を伝送され，各受信ノードに届く．中継のルータで，必要に応じてマルチキャストパケットの複製が行われる．また，マル

チキャストグループに属する受信ノードの管理情報を，送信ノードではなくネットワーク上のルータが行うことで，送信ノードの管理負荷の集中を回避している．

IP マルチキャストの様子を図 6 に示す．

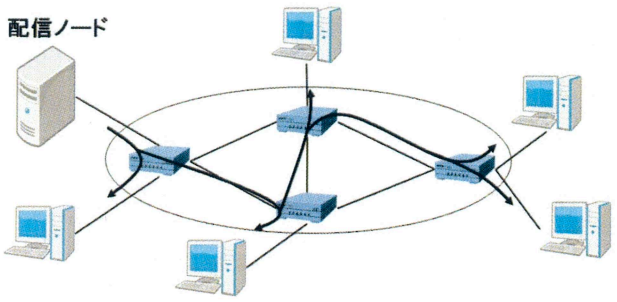


図 6　IP マルチキャスト

IP マルチキャストの長所として，1 つのデータをルータなどで複製をすることで複数の相手へ送信することにより，容量の大きなデータでも回線を圧迫することなく配信することができるという点があげられる．

また，短所として，ルーターでデータを複製するにはマルチキャスト対応のルータでないとできないという点があげられる．よって，送信先までの経路上のルータ全てがマルチキャスト対応ルータである必要がある．

2.3　ALM

ALM とは，その名の通りネットワーク層ではなくアプリケーション層でマルチキャストを行う手法である．IP マルチキャストではルータが行っていたマルチキャストに関する機能，即ちパケットの複製やマルチキャストルーティング，グループ管理などをアプリケーション層に組み込むことで実現している．ネットワーク層ではユニキャストのみをサポートする．また，ALM は P2P 型で通信を行い，ノード間はデータをユニキャストを用いてノード間を中継して全てのノードへ送信される．実際のパケット転送にはルータが介在しているが，論理的には送信ノードを根として，全受信ノードが節であり葉ともなる．各受信ノードで，マルチキャストに関する機能を提供する必要がある．

ALM の様子を図 7 に示す.

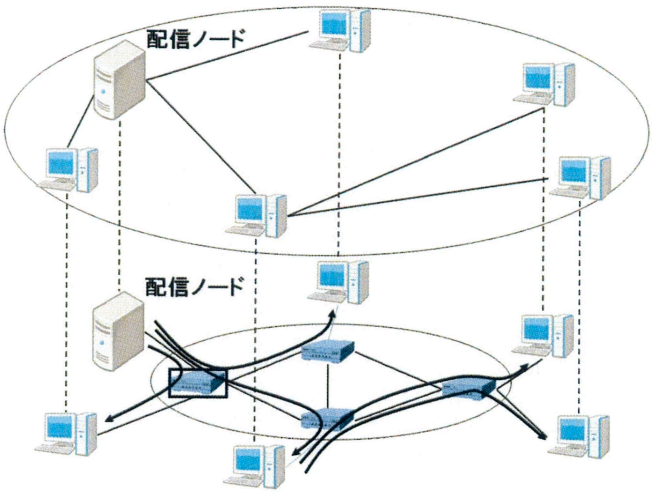


図 7　ALM

長所として，IP マルチキャストと異なりネットワーク層ではユニキャストしか行わないので経路上にマルチキャスト対応ルーターがなくても全てのノードにデータを送信できるという点があげられる．また，P2P によるデータ送信であるから強力なサーバを設置しなくてよいので，サーバ・クライアント型の場合に比べ負荷を分散できるという点があげられる．

短所として，今までネットワーク層で対応してきた機能をアプリケーション側で行うので，開発するアプリケーションが複雑になるという点があげられる．また、IP マルチキャストに比べて実ネットワークのトラフィックが増大するという点があげられる．

第 3 章
ALM の転送方式

本章では，ALM の転送方式について述べる．ALM のネットワーク構築方法として 3 種類の構築法が考えられる．ツリー型 ALM とメッシュ型 ALM，そしてデータ駆動型 ALM である．3.1 において，ツリー型 ALM について述べる．3.2 において，メッシュ型 ALM について述べる．3.3 において，データ駆動型 ALM について述べる．

3.1 ツリー型 ALM

ツリー型は，ノードがマルチキャストのためのデータ配信木を作成し，木構造に沿って根から葉に向かってデータを配信する．データがネットワーク上を流れる前に流路が決まっているため，データがあるノードに到達するまでの時間を低く抑えやすい手法である．ツリー型 ALM を用いたものに，CoopNet[1]，Narada[2] などが挙げられる．ALM では，各ノードが受信したデータを他のノードへ提供するので，受信用の下り帯域幅だけでなく，送信用の上り帯域幅が重要となる．例えば，500kbps のデータを受信している場合，上り帯域幅が 500kbps なら 1 ノードにしかデータ転送ができないが，5Mbps なら 10 ノードに同時にデータ転送ができる．ツリー型 ALM では子を持たないノード，つまり葉ノードの場合，他のノードへデータ転送を行わない．つまり，上り帯域幅を活用していない状態のノードが存在する．木構造に置いては葉ノードは案外多く，2 分木でも半分強は葉ノードとなる．このことは，上り帯域幅を重要視する ALM では大きな問題である．そこで，木構造でこの問題を解決する為に，複数の Scribe[3] と呼ばれる配信木を構築し，それぞれでデータの一部を流す手法である SplitStream[4] が提案された．

ツリー型 ALM の様子を図 8 に示す．

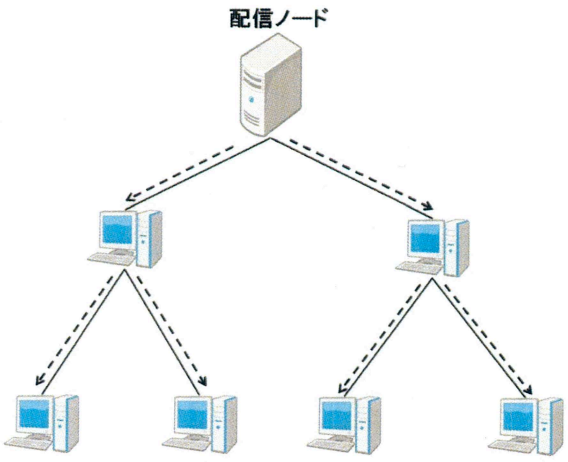


図 8　ツリー型 ALM

利点として，データ配信経路の管理や操作，最適化を行いやすい点がある．欠点として，ノードの離脱などが起こると素早く修復する必要があり，もし修復に問題があればデータ配信に失敗することがある．もし，故障，離脱したノードがデータ配信木の根の近くなら，その影響は大きなものとなる．

3.2　メッシュ型 ALM

メッシュ型は，ツリー型が持つ親子関係の制約を無くし，複数の隣接ノードへ flooding でデータを転送する．データの受信が初めてであれば，受信したノードを除いた全ての隣接ノードに対してデータを転送する．受信済みであれば転送しない方法をとっている．この flooding での手法では，無駄なデータ転送の量が多いという問題がある．flooding を用いてかつ無駄なデータ転送の量を抑える手法として，gossip プロトコル [5] というものがある．gossip プロトコルは文字通り，噂が人づてに伝わる様子に似た動作を行う．基本的には，転送処理として隣接ノードの中から転送先をランダムに選んで転送する動作を繰り返す．受信済みのノードへ一定回数転送を行ったら，転送を止める．

メッシュ型 ALM の様子を図 9 に示す．

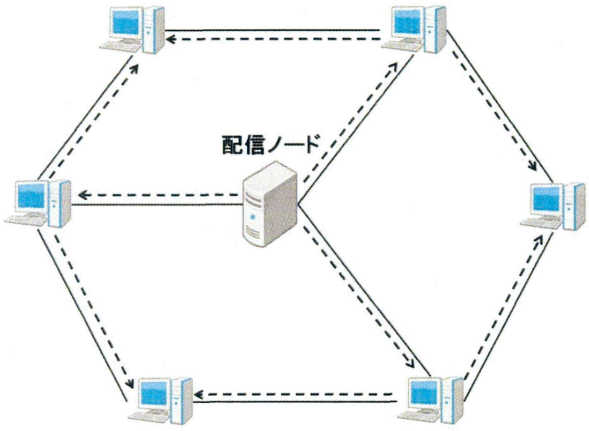


図 9　メッシュ型 ALM

利点として，他の隣接ノードからデータ受信ができるため，ノードの離脱に対して素早く修復する必要がない．欠点として，無駄なデータ転送が多いので，それを低減する工夫する必要がある．既存手法として Bullet[6] などが挙げられる．

3.3　データ駆動型 ALM

ツリー型・メッシュ型 ALM は基本的に push 型のプロトコルである．木構造では単一の親からしかデータを受け取らないので，pull 型の親への要求は無駄である．しかし，メッシュ型では flooding や gossip のように同一のデータが複数のノードから送信される．この無駄を省くために pull 型プロトコルは有効である．データ駆動型とは，メッシュ型のトポロジを構成し，データを隣接ノードへ一方的に送るのではなく，隣接ノードからの要求を受けて送信する pull 型の転送方式である．隣接ノードへデータ要求するには，隣接ノードのデータ保持情報が必要となるため，あらかじめデータ保持情報を隣接ノードに知らせておく必要がある．一般にデータ保持情報では，転送データを分割し，セグメント毎に分けてシーケンス番号を付け，その番号やビットマップで保持データを表現する．

データ駆動型 ALM の様子を図 10 に示す．

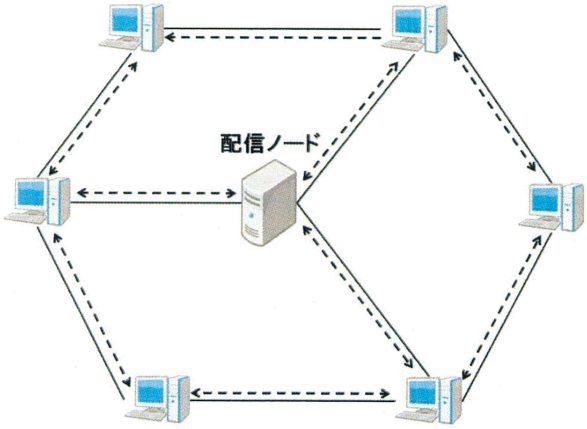


図 10　データ駆動型 ALM

利点として，メッシュ型で起こる無駄なデータ転送を無くすことができる．また，ノードの頻繁な出入りに強い性質を持っている．欠点として，要求を受けてから，データを送信するのでツリー型やメッシュ型に比べると遅延が大きくなる．また，データ保持情報などの制御パケットの通知頻度でも遅延の大小は変わってくる．

第 4 章
関連研究

本章では，データ駆動型 ALM の関連研究について述べる．4.1 において，データ駆動型 ALM の関連研究について述べる．4.2 において，CoolStreaming について述べる．4.3 において，Chainsaw について述べる．

4.1 データ駆動型 ALM の関連研究

データ駆動型 ALM の関連研究として，CoolStreaming や Chainsaw など [7][8][9] が挙げられる．どちらの場合もデータ転送方法として，まず転送するデータをセグメントと呼ばれる分割されたデータ片に分ける．その後，隣接ノード間でセグメント毎に送受信を行う．各ノードのセグメント保持の有無については，BitMap(以下 BM) と呼ばれるセグメント保持情報を定期的に送受信することで隣接ノードへ知らせることになっている．BM は 0 と 1 で表されるビット配列であり，0 で持っていないセグメント，1 で持っているセグメントを表している．オリジナルデータを持つデータ配信ノード (ソースノード) の BM は全て 1 となる．

また，近年では DONLE[10] などが提案され，盛んにデータ駆動型 ALM が研究されている．

以下では本研究で比較実験を行う CoolStreaming，Chainsaw のデータ転送方式と問題点について述べる．

4.2 CoolStreaming

CoolStreaming ではストリーミングデータを隣接ノードに転送する際には，ストリーミングデータを 1000msec 毎に分割したセグメントに分け，セグメント毎で隣接ノードとの送受信を行う．各ノードは自身の隣接ノードと一定時間毎に BM の交換を行う．

CoolStreaming のデータ転送の様子を図 11 に示す．

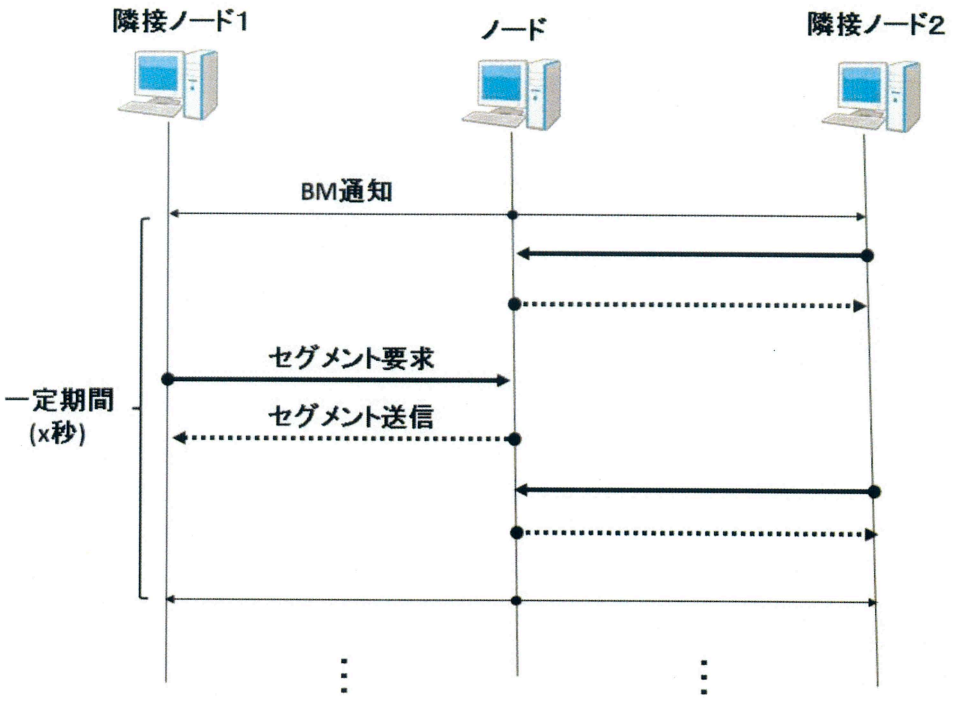


図 11　CoolStreaming のデータ転送の様子

1. 一定時間毎に隣接ノードへ BM の送信を行う.
   - この動作は一定時間経過しない限りアルゴリズム内で再び実行される事はない.
2. 隣接ノードから送られてくる BM を解析して，保持していないセグメントを持っているノードへセグメントの要求を出す．もし無ければ要求をしない.
3. 隣接ノードから要求したセグメントを受信したら，自身の BM を新たに更新する.
4. もし自身が設定した一定時間が過ぎていれば，隣接ノードへ BM の送信を行う.

(2)~(4) を繰り返し行うことでデータ送受信を行う.

この手法の問題点として，BM の通知頻度が高いと制御パケット量が増大してしまい帯域の無駄使いをしてしまうことが挙げらる．また，BM の通知頻度が低いと新たなセグメントを取得しても隣接ノードへの通知が遅れてしまい，隣接ノードのセグメント取得の時間が遅くなってしまうことが挙げられる.

4.3 Chainsaw

Chainsaw では転送データを隣接ノードに転送する際には，転送データを 16KB 毎に分割したセグメントに分け，セグメント毎に隣接ノードとの送受信を行う．各ノードは新たなセグメントを取得した時に，隣接ノードへ取得したセグメントの情報を知らせるための制御パケットを送信する．

Chainsaw のデータ転送の様子を図 12 に示す．

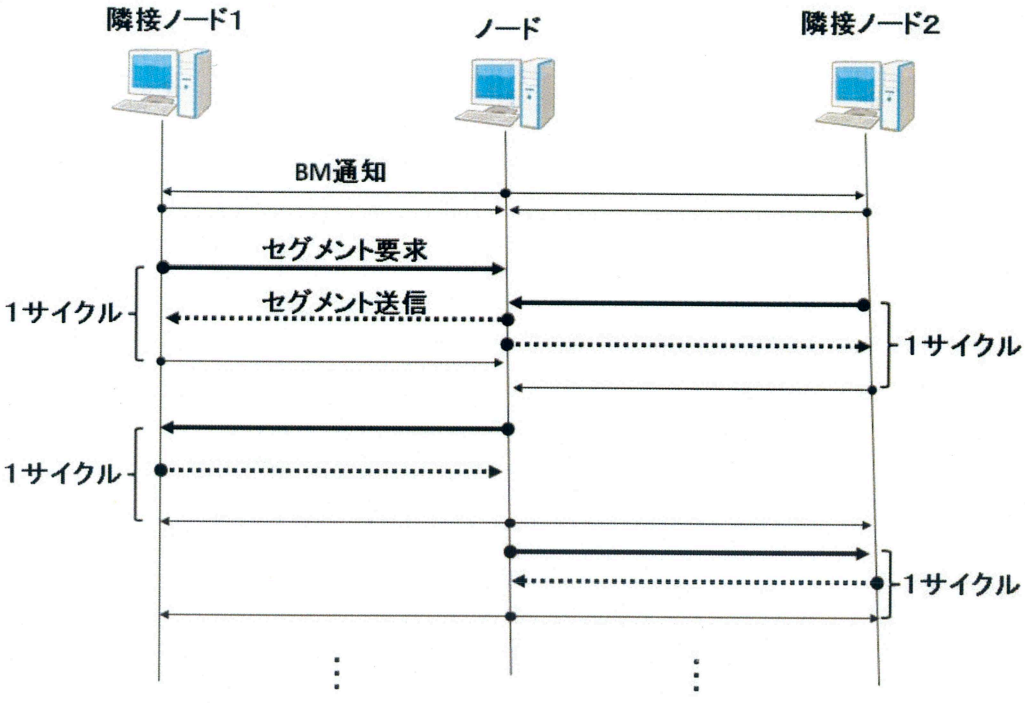


図 12　Chainsaw のデータ転送の様子

1. 隣接ノードへ BM を送信する．
2. 隣接ノードの BM を解析し，自分が持っていないセグメントを探す．
3. もし自分が持っていないセグメントがあれば，隣接ノードへ要求を出す．もしなければ何も要求しない．
4. 隣接ノードから要求したセグメントを受信する．
5. 隣接ノード全てに取得したセグメントの情報を送信する．

(2)〜(5) を繰り返し行うことでデータ送受信を行う．また，各ノードが保持している隣接ノー

ドの BM は，(1)～(5)の動作を実行している間でも同時進行で更新されていくので，問題なくデータ送受信が行える．

この手法の問題点として，セグメント受信毎に制御パケットを全ての隣接ノードへ送信することになるので，制御パケット量が増大してしまい帯域の無駄使いをしてしまうことが挙げられる．

第 5 章
提案手法

本章では，本研究で提案するデータ駆動型 ALM の手法について述べる．5.1 において，提案手法の概要について述べる．5.2 において，BM 送信タイミングについて述べる．5.3 において，要求セグメント解析について述べる．

5.1 提案手法の概要

CoolStreaming では一定時間毎に BM の送受信を行うことにより，BM 転送回数は一定数に抑えることができるが，新たなセグメントを取得してもすぐには隣接ノードへ通知されず，セグメントの送受信が一定期間されない場合がある．つまり，上り帯域の使用をうまく活用できていない問題がある．また，Chainsaw のように新たなセグメントを取得した時に BM の送受信を行う時では、上り帯域の使用効率は問題ないが，BM 送受信量が多くなってしまい，通信処理の負担が大幅に増えてしまう．

これらの欠点を補う為の手法として，新たなデータ転送方法とセグメント要求解析方法を提案する．

5.2 BM 送信タイミング

提案手法 [11][12] では，CoolStreaming や Chainsaw の欠点を補う為に，BM の送信のタイミングを新たに設定する．データ転送時における BM の送信を全ての隣接ノードへセグメントを送信し終えたときに行うことで，各ノードの上り帯域を効率良く使用し，BM の送信量を抑え，遅延を少なくできると考えられる．

提案手法のデータ転送の様子を図 13 に示す．

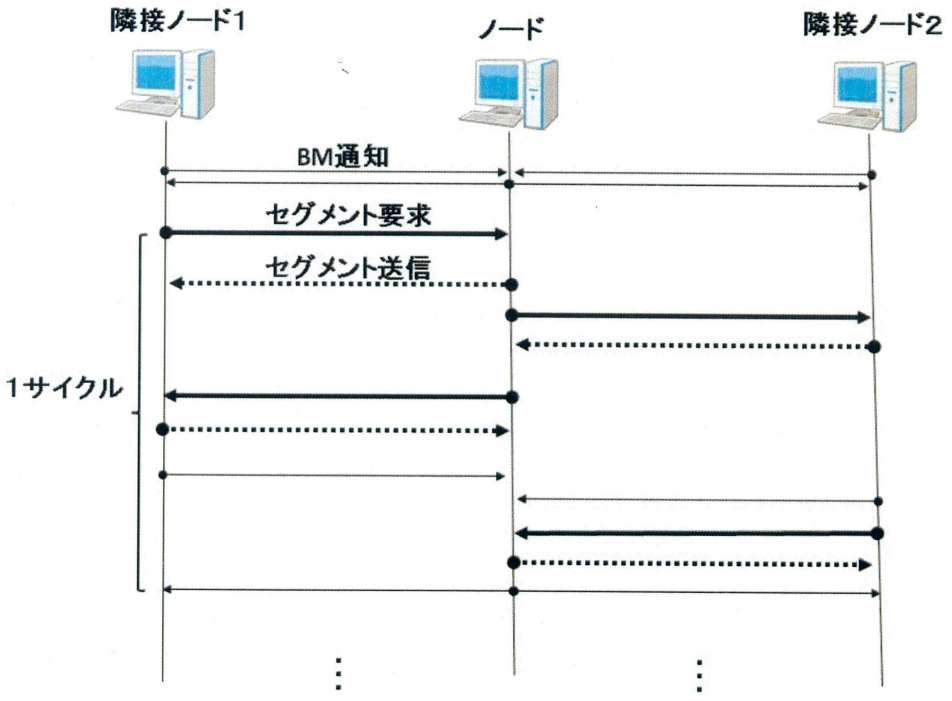


図 13　提案手法のデータ転送の様子

1. 各ノードは全ての隣接ノードへ BM を送信する
2. 全ての隣接ノードから BM を受信したら解析し，保持していないセグメントを隣接ノードに要求する
3. セグメントを受信する隣接ノードからの要求を受けたら，要求されたセグメントを隣接ノードへ送信する
4. 隣接ノードから要求したセグメントを受信する全ての隣接ノードへセグメントを送信したら，再び隣接ノードへ BM を送信する

後は (2)～(4) を繰り返し行うことでデータ送受信を行う.

5.3 要求セグメント解析

セグメント要求時に自身の BM と隣接ノード全ての BM を解析し，なるべく隣接ノードが所持していないセグメントを優先して取得する．これによりネットワーク全体にファイルが拡がる効率が良くなると考えられる．

提案手法の要求セグメント解析の様子を図 14 に示す．

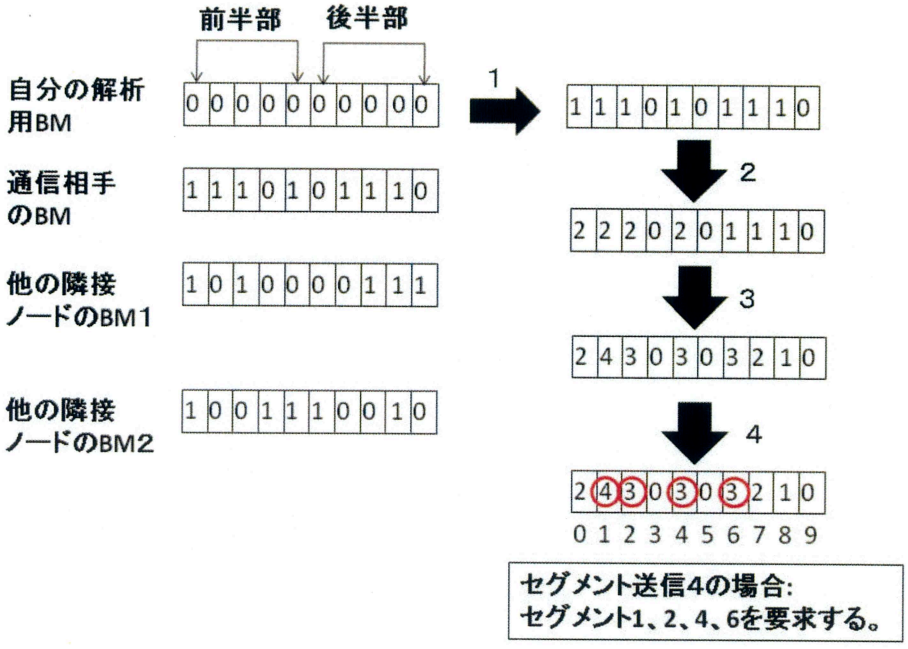


図 14　要求セグメント解析

図 14 の解析の流れを以下に載せる．

1. 通信相手の BM から自分が所有していないセグメントを調べる
2. 前半部を優先して取得するように重み付けを行う
3. 他の隣接ノードの BM キャッシュを調べ，通信相手が所有していて，他の隣接ノードが所有していないセグメントがあれば優先して取得するように重み付けをする
4. 最終的に最も重み付けの大きいセグメントから順に前から一定数を要求する

（4）で一定数を要求する理由は，セグメントは複数の隣接ノードから取得するので，1つの隣接ノードから全てのセグメントを取得しても効率は良くない．全ての隣接ノードから効率良くセグメントを取得するために，一定数セグメントを取得したら，再びセグメント要求解析を行うようになっている．

第 6 章
評価実験

本章では，提案手法の有効性を示すための実験について述べる．6.1 において，実験の概要について述べる．6.2 において，予備実験について述べる．6.3 において，シミュレーション実験について述べる．6.4 において，実機実験について述べる．6.5 において，実験の考察について述べる．

6.1 実験の概要

本研究では提案手法の有効性を示す為にシミュレーションによる比較実験，及び実機による比較実験を行う．それぞれの実験では，提案手法，CoolStreaming，Chainsaw を実装し実験を行った．

また今回提案するシステムの評価値として，上り帯域使用率 (Outbound Bandwidth Rate of Use，以下 OBRU)，平均 BM 送信回数 (Average Number of BM Transmission，以下 ANBT)，全体遅延時間 (Total Delay Time，以下 TDT) の 3 つを用いる．

OBRU とは，隣接ノードとの送受信時間に対する自身が持っているセグメントを隣接ノードへ送信する時間の割合である．OBRU は，式 (1) で求めれる．

$$OBRU = \frac{\sum S}{T} \times 100 \tag{1}$$

ここで，S は各セグメントの送信時間．T はデータ送受信時間である．

ANBT とは，ネットワーク全体での BM 送信回数を全ノード数で割った平均の数である．ANBT は，式 (2) で求められる．

$$ANBT = \frac{\sum B}{N} \tag{2}$$

ここで，B は各ノードの BM 送信回数．N は参加ノード数である．

TDT とは，データ転送開始時間から全ての参加ノードがデータを受信するまでの時間である．

シュミレーション実験で以上の評価値を求める．

6.2 予備実験

6.2.1 実験内容

既存手法との比較実験の前に予備実験として，5.3 で提案した要求セグメント解析の有効性を示すために，5.2 で提案した手法に対して要求セグメント解析の適応前，適応後での全体遅延時間の比較をシミュレーションで行った．実験内容については表 1 に示している．

表 1　実験内容

参加ノード数	100 or 1000 or 5000 or 10000
転送データサイズ	5(MB) or 100(MB)
隣接ノード数	3〜5
セグメントサイズ	16(KB)
BM 送信サイズ	128(bit)
上り帯域幅	5(Mbps)
下り帯域幅	20(Mbps)

また，データ駆動型 ALM ネットワークの構築について以下で述べる．

1. ノードがネットワークに参加した時，参加ノードが 4 ノード以下ならソースノードに接続．4 ノード以上ならソースノード以外の一般ノードへと接続する．
2. 自信が参加してから一定期間過ぎたら，他の参加ノードの情報を得る為にソースノードへとアクセスする．
3. ソースノードから得たリストを参照し，参加ノードからランダムに選んだノードを隣接ノードとする．もし選んだ参加ノードの隣接ノード数が最大で接続できなければ選んだノードを除外したリストから再びランダムに選んで，隣接ノードを探す．
4. 全てのノードが参加し，ネットワークを構築した後に転送データを送信する．

上記の実験環境で全体遅延時間の比較実験を行う．

6.2.2 実験結果

転送データサイズが 5MB のシミュレーション実験の結果を以下に示す.

実験結果 (TDT) を図 15 と表 2 に示す.

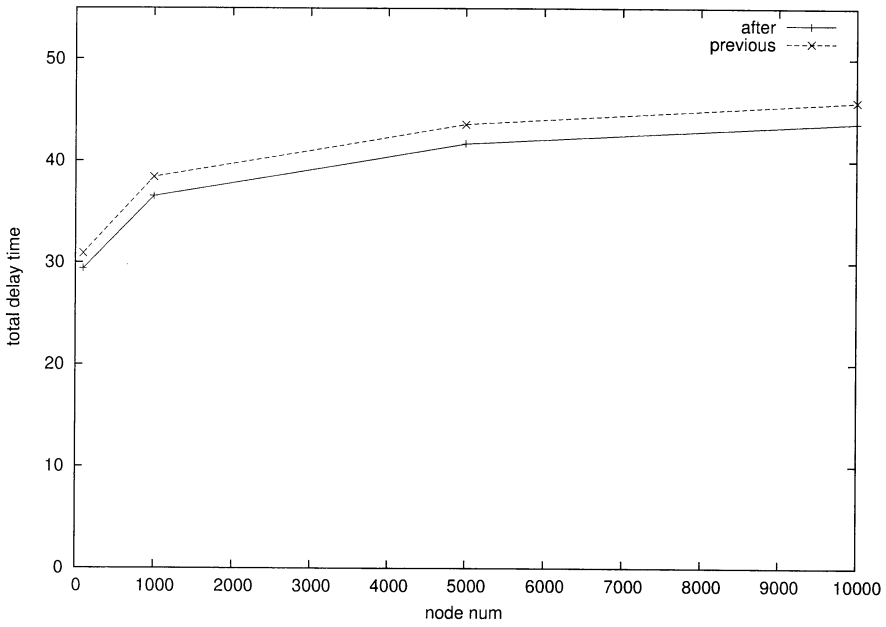


図 15　全体遅延

表 2　全体遅延

(s)	適応後 (after)	適応前 (before)
100	29.4	30.9
1000	36.5	38.4
5000	41.7	43.6
10000	43.7	45.8

転送データサイズが 100MB のシミュレーション実験の結果を以下に示す.
実験結果 (TDT) を図 16 と表 3 に示す.

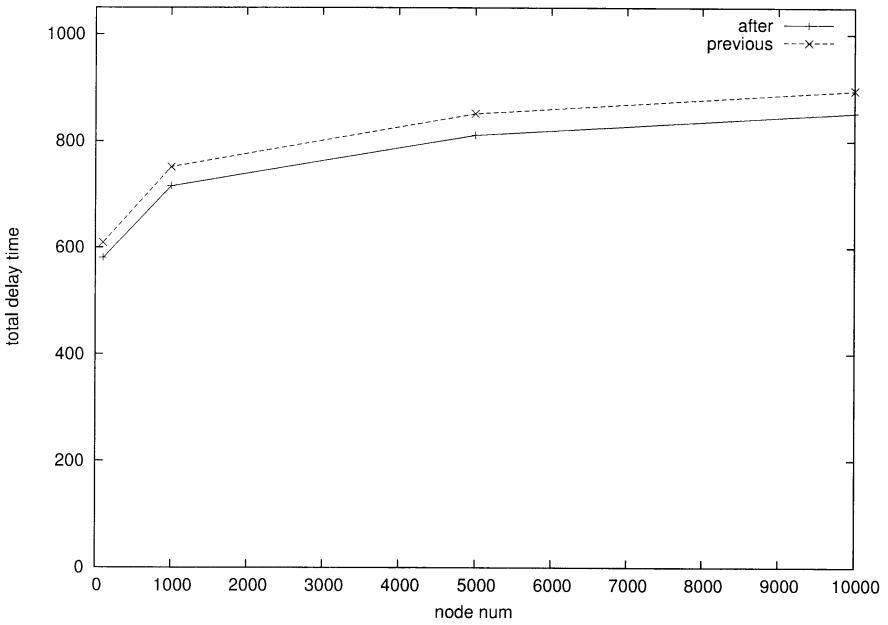


図 16　全体遅延

表 3　全体遅延

(s)	適応後 (after)	適応前 (before)
100	580.7	609.6
1000	715.9	751.6
5000	811.3	851.9
10000	852.7	895.3

6.2.3　考察

転送データサイズが 5MB, 100MB の両方とも要求セグメント解析適応前に比べて適応後の方が約 5 %ほど高速化された. この実験により要求セグメント解析を適応することで高速化されることを示せた. これより以下の実験での提案手法は全て要求セグメント解析を適応した手法とする.

6.3 シミュレーション実験

6.3.1 実験内容

シミュレーション実験では提案手法，CoolStreaming，Chainsaw をシミュレーション上で実行し比較を行った．実験内容については表 4 に示している．

表 4　実験内容

参加ノード数	100 or 1000 or 5000 or 10000
転送データサイズ	5(MB) or 100(MB)
隣接ノード数	3〜5
セグメントサイズ	16(KB)
BM 送信サイズ	128(bit)
上り帯域幅	5(Mbps)
下り帯域幅	20(Mbps)

また，データ駆動型 ALM ネットワークの構築について以下で述べる．

1. ノードがネットワークに参加した時，参加ノードが 4 ノード以下ならソースノードに接続．4 ノード以上ならソースノード以外の一般ノードへと接続する．
2. 自信が参加してから一定期間過ぎたら，他の参加ノードの情報を得る為にソースノードへとアクセスする．
3. ソースノードから得たリストを参照し，参加ノードからランダムに選んだノードを隣接ノードとする．もし選んだ参加ノードの隣接ノード数が最大で接続できなければ選んだノードを除外したリストから再びランダムに選んで，隣接ノードを探す．
4. 全てのノードが参加し，ネットワークを構築した後に転送データを送信する．

上記の実験環境で 6.1 で述べた評価方法を基に実験を行う．

6.3.2 実験結果 (転送データ 5MB)

転送データサイズが 5MB のシミュレーション実験の結果を以下に示す.

実験結果 (OBRU) を図 17 と表 5 に示す.

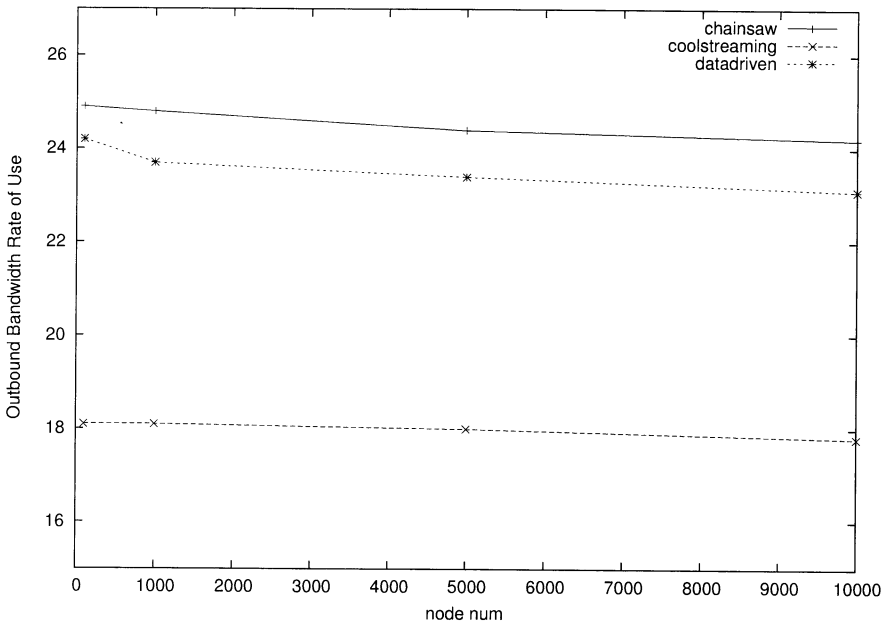


図 17　上り帯域使用率

表 5　上り帯域使用率

(%)	Chainsaw	CoolStreaming	提案手法 (datadriven)
100	24.9	18.1	24.2
1000	24.8	18.1	23.7
5000	24.4	18.0	23.4
10000	24.2	17.8	23.1

実験結果 (ANBT) を図 18 と表 6 に示す.

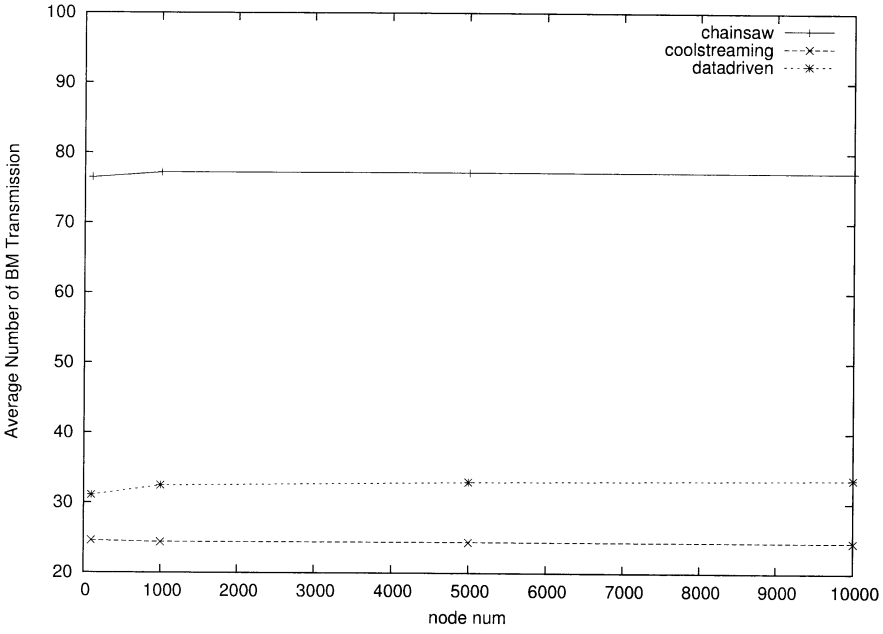


図 18　平均 BM 送信回数

表 6　平均 BM 送信回数

(回)	Chainsaw	CoolStreaming	提案手法 (datadriven)
100	76.5	24.6	31.1
1000	77.2	24.4	32.5
5000	77.2	24.4	33.0
10000	77.2	24.4	33.4

実験結果 (TDT) を図 19 と表 7 に示す.

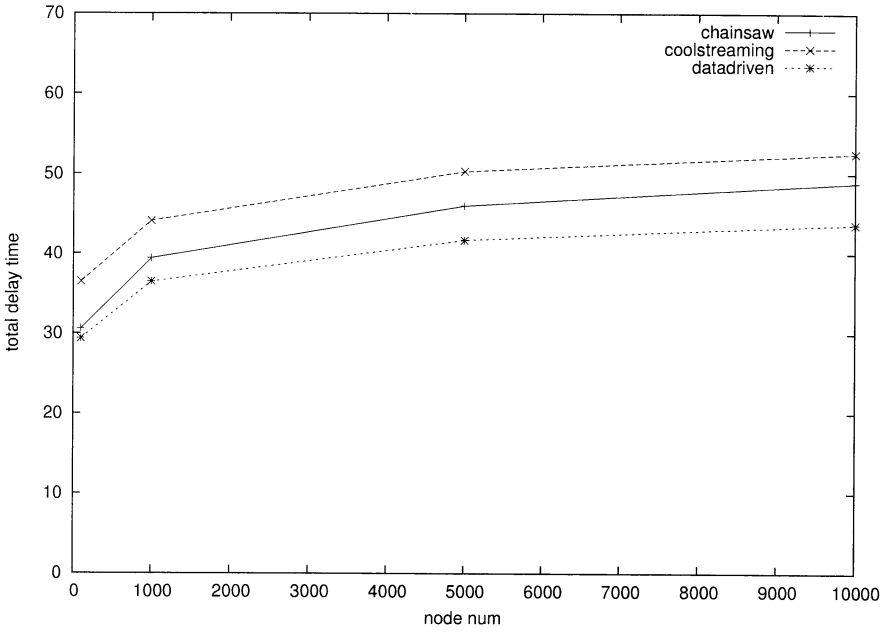


図 19　全体遅延

表 7　全体遅延

(s)	Chainsaw	CoolStreaming	提案手法 (datadriven)
100	30.6	36.5	29.4
1000	39.4	44.1	36.5
5000	46.0	50.3	41.7
10000	48.9	52.6	43.7

6.3.3 実験結果 (転送データ 100MB)

転送データサイズが 100MB のシミュレーション実験の結果を以下に示す.

実験結果 (OBRU) を図 20 と表 8 に示す.

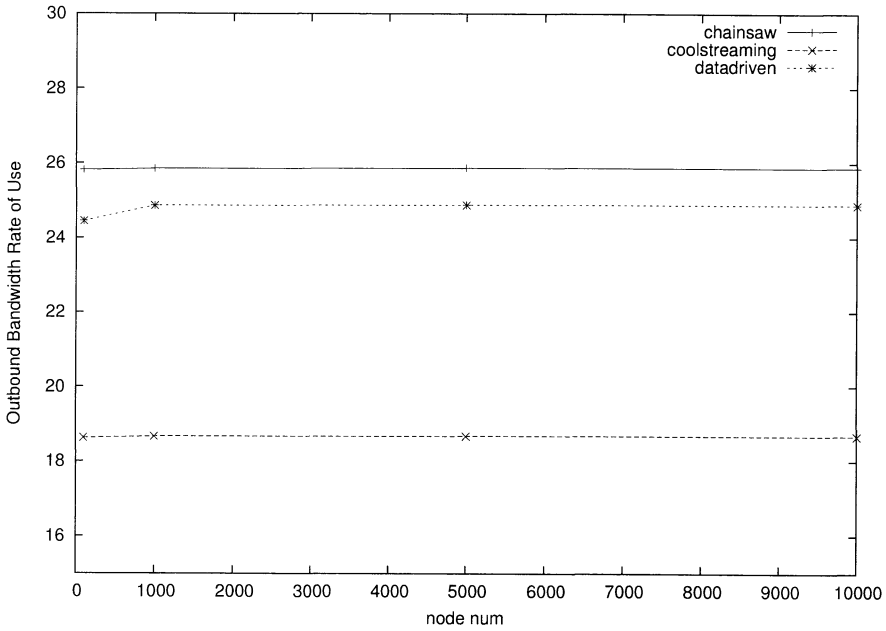


図 20　上り帯域使用率

表 8　上り帯域使用率

(%)	Chainsaw	CoolStreaming	提案手法 (datadriven)
100	25.8	18.6	24.5
1000	25.9	18.7	24.9
5000	25.9	18.7	24.9
10000	25.9	18.7	24.9

実験結果 (ANBT) を図 21 と表 9 に示す.

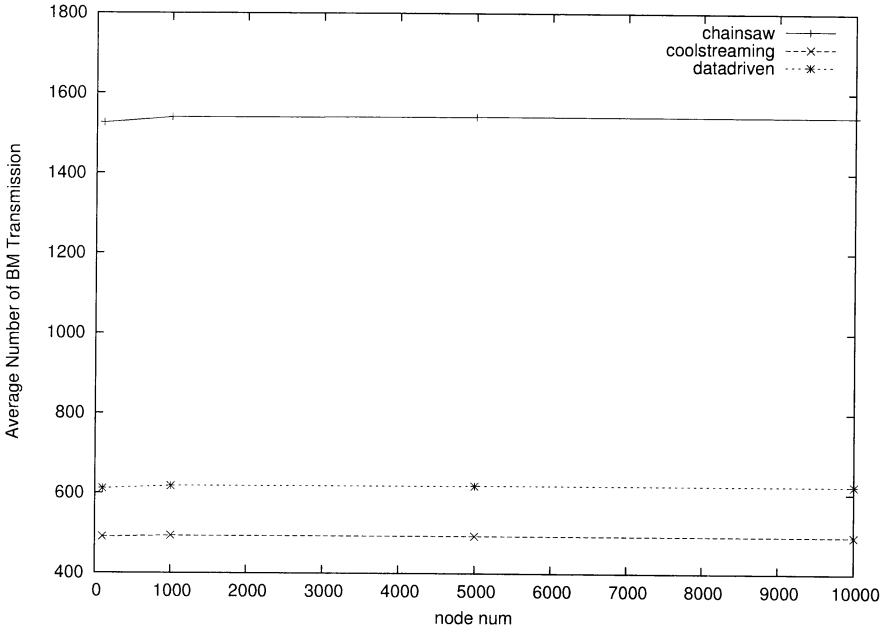


図 21　平均 BM 送信回数

表 9　平均 BM 送信回数

(回)	Chainsaw	CoolStreaming	提案手法 (datadriven)
100	1525.3	490.7	611.6
1000	1539.2	494.0	618.0
5000	1540.4	493.2	619.0
10000	1540.6	492.9	619.2

実験結果 (TDT) を図 22 と表 10 に示す.

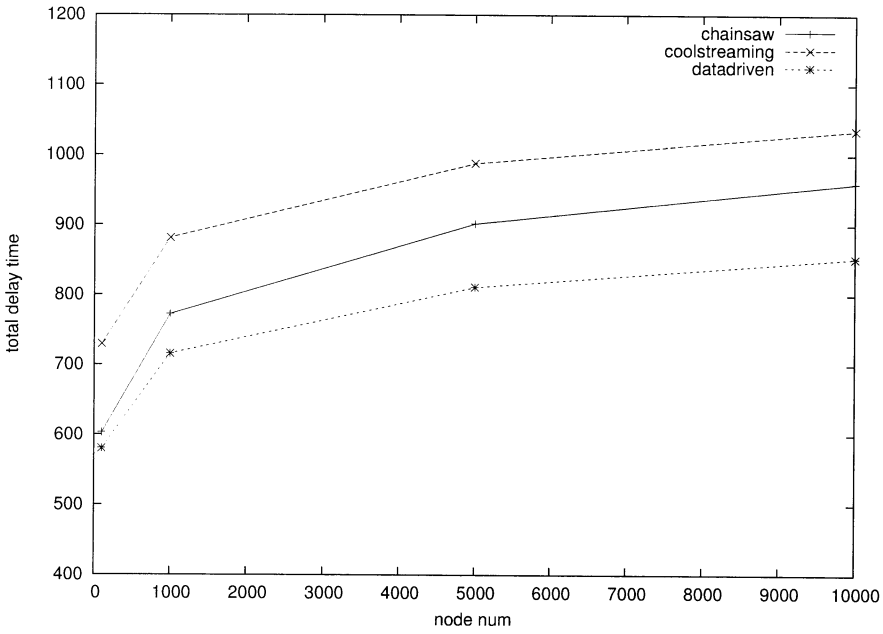


図 22　全体遅延

表 10　全体遅延

(s)	Chainsaw	CoolStreaming	提案手法 (datadriven)
100	603.1	729.7	580.7
1000	772.5	881.5	715.9
5000	901.7	988.2	811.3
10000	959.9	1035.0	852.7

6.4 実機実験

6.4.1 実験内容

実機実験として，インターネット環境を想定したシミュレーションシステムを用いて実際に提案手法の実験を行う．本実験でも提案手法，CoolStreaming，Chainsaw を実装し実機シミュレーションシステム上で比較を行った．実験内容については表 11 に示している．

表 11　実験内容

参加ノード数	64
転送データサイズ	5(MB)
隣接ノード数	3〜5
セグメントサイズ	16(KB)
BM 送信サイズ	128(bit)
上り帯域幅	1(Gbps)
下り帯域幅	1(Gbps)

また，データ駆動型 ALM ネットワークの構築について以下で述べる．

1. ノードがネットワークに参加した時，参加ノードが 4 ノード以下ならソースノードに接続．4 ノード以上ならソースノード以外の一般ノードへと接続する．
2. 自信が参加してから一定期間過ぎたら，他の参加ノードの情報を得る為にソースノードへとアクセスする．
3. ソースノードから得たリストを参照し，参加ノードからランダムに選んだノードを隣接ノードとする．もし選んだ参加ノードの隣接ノード数が最大で接続できなければ選んだノードを除外したリストから再びランダムに選んで，隣接ノードを探す．
4. 全てのノードが参加し，ネットワークを構築した後に転送データを送信する．

また，実験で使用したシミュレーションシステムは，4Core×2 の CPU を搭載したシミュレーションサーバ 8 台を用いて 1 台につき 8 ノード (各ノードには個別の NIC 有り) 動かし，計 6 4 ノードでデータ配信を行う．本システムを用いることにより，より実環境に近い実験が行えると考えられる．

上記の実験環境で 6.1 で述べた評価方法を基に実験を行う．

6.4.2 実験結果

実機実験の結果を以下に示す.

実験結果 (OBRU) を図 23 と表 12 に示す.

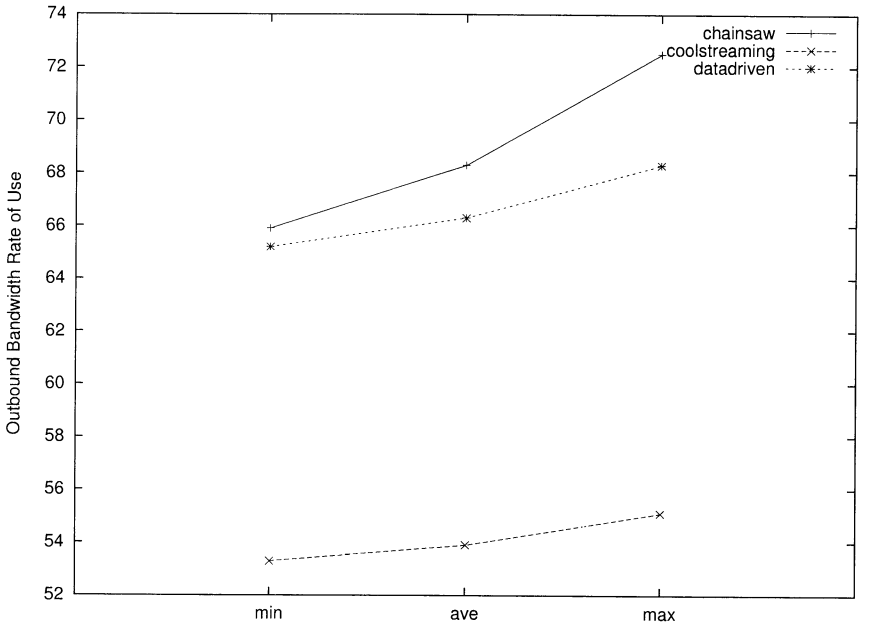


図 23　上り帯域使用率

表 12　上り帯域使用率

(%)	Chainsaw	CoolStreaming	提案手法 (datadriven)
min	65.9	53.3	65.2
ave	68.3	53.9	66.3
max	72.5	55.1	68.3

実験結果 (ANBT) を図 24 と表 13 に示す.

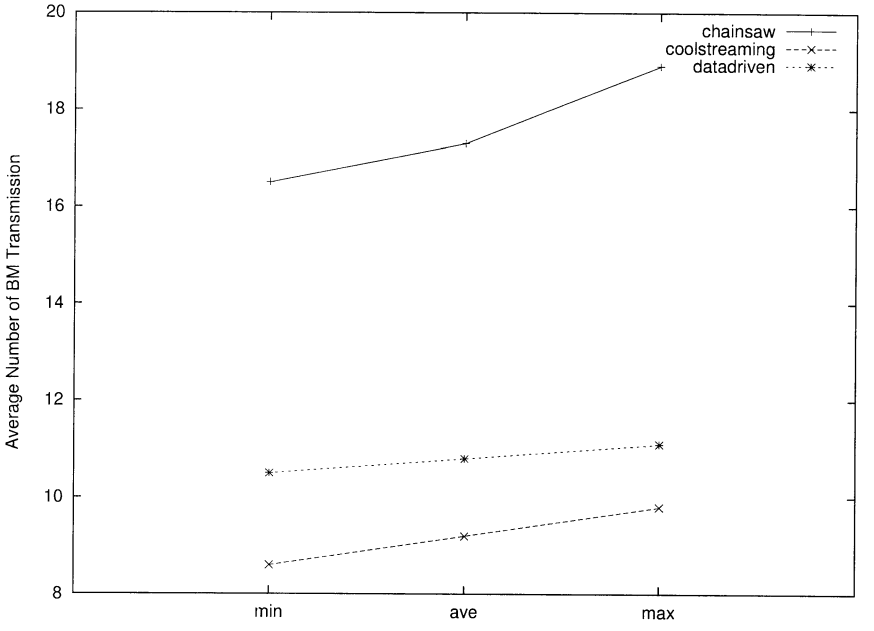


図 24　平均 BM 送信回数

表 13　平均 BM 送信回数

(%)	Chainsaw	CoolStreaming	提案手法 (datadriven)
min	16.5	8.6	10.5
ave	17.3	9.2	10.8
max	18.9	9.8	11.1

実験結果 (TDT) を図 25 と表 14 に示す.

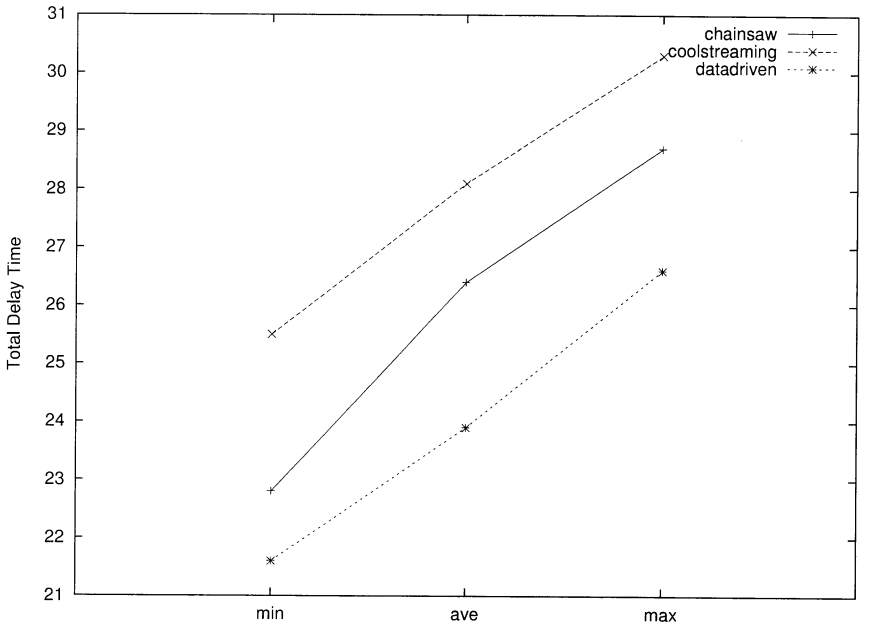


図 25　全体遅延

表 14　全体遅延

(%)	Chainsaw	CoolStreaming	提案手法 (datadriven)
min	22.8	25.5	21.6
ave	26.4	28.1	23.9
max	28.7	30.3	26.6

6.5 考察

まず，OBRU についてはChainsaw が最も良い結果となった．この理由としては，Chainsaw のデータ転送における制御パケットを新たなセグメントを取得するとすぐに隣接ノードへ送信するので，常に最新の BM からセグメントを要求できるなどがあげられる．このことより，効率良くデータの転送を行っていると言える．しかし，本提案手法も Chainsaw にはやや劣るがほぼ同じ数値が得られている．理由としては，隣接ノードへセグメントを送り終わった後に BM を送る事で，所有しているセグメントを効率良く送信しているからであると考えられる．

次に ANBT については，シミュレーション実験では提案手法，実機実験では CoolStreaming が最も良い結果となった．しかし，ANBT については CoolStreaming の BM 送信間隔によって結果は変わってしまう．今回の BM 送信間隔は 10ms と設定してある．提案手法は CoolStreaming よりも回数はやや多いが，Chainsaw に比べると十分に制御パケットを抑えられていると考えられる．

最後に TDT については，CoolStreaming より 2 割，Chainsaw よりは 1 割ほど提案手法の方が良い結果となった．理由としては，上り帯域を十分に活用でき，かつ制御パケット量をなるべく抑えることができたからだと考えられる．更に，セグメント要求時に提案した手法を取り入れた事も大きな要因だと考えられる．

おわりに

本研究では，データ駆動型 ALM での新たなデータ転送方式を提案し，シミュレータ・実機を用いてインターネット環境を想定した実装をして実験を行った．従来手法の CoolStreaming では，制御パケットである BM が一定時間経過しないと隣接ノードへ送信されないことから，ネットワークの上り帯域の使用をうまく活用できていない問題があった．また従来手法の Chainsaw では，セグメントを取得する毎に BM 更新の為の制御パケットを送信することから，BM 送受信量が大幅に多くなり，通信処理の負担が大きくなってしまう問題があった．それらの問題に対して，提案手法では新たに制御パケットの送信タイミングを設定し，要求セグメントの解析を効率良く行うことで，制御パケットを抑えつつ，上り帯域を効率良く使用して遅延を少なくすることを実現できた．

今後の課題としては，実機実験での大規模・大容量比較実験がある．しかし，情報工学科に導入された実機シミュレーションシステムでは大規模実験は難しい．よって PlanetLab[13] 等を用いた実機実験等があげられる．

謝辞

日ごろから多くの御指導を頂きました太田義勝教授，鈴木秀智准教授に深く感謝いたします．そして，日頃何かとお世話になりました落合美子事務員に感謝いたします．また，本論文作成にあたって特にお世話になりました太田義勝教授に深く感謝いたします．最後に，日頃から熱心に討論して頂いた研究室の諸氏に感謝いたします．

参考文献

[1] V.Padmanabhan, H. Wang, P. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking", Proceedings of the 12th international workshop on Network and operating systems support for digital audio and video, pp177-186ACM New York , NY, USA, 2002.

[2] Y. Chu, S. Rao, S. Sehan, and H. Zhang, "A case for end system multicast", IEEE Journal on Selected Areas in Communications, vol.20, no.8, pp.1456-1471, 2002.

[3] M. Castro, P. Druschel, A. Kermarrec, and A. Rowstron, "SCRIBE:A large-scale and decentralized application-level multicast infrastructure", IEEE Journal on Selected Areas in Communications Vol.20 No.8, Oct, 2002.

[4] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth Content Distribution in a Cooperative Environment",Proceedings of ACM SOSP'03, New York, USA, Oct.2003.

[5] A. J. Gancsh, A-M. Kermarrec, and L. Massoulie, "Peer-to-peer membership management for gossip-based protocols", IEEE Transactions on Computers, 52(2), Feb.2003.

[6] D. Kosti, A. Rodriguez, J. Albrecht, and A. Vahdat, "Bullet:High Bandwidth Data Dissemination Using an Overlay Mesh",SOSP, Proceedings of ACM SOSP'03, New York, USA, Oct.2003.

[7] X. Zhang, J. Liu, B. Li, and T-S. P. Yum, "CoolStreaming/DONet: A Data-Driven Overlay Network for Peer-to-Peer Live Media Streaming", in IEEE INFOCOM 2005, Miami, US, Mar.2005.

[8] V. Pai, K. Kumar, K. Tamilmani, V. Sambamurthy, and A. E. Mohr, "Chainsaw:Eliminating Trees from Overlay Multicast", in Proceedings of the 4th International Workshop on Peer-to-Peer Systems, Ithaca, NY, 2005.

[9] 池長慶彦, 首藤一幸, 村岡洋一, "実ネットワークに適応するオーバレイマルチキャスト放送基盤", 情報処理学会研究報告 2007-DSM-44, 2007.

[10] M. Zhang, C. Chen, Y. Xiong, Q. Yang, "Optimizing the Throughput of Data-Driven Based Streaming in Heterogeneous Overlay Network", Lecture Notes in Computer Science, ISSN 0302-9743, 2007.

[11] 越賀雅士, 太田義勝, 鈴木秀智:" データ駆動型 ALM によるデータ配信手法について", 情報

処理学会第 71 回全国大会 (3U-1), 2009.

[12] 越賀雅士, 太田義勝, 鈴木秀智:" データ駆動型 ALM の配信手法の高速化", 平成 21 年度 電気関係学会 東海支部連合大会 (O-354), 2009.

[13] PlanetLab Website: http://www.planet-lab.org/