平成19年度　　卒業研究論文

通信遅延削減のための遅延評価システムの作成と削減手法の評価

指導教官
齋藤 彰一 准教授

名古屋工業大学　情報工学科
平成16年度入学　16115069番

近藤 正基

目 次

第 1 章　はじめに **1**

第 2 章　オーバーレイネットワーク **3**

2.1　オーバーレイネットワークと P2P 3

2.1.1　背景 3

2.1.2　DHT 5

2.2　問題点 6

第 3 章　P2P における遅延評価 **8**

3.1　通信遅延の考慮 8

3.1.1　通信遅延の問題 8

3.1.2　新しい評価の必要性 9

3.2　関連研究 10

第 4 章　遅延評価の実装 **11**

4.1　Overlay Weaver 11

4.1.1　概要 11

4.1.2　ツールキットの構成 12

4.2　実装 13

4.2.1　反復/再帰探索 13

4.2.2　オーバーレイを構築するノード 14

4.2.3　遅延カウンタ 15

第 5 章　評価 **18**

5.1　Chord を使った経路制御方式 18

5.1.1　通信遅延を考慮した経路制御 18

5.2　実験環境 19

5.3　実験結果 19

5.4　考察 19

第 6 章　おわりに **20**

第1章

はじめに

近年、インスタントメッセンジャーなど多くの peer-to-peer 技術を用いたアプリケーションが使用されており、インターネット上でオーバーレイネットワークが構築されている。Ｐ２Ｐネットワークでは中央サーバが存在しないことで処理がお互いに分散できるため，個々のコンピューターの性能や回線の容量がそれほど要求されない。といった利点が存在するものの、目的のファイルやノードを高速に検索することが困難である。この問題の解決法として分散ハッシュ法 (Distributed Hashing) や分散ハッシュテーブル (Distributed Hash Table: DHT) と呼ばれる手法が注目されている。また P2P システムで良いパフォーマンスを得るためには、ルーティングアルゴリズムが重要である。そのため、Chord，Pastry，Tapestry,CAN など多くの研究が報告されており、これらのアルゴリズムは、DHT を基礎とするアルゴリズムと同じような手法を用いている。つまり、分散ハッシュテーブルを使用し、その上でオーバーレイネットワークを構築することで高速な検索が可能となる。

しかし、分散ハッシュテーブルを使用したオーバーレイネットワークは、実際の物理ネットワークの近さが分散ハッシュテーブルの近さとは無関係である。そのため、オーバーレイネットワーク上では無駄の無い検索ホップであっても、実際の物理ネットーワークでは、一度遠くのノードまで検索が進んだ後に、検索を開始したノード付近のノードに検索が戻って来る無駄が含まれる可能性がある。このような問題は、P2P ゲームや VoID といった計算機間の同期などが重要とされる時に大きな問題となっている。そのほか、匿名性を考慮しオーバーレイネットワーク上で最短経路を通らずに

ルーティングを行う場合通信の遅延がどれだけ起きるかといった問題は非常に重要である。

このようなオーバーレイネットワークを開発するうえで、開発者は実装、評価、デバッグといった手順を踏むことになる。しかし、実際にサーバを各所に分散配置し実験を行うのは困難であるため、通常、アルゴリズムの動作と有用性はシミュレーションで確認される。本研究ではそのオーバーレイ開発環境の一つである Overlay Weaver に遅延を評価するための機能を実装することで、さまざまなルーティングアルゴリズムを遅延の観点から評価することを可能とした。本論文では、遅延見積りの実装方法について述べ、実際に遅延を考慮にいれたルーティングアルゴリズムを実装し遅延評価を行った結果を紹介する。

以降、2 章では評価の対象となるオーバーレイネットワークと DHT の概要について述べ、3 章では関連研究を挙げる。続く 4 章では遅延評価の機能を実装する OverlayWeaver について述べ、実装方法を説明する。5 章では実際に遅延によってルーティングを制御するアルゴリズムを実装、シュミレートしその評価を行う。最後に 6 章で結論をまとめる。

第2章

オーバーレイネットワーク

本研究は、近年主体になりつつあるP2P技術でのDHTを用いたルーティングアルゴリズムの遅延評価を行っている。本章ではまずオーバーレイネットワークとP2P、DHTについて述べた後、その問題点を明確にする。

2.1 オーバーレイネットワークとP2P

2.1.1 背景

オーバーレイネットワークとは「既存のネットワークを抽象的に取り扱う」概念のことで、「IPアドレスの抽象化」ともいわれる。オーバーレイ・ネットワーク上のノードは、下位ネットワークのトポロジーを意識せずに通信することができる。

近年多くの人が参加するコミュニケーション手段としてインターネットが確たる地位をしめるようになった。特にBBS(電子掲示板)やインスタントメッセンジャーなどのようなコミュニケーションサービスとして広く利用されている。

このようなサービスは主にクライアントサーバ型で提供され、情報を一括で管理するサーバが中央に位置するのが一般的である。しかし近年の技術革新によってADSLや光ファイバを安価に利用できるようになったのを受け利用者へのサービスが広帯域かし、また利用者も増えていることからサーバへの転送量が大幅に増大し、そのためサービスの維持に大きなコストがかかるようになった。その一方で利用者の帯域幅が

増えたことからオーバーレイネットワーク上で様々なサービスを行うことが現実的となりサーバクライアントモデルと並んで P2P などのオーバーレイネットワークも選択肢の一つとなっている。

サーバクライアントモデルと P2P モデルの違いを以下に示す。

	サーバクライアントモデル	P2P モデル
長所	クライアントの処理が小さい データの更新が容易	処理が分散される 隠匿制が高い
短所	サーバの処理能力が要される 個人情報が 1 点に集中	管理、監視が困難

上記の通り P2P の特徴の一つはサーバーにデータを置かないことにある。ユーザーの興味があるデータを一元的なサーバーに置くのではなく、各々のユーザーが自分の PC(ピア) に持つ。しかし、このように分散してデータを持ってしまうと、自分が必要としているデータがどのノードにあるのかを探索する仕組みが必要になってくる。P2P システムは探索の仕組みによって以下のように分類できる。

(a) ハイブリッド P2P

クライアント・サーバーと P2P の複合という意味で、システム内にはデータの所在を知っているサーバー（インデックスサーバー）が存在し、各ノードはインデックスサーバーにファイルの所在を問い合わせ（クライアント・サーバー方式）実際のデータ取得はそのファイルを持つノードから直接行う（P2P)。

特徴として、名前解決サーバは単に相手の IP アドレスを通知するだけなのでサーバ、ネットワーク負荷が比較的小さい。また、ユーザ間で直接データをやり取りするので、サーバはユーザ間でやり取りしているデータを見ることができない。

(b) ピュア P2P

データ探索のためのサーバーを持たず、データの探索もピア同士で行う。特徴としてサーバを構築せずに通信を行えるが、短所は相手の IP アドレスを事前に知る必要がある。

(c) 動的クラスター型ハイブリッド P2P

探索のための特定のサーバーは持たないが、探索用に別の協調ネットワークが存在するシステム。探索用のノード群は要求元のノードに対して、お互いが協調して探索結果を返すため、要求元から見ればクラスターのように動作する。また探索用ノード群の構成員は固定的ではなく、特定の条件を満たした一般のノードが選ばれることもある。

本論は、検索が困難と思われる pure-P2P をとりあげ、その検索手法である DHT とその遅延の評価について論じていく。

2.1.2 DHT

pure-P2P のような、中央にサーバが存在せずそれぞれのノードが独立してコンテンツを保持している場合、特定のデータ項目を見付けることは困難である。この検索問題を解決する技術として開発されたアルゴリズムの一つとして分散ハッシュテーブル (DHT) を用いた検索手段がある。これはデータ項目は SHA-1 などのハッシュ関数用いて一意な ID との組とされ、各ノードはそれぞれが受け持つ範囲の ID を一意に決め対応するデータ項目を保持する方法である。データ項目を検索する際はもちろん DHT を実装するアルゴリズムはどのノードが与えられた ID に対応するデータを保持しているかを判断できなければならない。この問題を解くために、個々のノードは少数の他のノードの情報 (IP アドレス) を維持しておき、オーバーレイネットワークを構築している。

chord

Chord では個々のノードは ID 空間において自分から空間の大きさ 1/2、1/4、1/8、… と自分から 2 の階乗分の 1 だけ離れたノードの IP アドレスを、Skiplist の様な構造で持つ指標 (finger table) を保持している。また、他に自ノードより ID 空間で隣になるいくつかノードの IP アドレスのリスト (Successor list) を保持している。

与えられた Key に対して検索を行う時、要求元ノードは自分の知っているノードの

中からKeyを越えないもっとも近いノードに対して問い合わせを行う。問い合わせを受けたノードは、同じ様に知っているノードからKeyを越えないもっとも近いノードに問い合わせを行う。もしくは問い合わせを行うよう要求元ノードに返答する。

指標の２の階乗構造は少なくともKeyへの残りID空間距離を半分にするノードへ常に問い合わせを行うことを保証し、O(logN)メッセージで問い合わせを解決することを可能にしている。

Chordの設計で強調されている点はその強固制と正しさであり、頻繁に参加と脱退が起こる状況下でも証明可能な性質をもつ。Chordのアルゴリズムは、ノードの故障、参加の元でもID空間で検索Keyの前にくるノード一つでも情報を持っていれば問い合わせが徐々に検索Keyに向かって前進していくことを保証する。また定期的にfingertableとsuccessorlistを修正し、適切な情報を保持するよう補正される。検索に失敗する唯一の状況は、検索Keyの直後にくるノード全てが補正される前に同時に故障もしくは離脱した場合である。

2.2 問題点

前項の通り従来のオーバーレイネットワークではオーバーレイネットワーク上の距離と物理ネットワークでの経路上の距離は完全に独立したものとして扱われている。よって、オーバーレイネットワーク上では最適なルーティングでも、物理ネットワーク上では遠回りになる状況が発生する。

実際に本論で扱うP2P技術も、DHTを使った手法によってオーバーレイネットワーク上では無駄の無い検索ホップであっても物理ネットワーク上では、一度遠くのネットワークに存在するノードまで検索が進んだ後に、検索を開始したノードの近くのノードに検索が戻ってくる無駄が含まれている状況が存在している。

また、オーバーレイネットワークの普及により匿名性を考慮した通信路なども研究されている。これは主にデータ転送を第三者を経由して行うことで匿名性を確保する研究が多くを占めるが、このときエンドホスト間の遅延は全体のパフォーマンスを決める大きな要因となる。

このような問題に対処するため近年、通信遅延を考慮し、より効率的なオーバーレイネットワークを構築する手法が提案されている。

第3章

P2Pにおける遅延評価

前章でオーバーレイネットワークにおける問題点を示唆した。本章では本研究のキーとなる遅延の問題に焦点をあてP2Pにおける遅延の評価、また関連研究について説明する。

3.1 通信遅延の考慮

3.1.1 通信遅延の問題

2章で挙げた通り、一般的にオーバーレイネットワーク上の距離と物理ネットワーク上の距離は違うものである。これはP2P技術においても例外でなく前章で紹介した「Chord」をとって説明すると以下の図のように、一度遠くのノードに検索した後近くのノードに戻るという無駄が含まれる可能性がある。

また各ノードがオーバーレイネットワーク上に流れるデータを処理する際、処理機能の低いノードに経由するデータが多いとオーバーレイネットワーク全体のパフォーマンスが低下するおおきな要因となる。オーバレイネットワークは互いにリンク，ルータなどの物理網資源を共有，競合するため，あるオーバレイネットワークの制御が物理ネットワークを介して間接的に他のオーバレイネットワークに影響を与えてしまうという問題が指摘されている。

また近年P2Pネットワークの重要性が高まる中、オーバーレイネットワーク上の通信の匿名性に対する要求も高まってきている。匿名通信路を実現するためには第三者

のノードを経由して通信を行うことが考えられるが、第三者を経由することによって起きる遅延は重要な指標の一つと考えることが出来る。

このように、P2P 技術の需要が高まるにつれ様々な通信方式が開発されると考えられ、オーバーレイネットワークの問題点の一つである通信遅延の問題はこれから更に重要になってくると考えられる。

そして更に拡大していくであろう P2P ネットワークに対するこのような問題の解決の重要性は高まってきているといえる。

3.1.2 新しい評価の必要性

このような問題の解決策として、経路制御のためのオーバーレイネットワークを導入するものや、遅延時間を考慮にいれてネットワークを形成するもの、オーバーレイネットワークを動的に最適化するものなどがあげられている。

近年までは、オーバーレイネットワークという概念そのものが物理ネットワークとは完全に独立した物として扱われていた。つまり、開発支援環境上でオーバーレイネットワークの評価を行う際に通信遅延を考慮に入れる必要はなく、ルーティングのホップ数やメッセージ数を評価の基準とするだけで十分とされていた。しかし、このように現在ではエンドホスト間の通信遅延を考慮に入れたオーバーレイネットワークの構築手法や、ルーティングアルゴリズムが提案されている。エンドホスト間の通信遅延を物理的ネットワークの経路上の距離とみなし、オーバーレイネットワーク上でのルーティングやマルチキャストに反映させることで効率の良いサービスを提供することが出来る。このように P2P 技術が大きく展開を見せる中、通信遅延を視野に入れたルーティングやオーバーレイネットワークを提案、構築する際、これまでのようにホップ数やメッセージ数による評価だけでは不十分になりつつある。さらに大規模なオーバーレイネットワークが構築され、参加するピア数もさらに増加すると予想される中、通信遅延といった観点からの評価指標も重要になってくると考えられる。

3.2　関連研究

通信遅延を評価指標に入れシュミレート出来るものに ns-2 や p2psim などがある。

(1)ns-2

ns-2 はネットワークアプリケーションの評価のために幅広く利用されているネットワークシュミレータである。物理ネットワークまでシュミレートしており、輻輳やパケットロスといった現象を再現することが出来る。しかし計算量が大きく、本論で扱う OverlayWeaver にくらべシュミレーションで再現可能なノード数に限りがある。

(2)p2psim

p2psim は peer-to-peer プロトコルのシュミレータである。一台で 3000 台のノードの挙動を現実的な実行時間で処理可能である。また p2psim では通信を行うノード間の RTT び一覧表をもとに遅延をシュミレートすることが出来る。本研究と同じく遅延は固定的である。

上記の研究は本研究と同じくオーバーレイネットワークを構築開発する際に遅延を評価指標の一つとしてシュミレートを行えるものである。しかし、どちらの研究もルーティングアルゴリズム構築の際、アルゴリズム実装自身が通信や RPC といった低レベルの処理を行う為、アルゴリズム実装のコード量が多く開発者の負担となる。

第4章

遅延評価の実装

P2P における遅延評価の実装を行った。本章では遅延評価を追加するにあたって大きく参考にした OverlayWeaver と遅延評価の追加実装について説明する。

4.1 Overlay Weaver

4.1.1 概要

OverlayWeaver はオーバーレイ構築ツールキットの一つである。

P2P といった技術は現代においてまだ新しい技術であり、近年サーバークライアントモデルと並ぶ選択肢の一つになりつつある。

しかし、これらのアルゴリズムを設計・評価する際、可能であれば大規模なノード数

そんな中、OverlayWeaver は Dabek らが提案する、あらゆる Structured オーバーレイに共通するルーティング処理の抽象化に従い、ルーティング層とその上の DHT サービス、マルチキャストサービスを分離している。またルーティング層から各種のルーティングアルゴリズムに共通する処理をくくり出し、それぞれがもつ固有の処理との間のプログラミングインターフェースを見出している。それゆえ、OverlayWeaver でシュミレーションもしくは実験を行う際多くの選択肢から目的に合ったルーティングアルゴリズムを選択できる。そのほか、さまざまなアルゴリズムをたかだか数百ステップで実装することが可能であり、どのアルゴリズム実装を使うか実行時に決めることができるので、設計者は新規、既存アルゴリズム間の比較を大規模かつ公正に行うこ

とができる。

4.1.2　ツールキットの構成

OverlayWeaver はオーバーレイネットワークを構築する他、以下のツールを有している。

- 分散環境エミュレータ
- シナリオ生成器
- メッセージカウンタ
- メッセージ可視化ツール

分散環境エミュレータ

OverlayWeaver は分散環境エミュレータを提供しており、数千ノードを計算機 1 台でエミュレートできる。ルーティングアルゴリズムを設計、実装する際、同一のシナリオを異なるアルゴリズムで実行することでアルゴリズム間の公正な比較が可能となっている。

メッセージカウンタ

オーバーレイネットワークを構築する各ノードのメッセージサービスには、メッセージ送受信をネットワーク越しに、逐一報告させることが出来る。このメッセージカウンタは、この報告を受信してメッセージ数を数えることが出来る。このメッセージ数を見ることで、ルーティングアルゴリズムの評価を行うことが出来る。

4.2 実装

4.2.1 反復/再帰探索

ルーティングのアルゴリズムや経路が同一であっても経路上の各ノードが相互にどのように通信を行うかということというパターンはいくつか考えられる。また、この通信パターンは遅延の観点から見ると、全体の遅延を決める非常に大きな要因となる。現在この通信パターンとして、反復探索と再帰探索が知られている。

反復探索では探索の要求元が経路上の各ノードに対して自ら問い合わせを行い、次ホップを決めていく。もう一方の再帰探索では、インターネットのルーティング同様に、経路上の各ノードが探索要求を転送していく。それぞれの動作を図に示す。

再帰探索には、要求元への返答の返しかたによって 2 通りの通信パターンがある。通信回数を少なくするためには、到達先から要求元へ直接返答を返せばよいが、経路に沿って返答を返すことにも意義があり、どちらの方法もとられている。

これらそれぞれの通信パターンは、効率性や安全性といった点でそれぞれトレードオフの関係をもっているが、探索遅延としての関係を表に示す。

	反復	再帰 (fast)	再帰 (slow)
ホップ数	2n or 2(n+1)	n+1	2n
メッセージ数	同上	2n+1	同上

探索様式における遅延

反復: $(2n-1)\cdot t_{req} + t_{rep}$
or$(2n+1)\cdot t_{req} + t_{rep}$

再帰 (fast): $n\cdot t_{req} + t_{rep}$

再帰 (slow): $n\cdot t_{req} + n\cdot t_{rep}$

OverlayWeaver は反復探索と到達先から要求元へ返す再帰探索の実装を提供しており、本論の遅延評価もこの２つの通信パターンに対して実装を行う。

4.2.2 オーバーレイを構築するノード

オーバーレイを構築するノードは他ノードと通信するときメッセージカウンタに逐一どんなメッセージを送ったかを伝える。このとき送られるメッセージには以下の様な情報が付与されている。

- カウンタへのメッセージの送信元
- カウンタへのメッセージの種類
- 他ノードと通信したメッセージの内容
  - 送信元
  - 送信先
  - メッセージの種類

このメッセージに対して次の情報を付け加えた。

- ルーティング開始ノード
- ルーティングを判断するフラグ (ルーティングの最中か否か)
- ルーティング終了フラグ

ルーティング処理の最中というのは探索を開始してから解を得るまでの間であり、理想的な環境の場合図でしめしたメッセージ間となる。

この情報を加えてカウンタに送ることにより、カウンタ側ではルーティングにかかる時間を計測することが可能になり、また全体の遅延も計測することが出来るようになる。

また、ルーティング処理とアルゴリズムを分離している特性上、他のアルゴリズムへの差し替えが可能となっている。

4.2.3　遅延カウンタ

遅延のカウントは 4.1.2 で示したメッセージカウンタを拡張する形で実装する。

遅延カウンタでは、メッセージのカウントの他に前項でしめした情報を用いて処理を行う。処理の流れを以下に示す。

1. カウンタが立ち上げられた時に指定の遅延ファイルを読み込む
2. 前項で示したオーバーレイを構築するノードから送られてくる情報と、読み込んだ遅延情報ファイルをもとに、以下の観点から遅延の計測を行う
   - どのノードとどのノードが通信したか。
   - ある ID(Key) を探すのにかかった全体的な時間
3. 「delay」コマンドが打ち込まれた時、計測していた遅延をファイルに出力する。

まずカウンタを起動したときに指定したファイルを読み込む (1)。この時のファイルは CSV 形式で書かれた、検証を行うネットワークの遅延情報である。このファイルをもとに遅延の計測を行っていく。

つぎにオーバーレイを構築するノードから送られてくる情報に対して処理を行う (2)。このとき遅延カウンタはオーバーレイネットワークに参加しているノード数の 2 乗 (n × n) のテーブルを 2 つ持つ。このテーブルは送信元 (1〜n) と送信先 (1〜n) の情報である。カウンタは送られてくる情報に対して、送信元と送信先を割り出し読み込んだ遅延情報とマッチングし、2 点の通信間の遅延時間を取り出す。その後取り出した遅延時間をカウンタが持っている一方のテーブルにインクリメントしていく。

また、ルーティングを行っている最中の情報を得たとき、カウンタはルーティングごとに取り出した遅延情報を一時保持していく。探索が終了した情報を得たとき、最終的な送信元と送信先を割り出し対応する (2) とは別のテーブルに一時保持していた

遅延情報を足しあわせる (3)。この２つのテーブルの情報を見ることで、開発者はそのとき行ったシナリオ上でルーティングにかかる大まかな時間を知ることができる。また「delay」コマンドを入力することでそのときのテーブルが持っている情報をファイルで出力させるように実装を行った。

出力例

第5章

評価

4章で示した遅延評価システムを実際に評価を行う。この際、DHTを使ったルーティングアルゴリズムであるChordに対して遅延を考慮した経路制御を実装し評価を行った。本章ではChordと通信遅延を考慮したアルゴリズム、そして実験結果の検証を行う。

5.1 Chordを使った経路制御方式

2.1.2で示したChordのアルゴリズムに遅延を考慮して経路制御を行うよう変更した。そのルーティングアルゴリズムとChordとを本ツールを使って評価を行う。

5.1.1 通信遅延を考慮した経路制御

2.1.2でChordの経路付けを説明したが、これは2.2で示した通り物理ネットワーク上の距離とオーバーレイネットワーク上の距離が独立しているという問題を孕んでいる。

そこで、次に問い合わせるノードを次ホップ候補のうちから一番遅延の低いものに対して問い合わせを行うよう実装した。

このとき次ホップを決める要因となる他ノードとの遅延はまえもって用意されているものとし、その遅延情報をもとに次ホップを決める。

5.2 実験環境

5.3 実験結果

5.4 考察

第6章

おわりに

謝辞

本研究のために多大な御尽力を頂き、日頃から熱心な御指導を賜った名古屋工業大学の斎藤彰一准教授に深く感謝致します。

また本研究に対し熱心に御検討、御協力を頂きました、松尾啓志教授に深く感謝を致します。

加えて、本研究の際に多くの助言、協力を頂いた斎藤研究室ならびに松尾・津邑研究室の皆様に深く感謝致します。