2Q-7

サーバレス型インスタントメッセンジャーの開発

池谷 亮平　志田 晃一郎　横山 孝典
武蔵工業大学

1 はじめに

近年，ブロードバンドの普及によって常時接続率が高くなり個人のインターネットの利用が大幅に増加した．コミュニケーションツールとして多くの人に利用されているものにインスタントメッセンジャー（以下 IM）がある．

IM は多人数チャット，ボイスチャット，ファイル送信の機能を持つ．代表的なものに MSN Messenger[1] がある．IM には管理やメンテナンスが容易なサーバクライアント型が使われている．

しかしサーバクライアント型の IM を利用者側から見た問題点として以下の２つがある．

1. サーバが停止してしまうとサービスが停止してしまい，IM を使うことが出来ない．
2. サーバには多くの個人情報が集約している．情報が漏えいした場合ユーザに被害が及ぶ．

また管理者側から見た問題として次の点がある．

1. サーバ設置のためのコストがかかる．利用者が多くなれば大規模なシステムになりコストは多くかかる．

そこで本研究の目的は，上記 3 つの問題点を回避するひとつの方法として，サーバを使用しない IM を開発することである．IM では通信したいユーザが持つ固有の ID をサーバに送り，その IP アドレスを取得し通信するが，本研究ではサーバではなく分散ハッシュテーブルを用いることで実現する．今回，IM としての機能は最小限の 1 対 1 のチャットのみを実装し，分散ハッシュテーブルによって IM に参加しているユーザの名前からその IP アドレスを取得する機能を持つ IM を作成した．

2　分散ハッシュテーブル

分散ハッシュテーブルは，少ない通信回数で固有の名前からそれに対応する端末の IP アドレスの検索を高速に行うアルゴリズムである [2]．各端末はハッシュ関数で変換された固有のハッシュ ID を持ち，この ID によって各端末を識別する．さらに各端末はハッシュテーブルという他の端末の ID と端末の IP アドレスが格納されたテーブルを保持する．ハッシュ ID で構成されたハッシュ空間上で固有の名前を検索すると，それに対応するハッシュテーブルから目的の名前に対応する端末の IP アドレスを取得できる．

Chord[3] は分散ハッシュテーブルの代表的なアルゴリズムである．Chord のハッシュ空間は 2^m（m は自然数）の円状の 1 次元空間である．Chord はハッシュテーブルの他に検索用の前任者テーブル，後継者テーブル，スキップリスト形式テーブルを保持する．4bit の Chord のハッシュ空間の例を図 1 に示す．

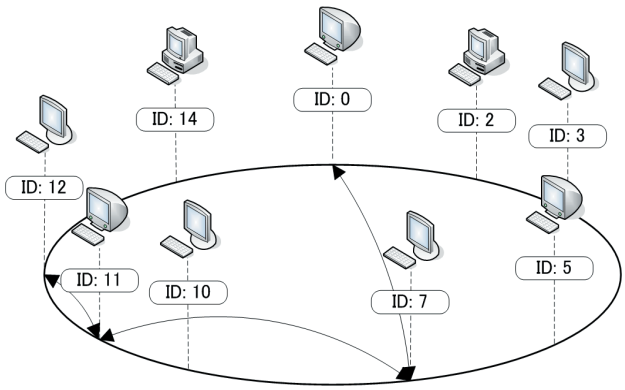

図 1: 4bit Chord のハッシュ空間

図 1 にある矢印は端末 0 から端末 12 が管理しているハッシュテーブルの端末情報を検索した様子を示している．端末 0 のハッシュテーブルには端末 12 の情報はない．そこで端末 0 はテーブル中端末 12 に最も近い端末 7 に端末 12 の検索を依頼する．端末 7 も同様に動作し端末 11 に検索を依頼する．端末 11 のテーブルには端末 12 の情報があるので，端末 12 より目的の端末の情報を端末 0 は受け取り検索は完了する．

本研究ではこの Chord の検索機能を利用し IM を実現する．

3　Chord を使用する IM の変更点

ユーザが実際に操作するのは IM である．IM への参加，離脱，端末の検索，は Chord を用いて行なわれる．作成した IM の各機能の動きを図 2 に示した．

Development of a Serverless-type InstantMessenger
Ryohei Ikeya, Koichiro Shida, Takanori Yokoyama
Musashi Institude of Technology

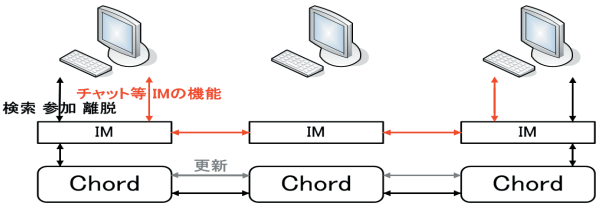

図 2: IM と Chord の動き

サーバレス型の IM の実現のために以下の機能の追加変更が必要である.

Chord では一度ハッシュ空間に参加すればその端末の情報はハッシュテーブルに残る. しかし急な端末の離脱によってその情報が失われる可能性がある. IM は通信したい相手の現在の状態が確認できることが利点なので, IM によって定期的に端末の情報がハッシュ空間上のハッシュテーブルに登録されているかを調べ, もし情報が消えてしまっている場合は再度情報を登録する機能を追加し端末情報の消失を解決する.

既存の IM と同じく端末や名前, IP アドレスを変更してもハッシュ空間上に接続できるが, 既存の IM のように友達を登録してあるデータをサーバから取得できない点も問題である. 異なる端末から接続はできても自身の友達リストを利用するには友達リストのファイルのコピーが必要である. そこで IM によって既に友達登録されている端末に対して, 自分の保有する最新の友達リストのコピーを保持させる. 他の端末から接続した場合には, オンラインの友達一人を検索し問い掛けることで自分の友達データのコピーを受け取る機能を追加しこの問題を解決する.

またある 2 台の端末が接続したとき, 2 台の IP からのハッシュ値が同一になってしまう場合がある. この場合は最初に接続されている端末 (端末 A) がハッシュテーブルの管理を行う. 後から接続された端末 (端末 B) は控えとして存在することになる. 端末 A が先に離脱する場合には端末 B が管理を引き継ぐ. 端末 B は他の端末に対して更新するよう支持し交代は完了する.

4 実装

Chord のプログラムに関しては, ハッシュ関数には SHA-1 を使用する. SHA-1 のハッシュ値は 160bit なのでハッシュ空間は 160bit となる. 友人の登録リストは名前と IP がテキストファイルとして保存される. 次回からはファイルからデータを読み込み直接通信することができる. 後継者テーブルと前任者テーブルは, 端末の急な離脱などで変化してしまうために, 新しく端末が登録された場合, 離脱した場合, そして時間で定期的に更新を行なう. 今回は 30 分で更新するよう設定したが, 30 分という時間は今後テストを重ねることによって変更になる可能性がある. スキップリスト形式テーブルは正確性はあまり求められないため, 更新は上記のテーブルよりゆとりを持った 1 時間ごとに行うよう設定した. Chord のプログラムはインスタントメッセンジャーのプログラムが開始されると同時に開始され参加プロセスを行う. IM のプログラムに関しては, 1 対 1 のチャットの機能を作成した. GUI は実装していないのですべてコマンドからの操作となる.

サーバレスに動作する IM のプロトタイプとして, ローカルネットワークで使用できるチャットと Chord のプログラムを Java によって作成し動作を確認した.

サーバを使用せずに名前検索により IP アドレスを取得しそれによってチャットができることを確認した.

5 おわりに

本研究ではローカルネットワークで端末を名前によって検索し, 接続されている目的の端末の IP を取得し, チャットするプログラムを作成した.

しかし, 今回の名前検索では他人も自分と同じ名前を使用できる点が問題である.

今後の予定として, 名前の重複問題の解決, プログラムのインターネットの対応化, 端末の情報更新, 友達リストのコピー, ハッシュ値の重複の解決が挙げられる. IM としての機能の追加として, 多人数チャット, ファイル送信, 通信及びファイルの暗号化, GUI 化が挙げられる.

参考文献

[1] http://messenger.msn.co.jp/

[2] H. Balakrishnan, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, Looking up data in P2P systems, Communications of the ACM, 46(2):43－48, 2003.

[3] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, Chord: A scalable peer-to-peer lookup service for internet applications, In Proc. of the 2001 Conf. on Applications, technologies, architectures, and protocols for computer communications, 149 － 160. ACM Press, 2001.

[4] Kenneth L. Calver, Micheal J. Conahoo, TCP/IP ソケットプログラミング Java 編, オーム社, 2003.