3T-5

インターネット上の距離に基づいた階層的クラスタを活用した分散ハッシュテーブル

林田紳也, 上田達也, 安倍広多, 石橋勇人, 松浦敏雄
大阪市立大学大学院創造都市研究科

1 はじめに

近年, P2P 技術において盛んに研究されているテーマの一つとして分散ハッシュテーブル (DHT) がある. しかし, 既存の DHT では, 検索するデータがネットワーク的に近いノードにある場合でも, オーバレイネットワーク上での検索パスが遠くのノードを経由する場合が少なくない.

我々は, インターネット上のノード間距離に応じてノードを図 1 のように階層的にクラスタリングする手法を提案している [1]. 本研究では, この階層的クラスタを利用することにより, 可能な限りインターネット上で近いノード間で通信を行う DHT を提案し, 上記の問題点を改善する.

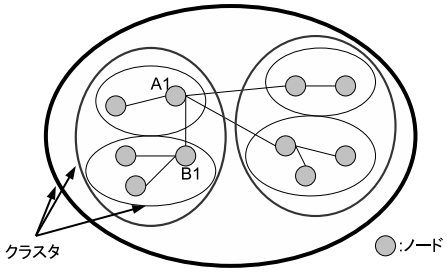

図 1 階層的クラスタ

2 提案するアルゴリズム

2.1 基本概念

本研究では, Chord[2] を提案手法の基礎とし, Chord と同様, キー k に対してデータ v を対応付けるサービスを提供する.

提案手法では, 各クラスタに 1 つの Chord リングを形成し, そのクラスタに属するノード集合をマッピングする. 図 1 の例では図 2 のようなリングを構成する. ノード A1, B1 を例にとると, 最下層のクラスタでは, A1, B1 それぞれが属するリングを別個に形成するが, 一つ上位の階層では, A1, B1 両方が属するリングを形成する. 最上位クラスタでは, 全てのノードが属するリングを形成することになる. このように, 各ノードは複数のリングに属する.

また, 各ノードは Chord と同様, 固有の識別子 (ID) を持っており, その値はどの階層のリングにおいても同一である.

各ノードは属するすべてのリング上で自身の前後のノード (predecessor, successor) へのポインタを持つ[*1]. また, ノード ID の増加方向を時計回りとする.

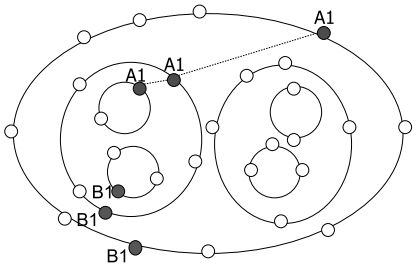

図 2 モデル図

2.2 アルゴリズム

2.2.1 定義

ノード N が属するリングを内側から順に $R_1(N)$, $R_2(N), \ldots, R_d(N)$ と表記する (d は階層的クラスタの深さである).

ノード N は, キー k が (N の predeccessor の ID, N の ID] の区間に入るデータを保持する. $R_i(N)$ でキー k のデータを保持するノード (ルートノード) を $R_i(N).Root(k)$ と表記する.

2.2.2 データの登録 (put)

本手法では, ノード N がキー k のデータ v を登録する場合, $R_1(N).Root(k), \ldots, R_d(N).Root(k)$ の各ノードに保存する. まず $R_1(N)$ において Chord と

A Distance-aware Distributed Hash Table using Hierarchical Clustering of Internet Nodes
Shinya Hayashida, Tatsuya Ueda, Kota Abe, Hayato Ishibashi, Toshio Matsuura
Graduate School of Creative Cities, Osaka City Univ.

[*1] ただし, 最下層を除く各クラスタでは Chord の Finger Table に相当するポインタは保持しない

同様の手法で $R_1(N).Root(k)$ を検索し, そのノードに (k, v) を保存する. 次に, $R_1(N).Root(k)$ を起点に $R_2(N)$ を k の値とノード ID を比較しながら反時計回りに進み, $R_2(N).Root(k)$ を発見し, そのノードに (k, v) を保存する (図 3). 同様な処理を $R_d(N)$ まで繰り返す.

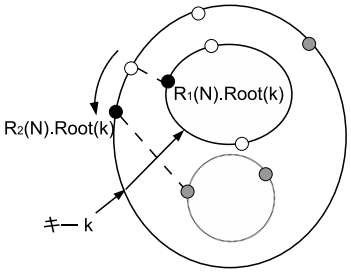

図 3　リング間のルートノードの関係

2.2.3　データの取得 (get)

ノード N がキー k のデータ v を取得する場合, まず $R_1(N)$ において Chord と同様な手法で $R_1(N).Root(k)$ を検索し v を保持しているか問い合わせる.

保持していない場合, データの登録時と同様の方法で $R_2(N).Root(k)$ を検索し, v を保持しているか問い合わせる. v を保持しているノードが見つかるまで階層を上りながら, この処理を繰り返す. $R_d(N)$ まで検索してもデータが見つからない場合, 検索は終了する.

$R_i(N)$ でデータが見つかった場合, $R_1(N).Root(k)$ から $R_{i-1}(N).Root(k)$ までのノードに検索結果をキャッシュしておくことで, 以後の検索を高速化することができる.

目的のデータがクラスタ C 内に存在する場合, そのデータを検索するためのトラフィックはクラスタ C 内に限定され, それより外の (遠い) クラスタのノードは経由しない.

2.2.4　ノードの参加 (join)

ノード N が DHT に参加する場合, すでに DHT に参加しており, 最下層においてノード N と同一のクラスタに属するノード M を参照し, Chord と同様な手法で $R_1(N)$ へと参加する. 次に $R_2(N)$ において, $R_1(N)$ における N の successor ノードを起点に, 反時計回りに $R_1(N)$ での predeccessor ノードを超えない範囲で, ノード ID を比較しながら順次検索し, N のノード ID を挟む直近の 2 つのノードの間に N を挿入する.

同様な処理を R_d まで行い, $R_1(N), R_2(N), \ldots, R_d(N)$ 全てのリングに参加した時点で, ノードの参加は完了する.

2.2.5　堅牢化

一般的に, P2P システムではノードの離脱等に備えてポインタに冗長性をもたせ, 必要に応じて修復する機能が必要である. 本手法では, Chord と同様な手法でこれを行う.

3　考察

全ノード数 n, 階層の深さ d(定数), クラスタ当りのサブクラスタ数を k(定数) とし, ノードは一様に分布していると仮定する. データの取得の際, R_1 で経由するノード数は平均 $\log(\frac{n}{k^{d-1}})$ であり, 一段上の階層でルートノードを検索する際に経由するノード数は平均 $k/2$ であるため, R_d まで検索を行った場合に経由するノード数は平均 $\log(\frac{n}{k^{d-1}}) + \frac{k}{2}(d-1)$ となる.

上式より本手法の検索に要するコストは $O(\log n)$ であり Chord と同等であるが, Chord では近くのデータを検索するために, 遠くのノードを経由することがあるのに対し, 本手法ではそのようなことは起こらない.

4　おわりに

本稿では, インターネット上の距離に基づいて構成された階層型のクラスタを利用した DHT を提案した.

今後の課題として, 実験による提案手法の有効性の検証, 本手法を活用した様々なアプリケーションの設計/実装等が挙げられる.

参考文献

[1] 上田達也, 安倍広多, 石橋勇人, 松浦敏雄. P2P 手法によるインターネットノードの階層的クラスタリング, 情報処理学会論文誌 Vol. 47, No. 4, pp. 1063–1076, 2006.

[2] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, Hari Balakrishnan, Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications. IEEE/ACM Transactions on Networking, Volume 11, Issue 1, pp. 17–32, 2003.