4S-03

構造化オーバレイネットワークにおけるノード状態を考慮した経路表構成手法

白石 裕輝 †
岐阜大学 工学部 電気電子・情報工学科

安田 真 ‡
岐阜工業高等専門学校 電気情報工学科

1 はじめに

オーバレイネットワーク (overlay network, OLN) は IP ネットワークなどのアンダーレイネットワークの上位に構築される論理的なネットワークである．分散インターネットアプリケーションの耐障害性・スケーラビリティを向上させるために不可欠な技術となっている．

近年，OLN 上に実現する分散ハッシュ表 (distributed hash table, DHT) に関する研究が盛んに行われてきた．DHT を実現するアルゴリズムの代表的なものに Chord [1] がある．これを柔軟な経路表 (flexible routing tables) という設計手法によって再設計した FRT-Chord [2] も提案されている．

分散インターネットアプリケーションではノードの通信帯域や処理能力，信頼性などを考慮して適切な役割分担を行うべきである．本研究では FRT-Chord をノード状態を考慮して経路表を構築するように拡張した．

2 FRT-Chord

FRT-Chord は Chord を柔軟な経路表に基づいて再設計したアルゴリズムである．

ノードとオブジェクトにはハッシュ関数によって ID が与えられ，consistent hashing に基づいて同一の ID 空間 $P = \{x \mid 0 \leq x < 2^m, x \in \mathbb{N}\}$ 上に配置される．

ID 間の距離は次のように定義される (定義 2.1) .

定義 2.1 (ID 間の距離 $\mathrm{d}(x_1, x_2)$)

$$\mathrm{d}(x_1, x_2) = x_2 - x_1 \mod 2^m \quad (x_1, x_2 \in P) \tag{1}$$

各ノードは経路表 $E = \{e_i\}$ $(i = 0, \ldots, |E| - 1)$ を持つ．経路表エントリ e_i は ID 空間 P 上のノード ID と IP アドレスの対である．経路表には sticky entry と呼ばれるエントリの集合 $\{e_0, \ldots, e_{k-1}, e_{|E|-1}\} \subset E$ $(k < |E|)$ が存在する．Sticky entry は Chord における successor list や predecessor と同様に到達性を保証する役割を果たす．以降では，エントリのノード ID を e_i と略記する．

ノード s から目標 ID t へのメッセージはエントリ $s' = \arg\min_{e_i \in E} \mathrm{d}(e_i, t)$ への転送を繰り返すことで到達する．

FRT-Chord は次のエントリ学習操作とエントリ厳選操作を繰り返すことで，ノード数 N について経路長 $O(\log N)$ を実現する経路表を構築する．

2.1 エントリ学習操作

エントリ学習操作では，ノード s が他のノードと通信するたびに新しいエントリ e_{new} を経路表に追加する (式 (2))．この時，各エントリがノード s からの距離 $\mathrm{d}(s, e_i)$ の昇順に並ぶように追加を行う．

$$E := E \cup \{e_{\mathrm{new}}\} \tag{2}$$

2.2 エントリ厳選操作

エントリ学習操作で経路表サイズ $|E|$ が最大エントリ数 L を越えようとした時，エントリ厳選操作を行う．

ノード s の経路表の各エントリの正規化間隔を次のように定義する (定義 2.2).

定義 2.2 (正規化間隔 S_i^E)

$$S_i^E = S^E(e_i) = \log \frac{\mathrm{d}(s, e_{i+1})}{\mathrm{d}(s, e_i)} \tag{3}$$

経路表 E から sticky entries を除いた $C = E \setminus \{e_0, \ldots, e_{k-1}, e_{|E|-1}\}$ $(k < |E|)$ を考える．C からエントリ $e_{\mathrm{removal}} = \arg\min_{e_i \in C} S^E(e_{i-1}) + S^E(e_i)$ を選び，経路表から削除することで $|E| \leq L$ を保つ．

Routing Table Construction Method Considering Node States for Structured Overlays

† Yuki Shiraishi, Dept. of Electrical, Electronics and Computer Engineering, Faculty of Engineering, Gifu University

‡ Makoto Yasuda, Dept. of Electrical and Computer Engineering, National Institute of Technology, Gifu College

3 提案手法

FRT-Chord のエントリ厳選操作を，経路長 $O(\log n)$ を維持しながらノード状態を考慮するように拡張する．

エントリ e_i のノード状態を表すパラメータを p_i $(i = 0, \ldots, |E| - 1)$ とし，エントリ e_i の p_i による順序関係 $\leq_{\mathrm{p}}$ を次のように定義する (式 (4)) .

$$e_1 \leq_{\mathrm{p}} e_2 \overset{\mathrm{def}}{\Leftrightarrow} p_1 \leq p_2 \tag{4}$$

エントリ学習操作のエントリ e_{new} がエントリ厳選操作で削除対象となったエントリ e_{removal} と等しい時，これを e_{j} とおく．$e_{\mathrm{j}-1}, e_{\mathrm{j}+1} \in C$ の時，e_{j} との正規化間隔を調べ，エントリ e_{r} を次のように決定する (式 (5)).

$$e_{\mathrm{r}} = \underset{e \in \{e_{\mathrm{j}-1}, e_{\mathrm{j}+1}\}}{\arg\min} \left| \log \frac{\mathrm{d}(s, e_{\mathrm{j}})}{\mathrm{d}(s, e)} \right| \tag{5}$$

この e_{r} と e_{j} を $\leq_{\mathrm{p}}$ によって比較し (式 (6)),

$$e'_{\mathrm{removal}} = \min_{\leq_{\mathrm{p}}} \{e_{\mathrm{r}}, e_{\mathrm{j}}\} \tag{6}$$

決定したエントリ e'_{removal} を経路表から削除する．

4 評価

ノード状態を表すパラメータ p_i に経路表の最大エントリ数 L を用いたアルゴリズムを Overlay Weaver [3] 上に実装し，次のシミュレーション実験を行った．

4.1 エントリ厳選操作の妥当性

$L = 160$ のノード 100 個 (ホスト名 1 〜 100), $L = 20$ のノード 900 個 (ホスト名 101 〜 1000) によってノード数 $N = 10^3$ の提案手法のネットワーク A を構成した．A に対してランダムに選んだノードからランダムな ID への探索を 10^6 回行い，すべてのノードの経路表に含まれるエントリ情報をヒストグラムにした (図 1).

図 1 より，最大エントリ数の大きいノードのエントリが多数経路表に維持されていることが確認できる．従って，拡張したエントリ厳選操作は妥当であると言える．

4.2 経路長の $O(\log N)$ 性

A について新たに $L = 80$ のノード n を参加させ，このノードからランダムな探索を 25 回，100 回行った時，n が持つ経路表のエントリ分布を調べた (図 2).

図 2 より，探索回数の増加に伴ってエントリ分布が経路長 $O(\log N)$ を実現する最良経路表に近づいていることがわかる．従って，経路長の $O(\log N)$ 性が言える．

4.3 フォワーディング効率

$L = 160$ のノード 1,000 個，$L = 20$ のノード 9,000 個によって $N = 10^4$ の提案手法のネットワーク B と

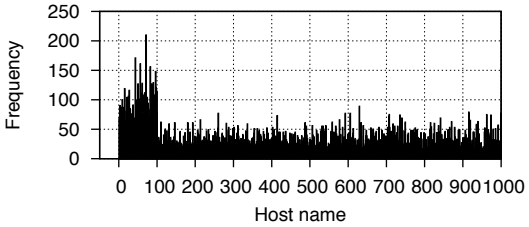

図 1 経路表エントリのヒストグラム

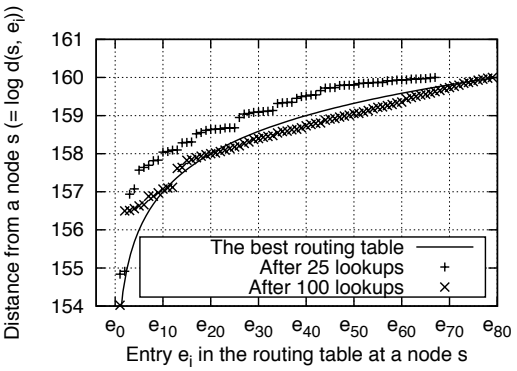

図 2 経路表エントリの分布

FRT-Chord のネットワーク C を構成し，ランダムな探索を 10^7 回行った．その後，B と C に対してランダムな探索を 50 回行った時の平均経路長を調べた．

C と比較して B では平均経路長が 21% 削減されており，転送が高効率化されていることが確認できた．

5 まとめ

FRT-Chord をもとに，経路長 $O(\log N)$ を維持しつつノード状態を考慮して経路表を構成するように拡張したアルゴリズムを提案した．最大エントリ数を考慮したアルゴリズムのシミュレーション実験では，最大エントリ数の大きいノードを経路表に維持できていることが確認できた．これによって，経路短縮率の低いノードへの転送を低減し，経路長を削減することができた．

参考文献

[1] I. Stoica, R. Morris, et al., “Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications,” *ACM SIGCOMM ’01*, pp. 149–160, 2001.

[2] H. Nagao, K. Shudo, “Flexible Routing Tables: Designing Routing Algorithms for Overlays Based on a Total Order on a Routing Table Set,” *IEEE P2P ’11*, pp.72–81, 2011.

[3] K. Shudo, Y. Tanaka, et al., “Overlay Weaver: An Overlay Construction Toolkit,” *Comput. Commun.*, pp.402–412, 2008.