仮想化DAY最新テクノロジーセッション

分散ハッシュテーブル (DHT)

InternetWeek 2009.11.26　14:30 - 15:50

株式会社ライブドア　伊勢　幸一

livedoor®

Agenda

1. DHT Summary
2. 代表的なアルゴリズム
3. Chordアルゴリズム
4. 性能
5. 実装

ちょっと自己紹介

1996年 「有限会社オン・ザ・エッジ」設立
2000年 東証マザーズに株式会社として上場
2002年 ライブドアから営業権を譲受
2003年 「エッジ株式会社」に社名変更
2004年 「株式会社ライブドア」に社名変更
・
・
2007年 「ライブドアホールディングス」に社名変更
(新) 株式会社ライブドアを設立
2008年 情報環境技術研究室を新設

ちょっと自己紹介

2008年4月　ネットワーク事業部内に
通信技術研究準備室として設立

2008年10月　情報環境技術研究室が新設

研究テーマ

次世代ネットワークアーキテクチュア

P2P配信技術

オーバーレイネットワーク技術

仮想化技術

論文発表　電子情報通信学会IA研究会

「クラウドシステムのための分散URI方式の評価」

DHT技術によるブログストレージの分散化と検索の実証実験

分散ハッシュテーブル

- P2Pオーバーレイネットワーク
- Key-Valueペア分散探索技術
- IDにConsistent-Hashを利用
- スケールアウトしやすい

DHT Summary(ざっくりとしたイメージ)

- ノードとデータを同じ空間にマッピング
- Key-Valueを近接するノードにストア

Key = ‘Hello’

Value = ‘World’

Hash(‘Hello’) → 0xcafebeef

Node = ‘n1.livedoor.jp’

Hash(‘n1.livedoor.jp’) → 0xcafebeff

Put(‘Hello’,’World’) → n1.livedoor.jp

‘World’ ← Get(‘Hello’) ← n1.livedoor.jp

代表的なアルゴリズム

ハッシュ空間の定義と探索方法による違い

- CAN　(2001) N次トーラス
- Tapestry, Pastry (2001)　Plaxton
- Kademlia (2002) 二分木
- Chord (2001)　環状
- Koorde　(2004) Chord改
- Broose　(2004) Koorde+Kademlia改
- Skip Graph　(2003) 厳密に言うとDHTではない

この他にもいっぱい！

CAN Content Addressable Network

論文 A Scalable Content Addressable Network (2001 UCB & ATT)

N次元トーラス空間にノードIDとコンテンツIDをマッピング

探索ホップ数= $O(dN^{1/d})$

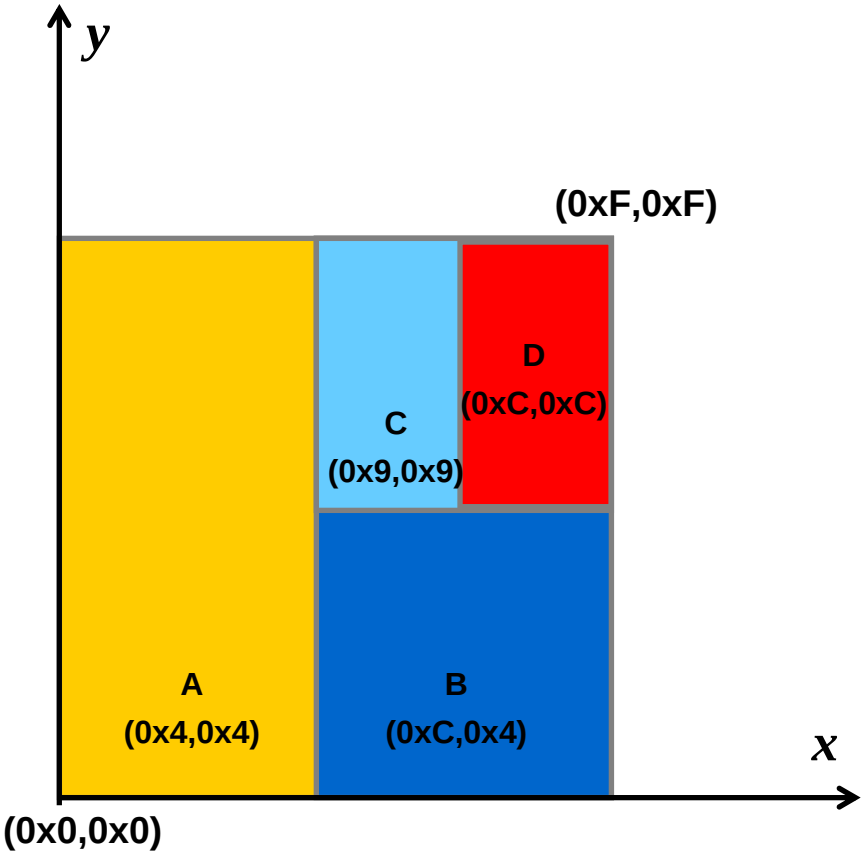
X = Hash1('Hello') = 0xE

Y = Hash2('Hello') = 0xC

Put('Hello','World') → ノードD

(0xE,0xC)のコンテンツをAから探索

A→C→D→(0xE,0xC)→'World'



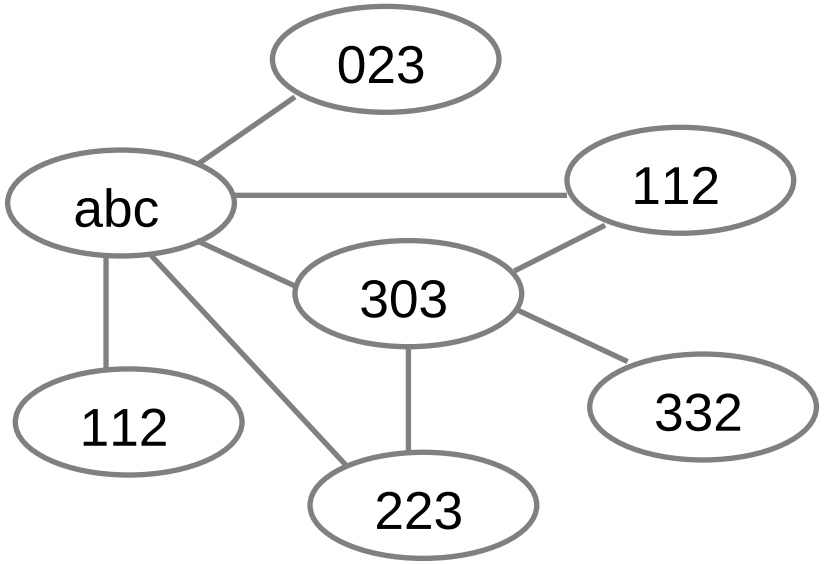
Pastry PlaxtonアルゴリズムによるDHT

論文　Scalable distributed object location and routing

for large-scale peer-to-peer systems (2001　MS & Rice)

Plaxtonアルゴリズム

IDは基数(b)と桁数(n)により構成 (例 3桁4進数)



ノードabcの経路表

0-23	1-12	2-23	3-03
a0-1	a1-0	a2-1	a3-2
ab0	ab1	ab2	ab3

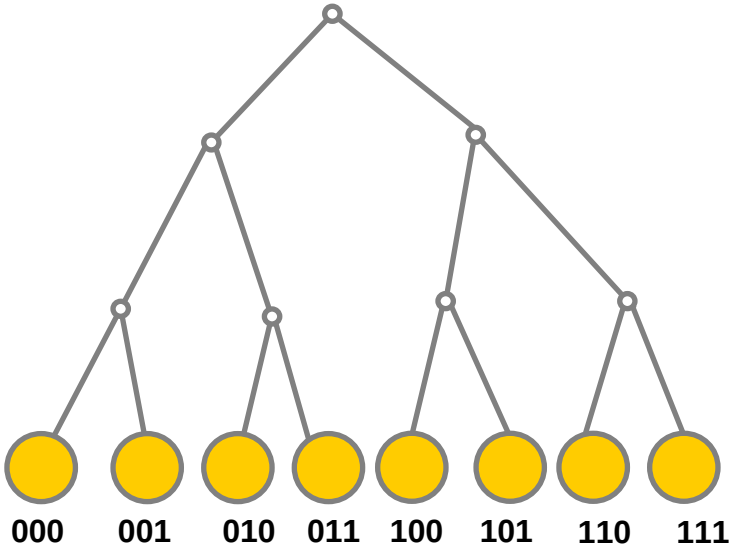
Pastryアルゴリズム　$O(b \log N)$

Plaxtonテーブルとリーフセット、ネイバーフッドセットの組み合わせ

Kademlia(カデムリア)

論文 A Peer-to-peer information System
Based on the XOR Metric (2002 NYU)

二分木構造のID空間
XORを空間の距離として定義



距離 = XOR(ノードID、Key)
ノードA = 001
ノードB = 010
Key = 100
XOR(001, 100) = 010 = 2
XOR(010, 100) = 001 = 1 → ノードBにValueをストア
プレフィックス一致長によるルーティングテーブル

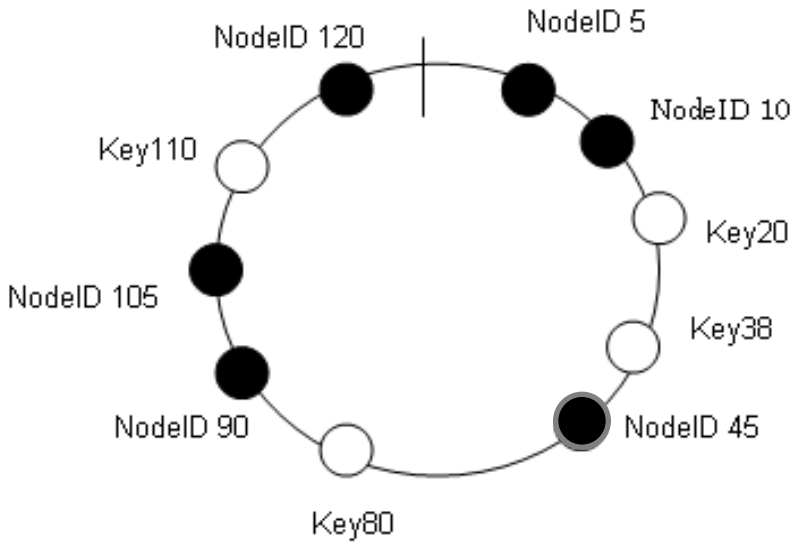
Chord

論文　A Scalable Peer-to-Peer Lookup Service for Internet Applications (2001 MIT)

160bit SHA-1によるハッシュ化

1次元環状空間 にノードIDをマッピング

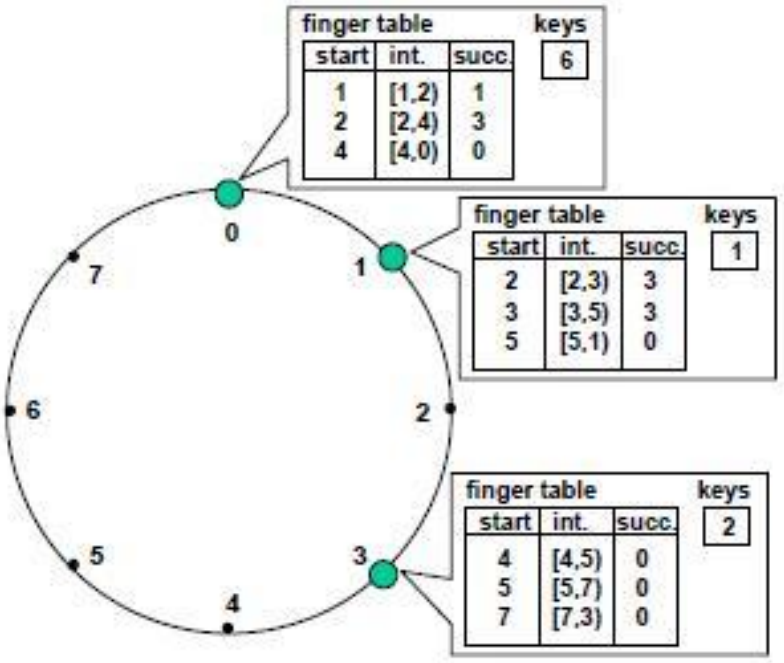
スキップリストによる探索(Finger Table) $O(\log N)$



Chord

Finger Table （mビット空間の場合、m個のエントリ）

1. finger[i].start = $(n+2^{i-1}) \bmod 2^m$
2. Interval (範囲) = finger[i].start ≦ interval ＜ finger[i+1].start
3. successor (id) = interval内の一番近いノードID

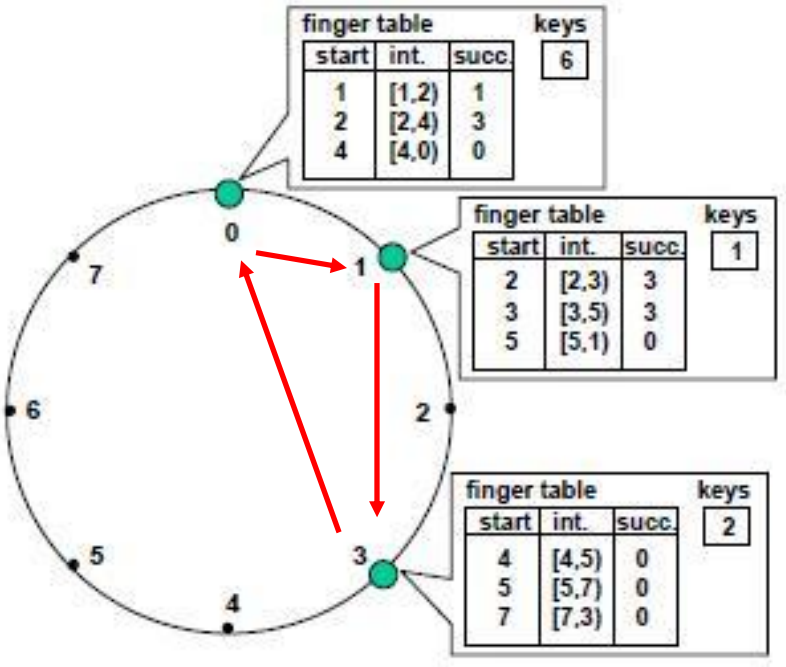


3ビット空間での例（原論文より抜粋）

Chord

探索「ノード3がキー1を探す」

1. ノード3のFinger Tableから[7,3)のsuccessorを見つける
2. ノード0のFinger Tableから[1,2)のsuccessorを知りノード3に返す
3. ノード3はノード1にキー1をリクエストする



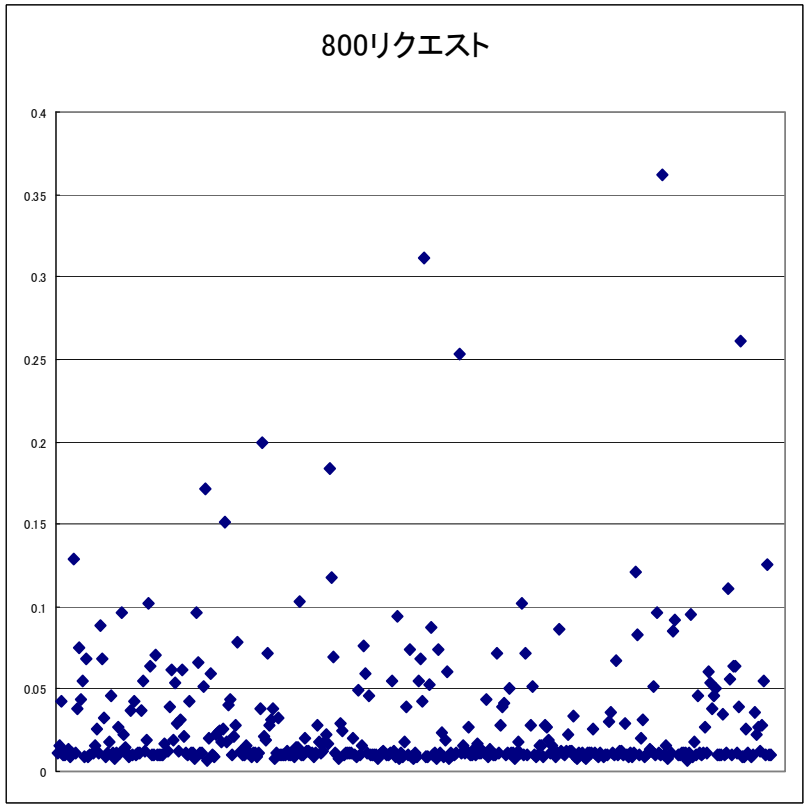
3ビット空間での例（原論文より抜粋）

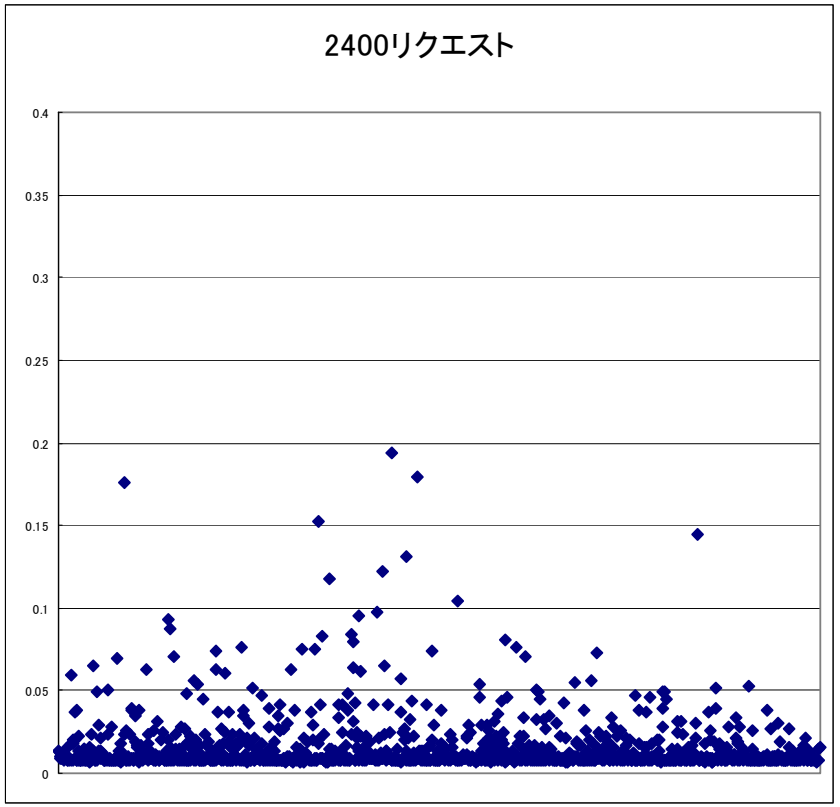
探索シーケンスの擬似コード

```
// ノードn上で、id のsucecssorを探す
  n.find_successor(id) { // ノード3上のプロシージャを実行
    n' = find_predecessor(id);  // idのpredecessorを探す、この場合 n' はノード0。
     return n'.successor;  // ノード  n' = 0 の finger table から id の  successorノードIDを返す
  }



  // ノード n で  id の predecessorを探す
  n.find_predecessor(id) {
     n' = n;  // まず、自分から
     while(id !∈ (n', n'.successor]) {     // finger  table内のエントリで、id が  intervalに入るまで
        n' = n'.closest_preceding_finger(id);     // id に最も近いノードIDを  n' とする
     }
     return n';  // finger table内で最もidに近いpredecessorを見つけた！
  }

// id が interval 内に含まれるエントリを探す
  n.closest_preceding_finger(id) { // 自分のfinger tableをみるよ！
     for i = m downto 1 {   // finger table の下の方から順番に
        if(finger[i].node ∈ (n, id)) {  // successorノードが 自分  と id との間にあれば、
           return finger[i].node;     // そのsuccessorノードIDを返す
        }
        return n;  // 無ければ、それは自分だよ
     }
  }
```

性能　SOD(Secure Overlay DHT)

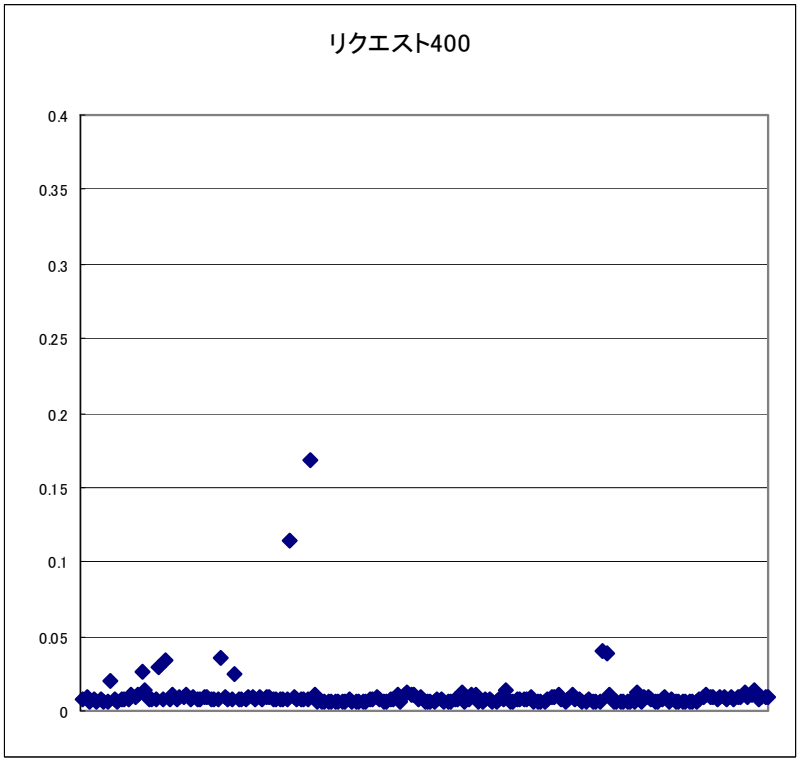
ノード数固定　100ノードでのレスポンス性能

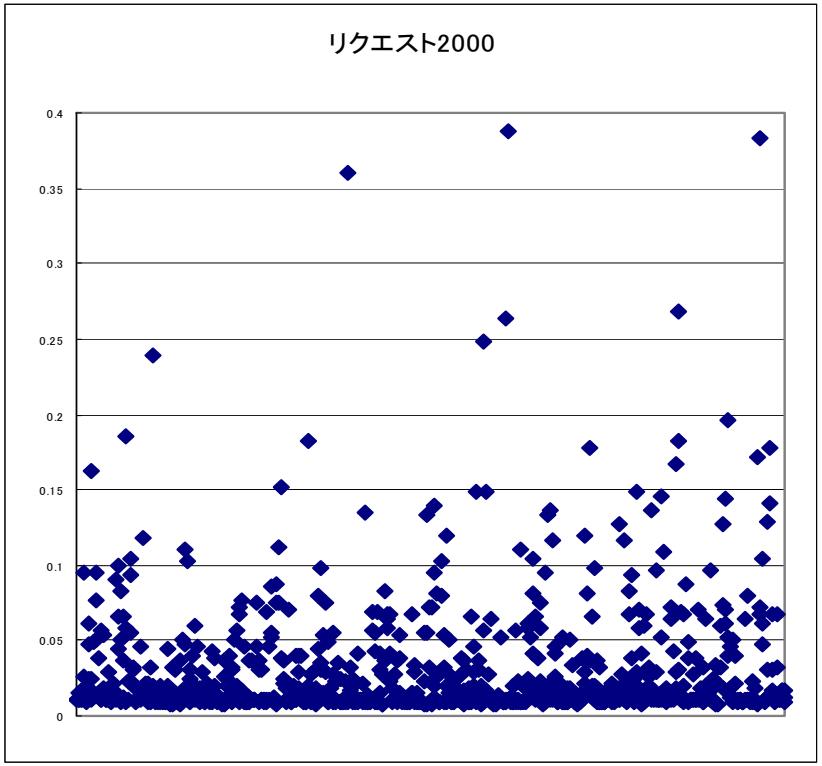




性能　SOD(Secure Overlay DHT)

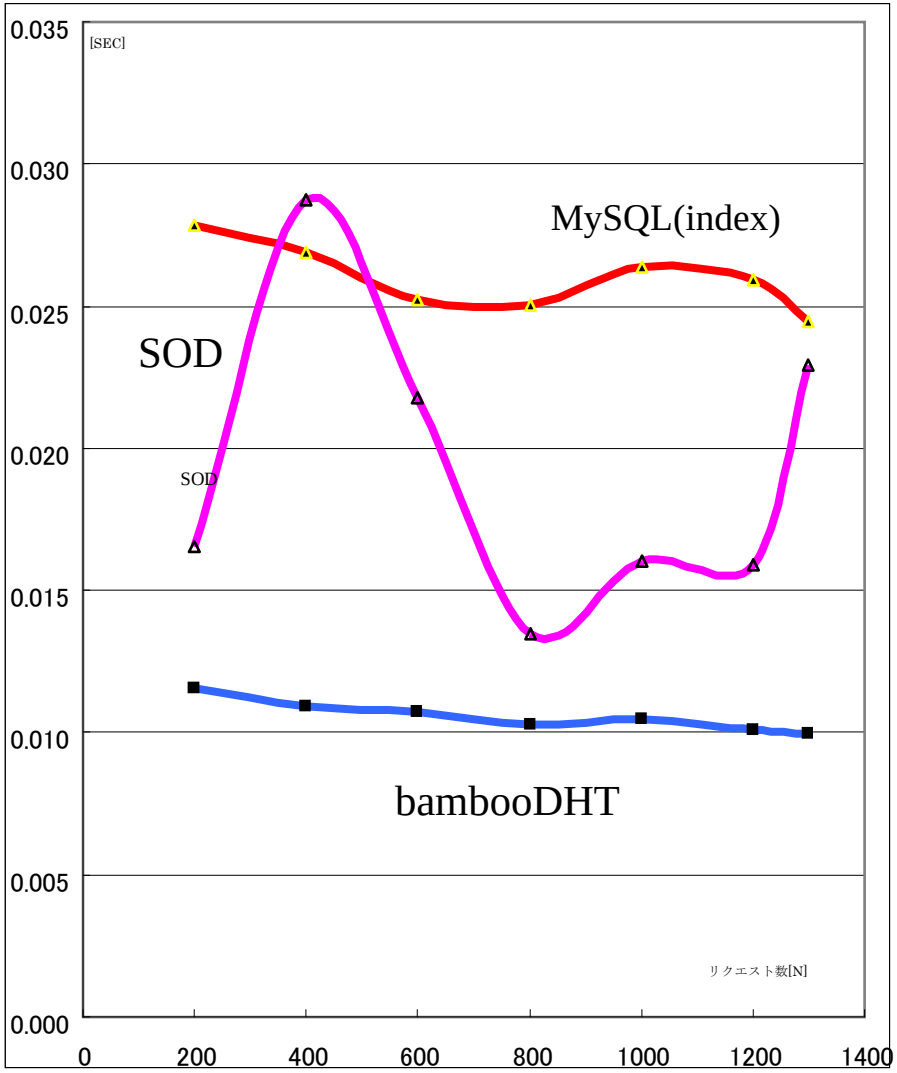
複数のASを跨いでDHTを構成





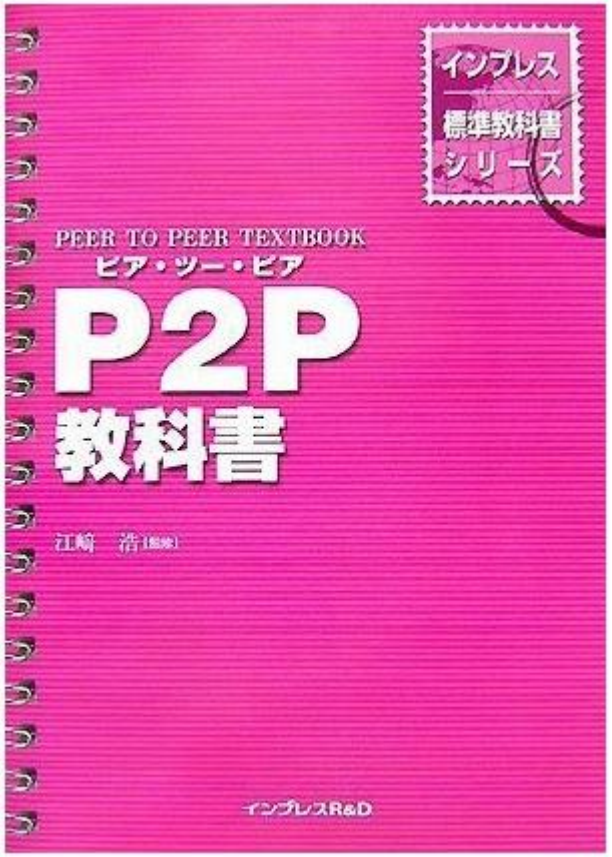
性能　SOD(Secure Overlay DHT)

MySQLとBambooDHTとの比較



代表的なアルゴリズム(捕捉)

- CAN
- Tapestry, Pastry (Microsoft)
- Kademlia (BitTorrent)
- Chord
- Koorde
- Broose
- Skip Graph (Yale & Google)



P2P教科書絶賛発売中！(2008.1.1発行)

実装

OverlayWeaver (Java)

Chord, Kademlia, Koorde, Pastry, Tapestry http://overlayweaver.sourceforge.net/

Bamboo DHT(Java) Pastry http://bamboo-dht.org/

OpenChord (Java) Chord http://open-chord.sourceforge.net/

Chimera(C) Tapestry http://current.cs.ucsb.edu/projects/chimera/

libtorrent (C++) Kademlia http://www.rasterbar.com/products/libtorrent/

ブログ「驟雨のカーネル探検隊(只今遭難中w」
オープンソースなDHT実装まとめ

http://d.hatena.ne.jp/syuu1228/20091015/1255577562

まとめ

1. 大規模P2Pオーバレイネットワークの構築が可能
2. データ(Key-Valueペア)量に依存しない探索性能
3. 揺らぎが大、ネットワーク距離の影響を受けやすい
4. 範囲検索ができない (Skip Graphはサポート)
5. ノードの参加離脱に対してフレキシブル

DHTとKVS	DHT	KVS
経路長	Multi-hop	No-hop
トポロジ情報	部分的に把握	全体を把握
適用規模	大規模	小規模

終了