Peer to Peer:

本格的P2Pアプリケーション時代のための基礎知識とネットワーク運用技術

砂原秀樹
油谷曉
奈良先端科学技術大学院大学
情報科学センター

スケジュール

- 前半
  - Peer to Peer アプリケーションの基礎知識
    - 砂原
  - ネットワーク運用技術
    - 油谷

Peer to Peerアプリケーションの基礎

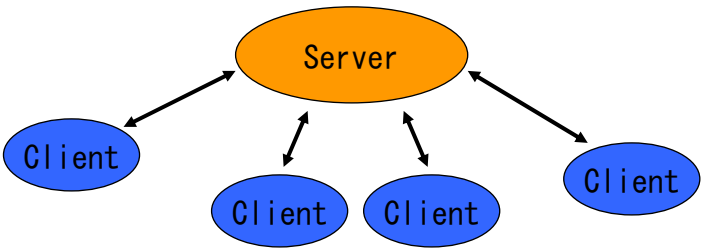
砂原秀樹
奈良先端科学技術大学院大学

P2Pアプリケーションとは?

- サーバ・クライアント型
  - サービスを提供するサーバと、それを利用するクライアント
- Peer to Peer型
  - ?????
  - 対等なノード同士のコミュニケーション

サーバ・クライアント

- 2つのプログラム
  - サーバ:サービスを提供するプログラム
  - クライアント:サービスを利用するプログラム

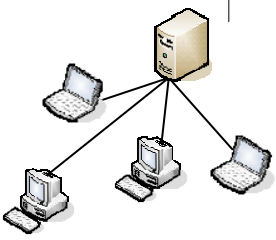


これまでのインターネットの宿題

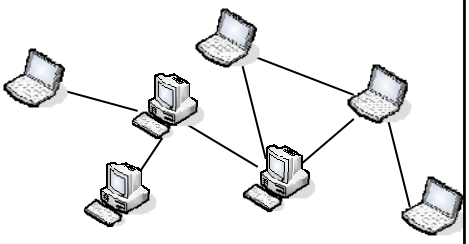
- Well Known Port
  - ポート番号の決めうち
  - その決められたポート番号で決められたサービスが提供されているとは限らない
- 規模の拡大と耐故障性
  - スケーラビリティの確保
  - 耐故障性の確保

クライアント/サーバとP2P

- 本質的にどこが違うのか

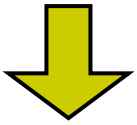


- クライアント・サーバ
  - 中心点の存在
  - 全体構成を集中管理
- P2P
  - 中心がない
  - 誰も全体を知らない

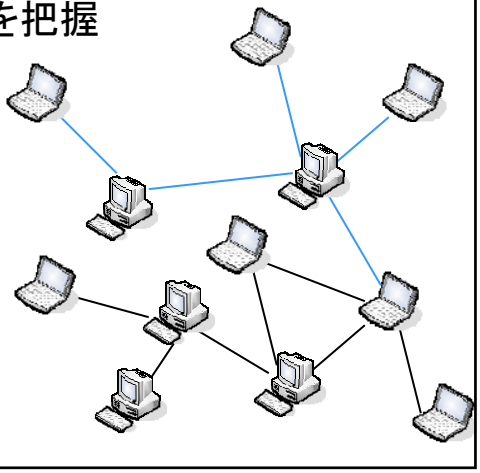


P2Pが持つ特徴

- システム全体の中心を持たない
- 各ノードが全体の一部を把握



- スケーラビリティ
- 耐故障性
- アドホック性



課題

- ノードの発見・サービスの発見
- データ検索機能
- ファイル転送
- セキュリティとプライバシと匿名性

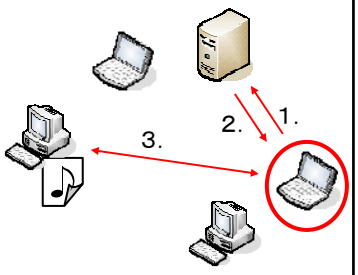
近年のP2P技術の遍歴(概略)

- ハイブリッド型P2P (napster)
- ピュア型P2P
  - 幅優先探索(gnutella)
  - 深さ優先探索(freenet, winny)
- 分散ハッシュテーブル型P2P
  - Plaxton (pastry)
  - Skiplist(chord)

ハイブリッド型P2P

- Napster(1999)
  - mp3ファイルの共有
  - ファイルのリストをサーバに保管
- 検索
  1. サーバにファイル名(曲名)を問い合わせる
  2. ファイルを持っているノードを知る
  3. 1対1で直接通信し、ファイルを得る

- 全米レコード工業会(RIAA)から提訴→破産
  - Roxioによる買収
    - DRMを採用した音楽配信(2003)
  - Napster Japan(2006)
    - タワーレコードとの合併会社

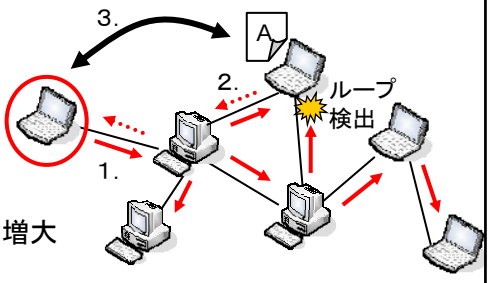


Napsterモデル

- ノードの発見
  - ディレクトリサーバを共有
- 検索は、ディレクトリサーバ上
- ファイル交換はデータを共有するノード同士直接
  - ノードはクライアントでありサーバ

幅優先探索(フラッディング)

- Gnutella(2000)
  - （mp3に限らず）ファイルを共有するP2Pソフト
  - 中央サーバが不要
- 検索
  1. フラッディングによる検索
  2. 検索ヒット(経路を逆に辿る)
  3. ファイルの取得
- 利点、欠点
  - ○ 柔軟な検索(部分一致)
  - × スケーラビリティ
    - フラッディング→トラフィック増大
- 効率: キャッシュは無い



Gnutellaモデル

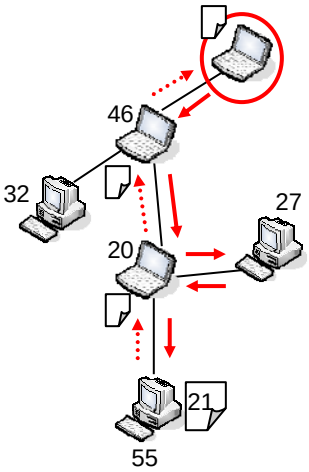
- ノード発見
  - シードノードリストの共有
- ファイル検索
  - 問い合わせの伝播
  - 問い合わせID
    - 同一問い合わせの削除
  - 転送カウント
    - 無限の転送の排除
- ファイル転送
  - ノード同士直接

深さ優先探索

- Freenet(2000)
  - 匿名性を考慮したファイル共有ソフト
- 幅優先探索(gnutella)との違い
  - 検索クエリーは隣接ノード1ずつ順に転送する
    - フラッディングしない
  - 人気のあるファイルはすばやく拡散し、自動的にクラスタ化される

Freenetでのファイル検索

- ハッシュ値を利用する
  - ファイル名、ノードのIPアドレス
- 検索(ハッシュ値21のファイル)
  - 目的のハッシュ値に近いノードから順にクエリを転送する
  - 検索ヒット後は経路を逆に辿る
    - 途中のノードにファイルに関する情報がキャッシュされる
  - ノード同士が直接通信し、ファイルを取得

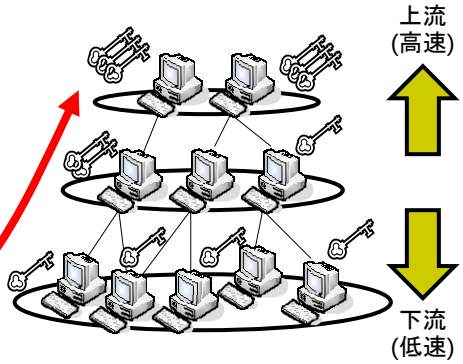


Freenetにおける匿名性

- 情報断片
  - 情報を断片化し分散して保有することで匿名性を確保
  - 情報の再収集は困難

Winnyにおける階層化

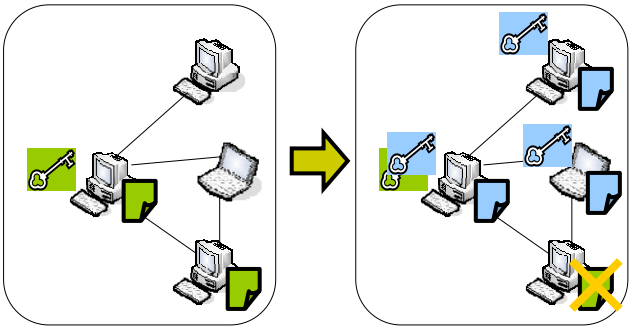
- Freenetをヒントにファイル・通信の暗号化、ノードの階層化など様々な工夫が凝らされている
- ファイルの場所を示すキーを上位層に向け拡散
  - 上流ノードほどキーが集まる
- 回線速度による階層化
  - スケーラビリティの確保



・主に上流に向けてキーを受け渡す
・主に上流に向けて検索をかける

Winnyでのファイルの拡散

- 一旦ダウンロードしたファイルは別のキーとキャッシュファイルの組み合わせになる
- 元ファイルを消しても別のキーが拡散してしまう
  - 人気があるファイルの拡散をとめることができない
  - そもそも誰がファイルをアップロードしたかも分からない状態



Winnyのモデル

- ノード発見
  - シードノードリスト
- ファイル検索
  - 問い合わせの伝播
- ファイル転送
  - キャッシュの導入
  - 転送の再開機能
    - 同一キーの分散されたキャッシュからの集約

Winnyの匿名性

- Proxyによる匿名性
  - キャッシュを拡散することで、そもそも誰が公開したものかわからなくする

ファイルの公開

- 公開されたファイル
  - キー (ファイルの要約情報)
    - ファイル名、サイズ、更新時刻、ファイル識別ID(ハッシュ値)、保有ノードのIPアドレスとポート番号
  - ファイル
- キーは隣接ノードに拡散されていく
  - キーの寿命に従って廃棄
  - 公開ノードが稼働しているなら定期的にキーは配布されているのでキーが存在することは、公開ノードの稼働の可能性を高める

ファイルの検索

- 問い合わせを伝播
- キーに一致する情報が格納されていれば、そのキー情報を問い合わせ元へ伝播

ファイル転送

- 受け取ったキーを元に直接ノードからファイルを取得
- 取得されたファイルは、さらに他のノードへの公開情報として格納される
  - キャッシュ
  - 新しいキーが作成され拡散される
- 他のノードがダウンロードし同様の操作が繰り返される

積極的キャッシュの生成

- キーは、一定の確率で書き換えられる
  - キーの拡散の途中で、そのノードをファイル保有ノードとして書き換える
- ファイル転送要求が到着したら、オリジナルファイル保有ノードにファイル実体を要求しキャッシュとして保持
  - 中継

ネットワークの階層化

- 取得するばかりで提供しないノードの排除
  - データ流量のバランス
- 上流と下流
  - データの流れの多いノード階層と末端ノード
  - 上流にキーとファイル実体が集まる
- 上流に問い合わせを送るとキーにマッチする
  - 無秩序な問い合わせの伝播を制約

クラスタリング

- 検索キーワードによる興味の分別とクラスタリング
  - 検索キーワードの似たノード同士を再接続
  - 同様のキーワードを用いるノード同士を近づけることで、クラスタリングを行う
  - 検索の効率化

Winnyの匿名化の仕組み

- キャッシュファイルの暗号化
  - どこに何が格納されているかをわかりにくくする
- 通信路の暗号化
  - やりとりされている情報による類推を防ぐ

データ転送の効率化

- キャッシュのブロック化
  - キャッシュは64k byte単位でブロック化され暗号化されている
- 異なるノードから異なるブロックを並行してダウンロードすることで転送速度を向上させる
  - TCPの限界を超えた転送
- アクセスが集中すると接続を解除する
  - アクセスの分散化
- 各ノードのダウンロード速度とアップロード速度の制御
  - ノード同士の通信量の公平化
  - Give and Takeの自動的実現

Winnyが成したこと

- ネットワーク(オーバレイネットワーク)への容易な参加
- P2Pの可能性の提示
  - スケーラビリティ
- やらなければならないこと
  - 情報の流通の制御
  - 情報の識別

BitTorrent

- 2001年にブラム・コーエン(Bram Cohen)氏が開発したファイル共有システムのプロトコル
  - ファイルの断片を分散して保持することで、段ロードの高速化を図る
  - 多くの利用者がダウンロードをすることによって分散化が進みダウンロードの速度が上がる
  - 匿名性は無い

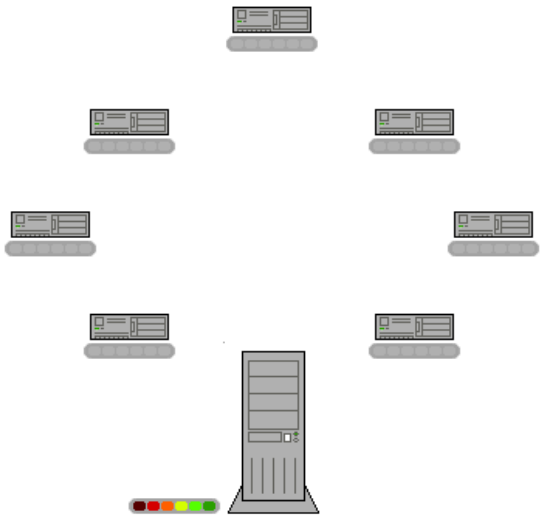
用語

- トラッカー (Tracker)
  - 新しく接続してきたノードにPeer群のIPアドレスを教えるサーバ。トレントファイルを集めたインデックス・サイトを呼ぶことがあるが、これは誤り。
- トレントファイル
  - トラッカーへのリンクやメタ情報を含むファイル。拡張子が *.torrentとなっている。クライアントはこのファイルを読み込みファイルを取得する。
- ピア (Peer)
  - 実際にファイルの断片を保持するノード群。
- シード／シーダー (seed/seeder)
  - 完全なファイルを格納しているノード。ファイルの初期提供者及びダウンロードを完了し他への公開を行っているノード双方を指す。

用語

- リーチャー（leecher < leech：蛭）
  - ダウンロード中のノード。そもそもコーエンはデータの提供をせずダウンロードだけをするノードを指していたが、現状ではすべてのダウンロード中のノードを指すようになった。
- スウォーム（swarm：群れ）
  - 同じトレントファイルによって同じファイルを提供/ダウンロード中のノードのグループ。ほとんどの場合一つのノードはその一部とだけ、直接データのやりとりを行っている。
- シェア・レシオ（Share Ratio）
  - アップロード量とダウンロード量との比。オープンソースソフトウェアなど、開発者が継続的にシードの提供を続けている場合を除いて、基本的に1：1 に達するまで共有を続けるべきであるとされる。

動き



出典: フリー百科事典『ウィキペディア（Wikipedia）』

.torrent

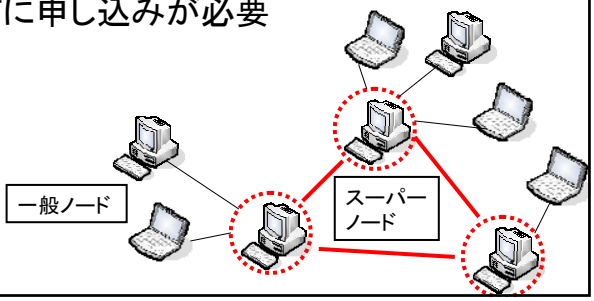
- url of the tracker
- Pieces <hash1,hash2,….hashn>
- Piece length
- Name
- Length
- Files
  - Path
  - length

Tracker

- Peer cache
  - IP, port, peer id
- State information
  - Completed
  - Downloading
- Returns random list

skype

- Skype Technologies(スウェーデン)
  - 元KaZaA開発メンバーにより開発
- VoIP + P2Pにより無料の音声通話サービスを実現
- 一般の固定電話や携帯電話とも通話可能
  - 有料のSkypeOUTに申し込みが必要



skype

- P2P式のIP電話
  - KaZaAというP2P型ファイル交換サービスを作ったメンバーが開発
- インスタントメッセージ機能も有する
- Skype Outで、PSTN網(電話番号がついた世界)へも通信可
- Skype Inで電話番号を取得することも可能(日本では、Fusion Communicationがサポート)
- 留守番電話サービス(ボイスメールサービス)など

ネタもと

- “An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol” by S. Baset and H. Schulzrinne at Columbia, September 15, 2004
  - “We observe, ..., we think, ..., we conjecture..., etc.”
- リバースエンジニアリングなので、本当かどうかは不明
  - そもそも、そういうことしていいのか？
  - 接続状況の観測をしているとわかることもある

skypeの技術的特徴

- **Codec---provided by “Global IP Sound (GIPS)”**
  - **67 bytes packet payload**
  - **24 to 120 kbps**
  - **http://www.globalipsound.com/**
- **ソフトウェアによるDSP処理**
  - **Codec, ミキシング、エコーキャンセル**
  - **ThinkPad T30/T40クラスのPCでアクティブskypeセッションを処理するのに40-60%のCPUを消費する**
- **スーパーノードがユーザの場所、オンライン/オフライン等の情報を管理**
- **一つのスーパーノードで500ユーザーを管理 (アルゴリズムとしてはDHT - 分散ハッシュ表をつかっているらしい**
- **NATやfirewallを越える機能を有する**
  - **Peer relay/TCPトンネル**
- **暗号化されたメディアチャネル**

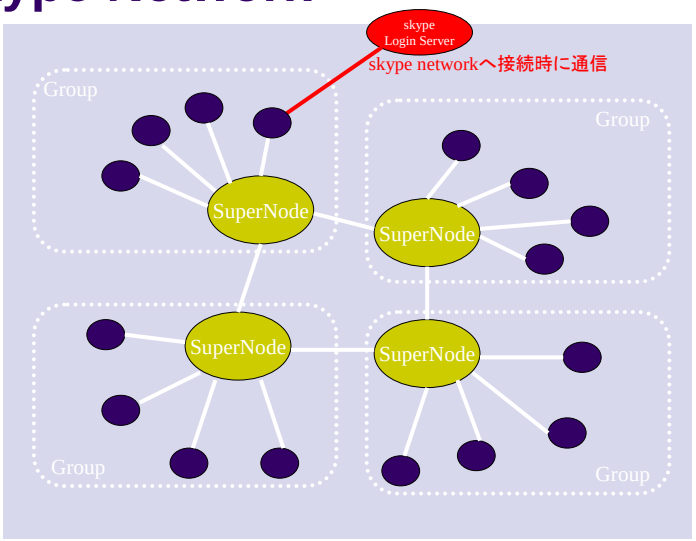
手順

- **Host Cache List**
  - **スーパーノードのIPアドレス/ポート番号のリスト**
  - **初期値は、インストール時にskype.comに接続する際に与えられる?**
- **Host Cache Listの中のスーパーノードに接続**
- **スーパーノードは、ログインサーバへ誘導、そこで認証**
- **接続完了後近隣ノード(22ノード?)へOnlineを通知**

スーパーノード

- 以下の条件を満たすノードが自動的に選ばれる
  - グローバルアドレスを持つ
  - CPUの能力が高い
  - 大容量のメモリを持つ
  - 起動時間が長い
  - ?? RTTが短い ??

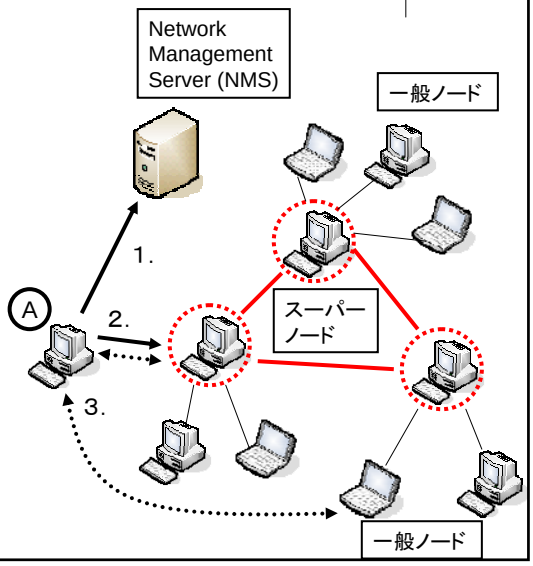
skype Network



skype

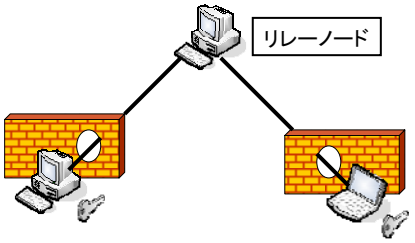
- スーパーノードが基盤
  - ノード情報一覧を同期
  - 全体の1%程度
- スーパーノードの条件
  - 一定時間以上ネットワークに参加
    - ＋グローバルIPを持つ
  - 内向きのセッションを張れる
    - FW, NATの制限が小さい
  - 回線が太く、計算機の能力が高い
- MNS
  - スーパーノードを管理
- 新規参加ノード（A）

1. スーパーノードの問い合わせ
2. Skypeネットワークに参加
3. 接続先の検索
   - スーパーノードが解決



skype

- 内向きセッションを張れないノード同士の通話
  - リレーノードが通話データを橋渡
    - スーパーノードと同程度の性能が要求される
  - 通話データは暗号化されているので盗聴が困難
- 通話データの暗号化
  - 対称鍵暗号(AESを利用)256bit長の鍵を利用
  - 上記鍵の受け渡しには、PKIベースの2048bit長の鍵が利用される(PKI証明書はログイン時にSkypeのサーバにより認証)



分散ハッシュテーブル(DHT)

- DHT以前のP2Pの問題点
  - スケーラビリティの限界
  - 検索時における不確実性
    - 情報があるのか、無いのか判断できない
    - 探し出すまでに数日かかる事もある

- ハッシュを基とした新しいP2Pネットワークで上記の問題点を解決

DHTの基礎概念

- フラットで広大なID空間(160bitsが多い)
  - ノードとデータが同一のID空間にハッシュ関数を利用して配置される

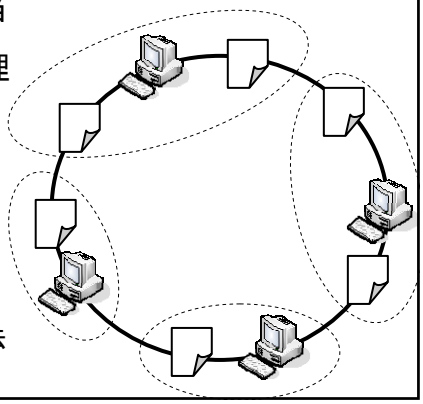
•個々のノードはID空間の一部を担当

•担当領域内に該当するデータを管理

•ハッシュにより均一に負荷分散

•個々のDHTにおける主な違い

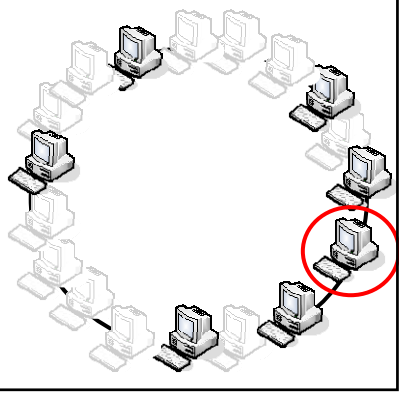
- •情報の検索手法
- •P2Pネットワークを維持するための手法



検索における基礎戦略

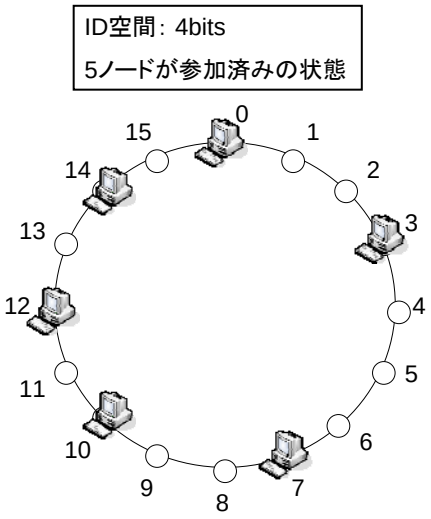
- 近隣ノードの情報は密に持つ
- 遠方のノードの情報は疎に持つ

- ID空間全域に到達性を持つ
- 管理するノードの情報量を抑える
- 提案されているDHTは検索速度、ノードの情報量共にLogオーダー



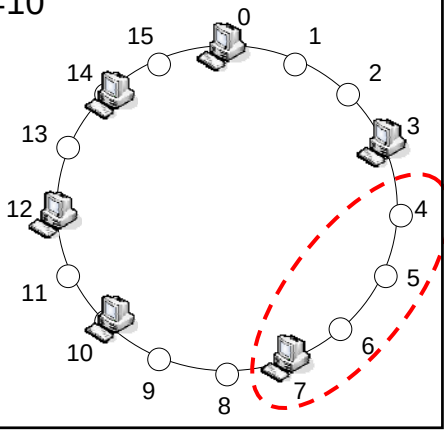
Chord

- 検索アルゴリズム
  - Skiplist
- ID空間160bits
  - SHA-1を利用
- 検索速度
  - O(logN)
  - N:ノードの数



Chord

- Finger関数の導入
  - あるIDに対して次のノードが存在するIDを返す
  - 例) Finger(1)=3, Finger(8)=10
- ノードの担当領域
  - ID:nの担当ノードはFinger(n)
  - 例) Node:7は4-7が担当領域
    - Finger(4)=Finger(5)=Finger(6) =Finger(7) = 7
  - (参加ノード間が担当領域)



Chord

- skiplistの経路表(m-bitsのID空間)
  - Finger(NodeID+2^i mod 2^m)　i: 0～m-1
- 例）NodeID:0の場合（4bitsのID空間）
  - Finger(0+2^i mod 16) i: 0～3

• 検索例) 14の情報をノード0が検索

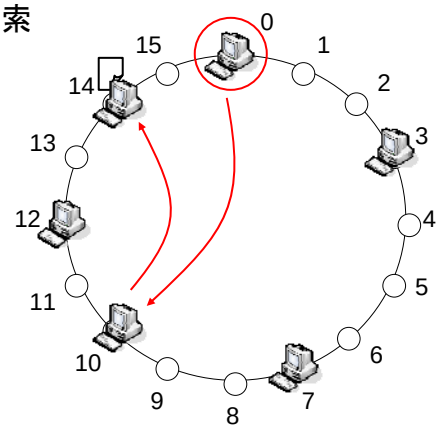
  • 0 → 10 → 14

NodeID: 0の経路表

ID	1	2	4	8
node	3	3	7	10

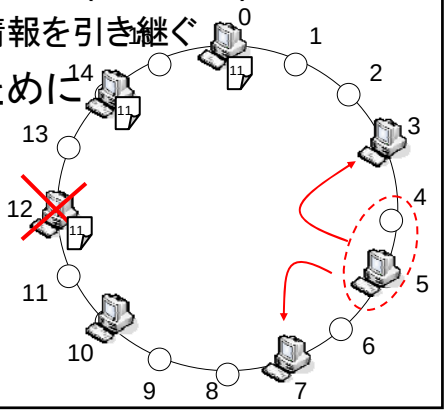
NodeID: 10の経路表

ID	11	12	14	2
node	12	12	14	3



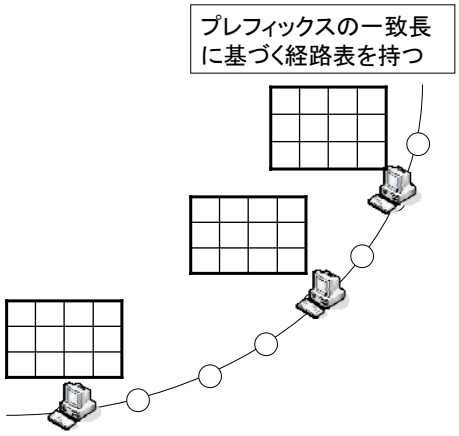
Chord

- 新規参加(ノード5が加わる場合)
  - Finger(5)のノード(NodeID:7)の情報を調べる
    - 既存参加ノードを事前に知る必要あり
  - ノード7から一つ前のノードの情報(NodeID:3)を取得
  - ノード7からID4,5に関する情報を引き継ぐ
- 離脱よる情報紛失を防ぐために
  - 定期的に経路表を更新
  - 時計回りにk個のノードに予めキャッシュを渡しておく



pastry

- 検索アルゴリズム
  - Plaxton
- ID空間128bits
- 検索速度
  - O(logN)
  - N:ノードの数



pastry

- プレフィックスの一致長に基づく経路表
  - 一致長が長い　→　ID空間内で近い距離
  - 一致長が短い　→　ID空間内で遠い距離
- 例）NodeID: 52345の経路表(10進数)
  - 注）Pastryでは16進数で取り扱う事が多い

プレフィックス一致の後に付け足す数

遠方のノードの情報
は疎に持つ

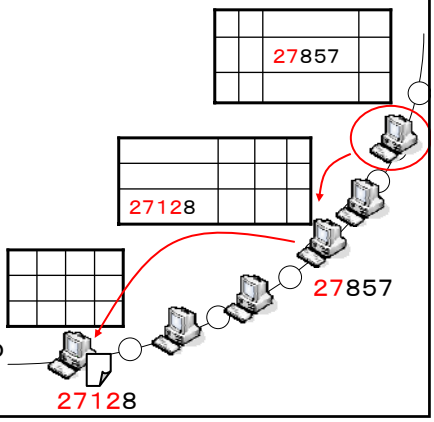
プレフィックスの一致長

近隣ノードの情報
は密に持つ

	0	1	2	・・・	9
0	03857	13562	22358		93756
1	50345	51872	52643		59359
2	52083	52153	52283		52984
3	52303	52312	52325		52390
4	52340	52341	52342		52349

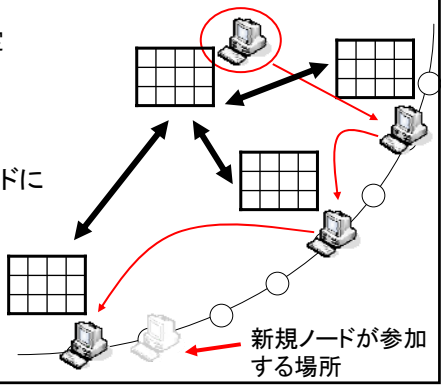
pastry

- 検索
  - 経路表から検索対象のIDと最長一致するノードを選び、検索クエリーを転送する
  - 例）ID:27121を検索する場合
    - 検索ノード→27857→27128
    - ノード27128が検索対象を管理
- 検索手法の拡張
  - Leaf set
    - 近隣ノードを一定数保持
    - ショートカット経路の確保
  - Neighborhood set
    - RTTの小さいノードを一定数保持
    - 経路表更新の際に優先候補となる



pastry

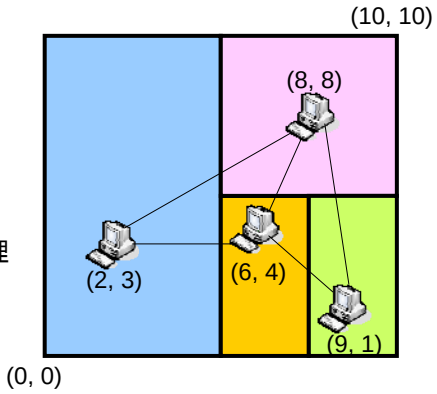
- 新規参加
  - 新規参加ノードはハッシュ関数により自分のノードをIDを決定する
  - 初期ノードを足がかりに情報検索と同じ手法で参加ノードのIDを管理しているノードを検索する
  - 参加ノードは中継ノードと経路表を交換し、また中継ノードの経路表は順次更新される
  - Leaf setは参加ノードと最もIDの近いノードから、neighborhood setは初期ノードから情報を貰う
    - 初期ノードはIP層として近いという仮定
- 離脱
  - 定期的に経路表を更新
  - 経路表を使って離脱ノードに近いノードに問い合わせて、離脱ノードの代わりのノード探し、経路表に追加する。



CAN

- ハッシュ空間
  - N次元トーラス
    - ここでは2次元で説明（0～10の領域を使う）
- 検索速度
  - オーダーsqrt(N) N:ノード数

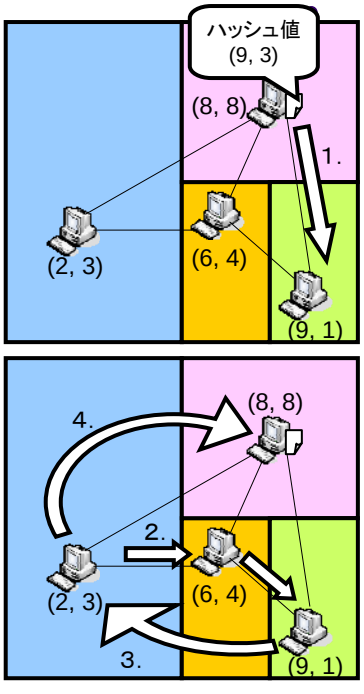
    注）2次元で考えた場合
- ノード情報の管理
  - 隣接している領域のノード情報を管理
  - 隣接ノード同士で情報のバックアップ



CAN

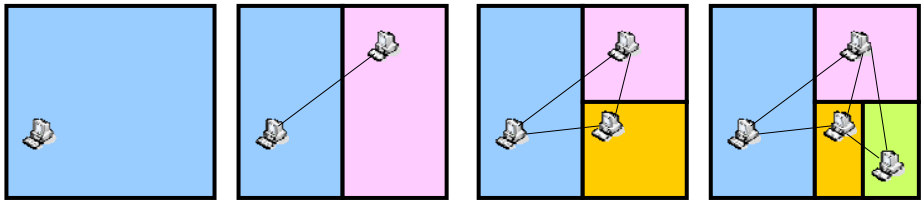
- 情報の登録
  1. ファイル名をハッシュ関数にかける
  1. 得られたハッシュ値を管理しているノードにハッシュ値とノード情報を通知する
- 情報の検索
  2. 欲しいファイル名をハッシュ関数にかける

     ハッシュ値(9, 3)のファイルが欲しいとする
  2. 得られたハッシュ値を管理しているノードに問い合わせる
  3. 情報を所持しているノード情報を得る
  4. 直接ノード同士で通信し、目的の情報を得る



CAN

- 新規ノードの参加
  - ハッシュ関数を利用して自分のIDを決定する
  - 参加ノードのIDを管理しているノードを探し、領域を半分わけてもらう
  - 同時に隣接ノードの情報を更新する
- ノードの離脱
  - 隣接ノードが領域を拡張し、離脱ノードの領域を引き継ぐ
  - ノードやデータの情報は隣接ノードで予めバックアップ



DHTまとめ

- ハッシュ関数が生成するランダムなID
  - 情報＆ノードがID空間に均一に分布する
  - ランダムなIDが負荷分散を担保する
- 近隣は密に、遠方は疎に情報を持つ
  - ID空間全域への到達性を確保しつつ、logオーダーのルーティングテーブルですむ
  - 検索も同様にlogオーダー(非常に高速)
- どの情報をだれが管理すべきが決定されている
  - 情報の有無を検索後に入手できる
- 欠点
  - ハッシュを利用しているため完全一致検索のみ可能

P2Pネットワークへの攻撃

- 無意味なファイルの大量配布
  - 興味を引きそうなファイル名と無意味なデータの配布
- ファイルのねつ造
  - 偽物のファイル実体を多数生成する
- ウィルス問題
- アップロード量とダウンロード量のバランスの破壊
  - プログラムの改変

最近のトピック

- 既存サービスの置き換え
  - DNS、トレーサビリティなど
- セキュアオーバレイ
- ユビキタス環境への適応
  - センサネットワーク、モバイル

P2P SIP (IETF)

- P2PベースのInternet電話技術の標準化
  - http://www.p2psip.org/
- Paris(Ad Hoc), Vancouver(Ad Hoc), Dallas (BOF)
- WG化の準備作業中
  - Charterを作成中 (Mailing List)

P2Pが生み出した技術

- 分散検索
  - DHT(分散ハッシュテーブル)
  - Chord
  - Pastry
  - CAN
  - …
- 分散キャッシュ
  - アクセス履歴に基づくデータの分散
  - 部分キャッシュとデータの再構成
- 高速転送技術
  - 分散したデータの収集と転送(複数のTCPコネクション)

P2Pが生み出した文化

- 利用者のネットワーク参加への障壁の簡易化
- 匿名性と意見交換
- 情報の共有文化
  - データそのものは無価値
  - その利用方法に価値がある
- データの流量の制御
  - Uploadの量とDownloadの量
  - Give and Take
- 本当のインターネットの姿

複雑すぎる現在のインターネット

- 利用できるようになるまでの準備
  - アクセスライン
  - 種々のサーバの設定
    - DHCP, DNS, ○○サーバ
  - さらに増える機能要素
    - Home Agent
    - Security (認証局など)
- 本来不要のもの
  - Firewall, 検疫ネットワーク, NAT
- 管理コスト
- 問題解決手順の複雑さ

本来のインターネット

- End-to-Endコミュニケーション
  - 両端ががんばる
  - 途中はうまくやる
- セキュリティ
  - Endノードは何が起こっているか知っているはず
- これから必要なこと
  - ノードの認証
  - 攻撃を排除するバックボーン
  - 信頼できるノードで構成されるオーバレイネットワークとその上に構築されるサービス