P2P の概念

典型的な問題 (typical task)
「ある条件をみたすホストを探す」
(to look up a host which meets some requirement)

「条件」-- 例: あるファイルを持っている
(requirement -- e.g. the host keeps a certain file)
「探す」-- 接続するのに必要な情報を入手する
(look up -- get information to connect to the host
  TCP/IPでは, (IP アドレス, ポート番号)を知る
((IPaddr,port) for TCP/IP)

完全な分散システム (distributed system)
ノードが対称 (symmetric nodes)
  中央制御,階層がない (no central control, no hierarchies)
  参加離脱の管理なし (no control on join and leave of nodes)

scalability
  台数が増加しても動作する

アドホック性 (ad hoc)
  (参加離脱の自由) (unconstrained join and leave)

耐障害性 (resistance for failures)
  (単一障害点なし) (no single-point-of-failure)

P2P の歴史 (histories)

1999 Napster
  楽曲のメタデータだけをサーバに集約
  (keep metadata of music files on a server)
    メタデータ - データに関するデータ
      ここでは楽曲のタイトル, ミュージシャン, ...
  コンテンツ公開は自分のノード上でOK
  (music files are placed on users' machines)
  コンテンツの取得はノード同士で
  (transport files between user machines)

  著作権問題から2001/07にサーバを停止
  (server was closed by copyright problems)

2000 Gnutella
  ファイル共有 (file sharing)
  サーバなし (no servers)

2001 Kazaa
  ファイル共有 (file sharing)

2001 BitTorrent
  ファイルの配信 (delivery of files)

2002 Winny
  ファイル共有 (file sharing)

2004 Share
  ファイル共有 (file sharing)

探索の問題 (searching files)

キーから実体を探す (search an item from the key)
(ファイル名からファイルの所有ホストを探す)
(i.e. search a file from the name of the file)

Three ways to keep information:

(1)メタデータとデータを同じホストに
(keeping the data(item) and the metadata(key) on a server)

普通のファイルサーバ, uploader
(usual file servers)

(2)メタデータだけを集約 - hybrid P2P
(collect only metadata on a server)
データは各ノードに - Napster
(data files are located on users' node)
配信の負荷軽減
(the load to deliver files are distributed)
公開が簡単
(easy to share files - only upload the metadata)

(3)メタデータも分散 - Pure P2P
(metadata is also distributed)

(3a) unstructured
ホストは random graph で結合 - 特にルールがない
(nodes are connected without any rules)
キーの割り当てもランダム
(keys are distributed without rule)
あるファイルをどのノードが管理するか
(i.e. which node will keep a file is not determined)
参加離脱の管理コストが低い
(low administration cost for joins/leaves)

(3b) structured
キーをホストに割り当てる方法が決まっている
(rules to assign keys to hosts)
効率の良い探索が可能
(efficient lookup is possible)
参加離脱の管理コストが発生
(administration cost for joins/leaves)

探し方 (the search)
unstrucured
flooding (Gnutella)
方向づけ (FreeNet)
structured
DHT(Distributed Hash Table) (Chord, Pastry, ...)

==========

Small World

"Collective dynamics of 'small-world' networks"
Duncan J. Watts and Steven H. Strogatz
Nature 393, 440-442 (4 June 1998)

Small World experiment (Wikipedia)

==========

The small world experiment comprised several experiments conducted by Stanley Milgram examining the average path length for social networks of people in the United States. The research was groundbreaking in that it suggested that human society is a small world type network characterized by short path lengths. The experiments are often associated with the phrase "six degrees of separation", although Milgram did not use this term himself.

Basic procedure

1. Though the experiment went through several variations, Milgram typically chose individuals in the U.S. cities of Omaha, Nebraska and Wichita, Kansas to be the starting points and Boston, Massachusetts to be the end point of a chain of correspondence. These cities were selected because they represented a great distance in the United States, both socially and geographically.

2. Information packets were initially sent to randomly selected individuals in Omaha or Wichita. They included letters, which

detailed the study’s purpose, and basic information about a target contact person in Boston. It additionally contained a roster on which they could write their own name, as well as business reply cards that were pre-addressed to Harvard.

3. Upon receiving the invitation to participate, the recipient was asked whether he or she personally knew the contact person described in the letter. If so, the person was to forward the letter directly to that person. For the purposes of this study, knowing someone "personally" was defined as knowing them on a first-name basis.

4. In the more likely case that the person did not personally know the target, then the person was to think of a friend or relative they know personally that is more likely to know the target. They were then directed to sign their name on the roster and forward the packet to that person. A postcard was also mailed to the researchers at Harvard so that they could track the chain’s progression toward the target.

5. When and if the package eventually reached the contact person in Boston, the researchers could examine the roster to count the number of times it had been forwarded from person to person. Additionally, for packages that never reached the destination, the incoming postcards helped identify the break point in the chain.

Results

Shortly after the experiments began, letters would begin arriving to the targets and the researchers would receive postcards from the respondents. Sometimes the packet would arrive to the target in as few as one or two hops, while some chains were composed of as many as nine or ten links. However, a significant problem was that often people refused to pass the letter forward, and thus the chain never reached its destination. In one case, 232 of the 296 letters never reached the destination.[3]

However, 64 of the letters eventually did reach the target contact. Among these chains, the average path length fell around 5.5 or six. Hence, the researchers concluded that people in the United States are separated by about six people on average. And, although Milgram himself never used the phrase "six degrees of separation", these findings likely contributed to its widespread acceptance.[1]

In an experiment in which 160 letters were mailed out, 24 reached the target in his Sharon, Massachusetts home. Of those 24, 16 were given to the target person by the same person Milgram calls "Mr. Jacobs", a clothing merchant. Of those that reached him at his office, more than half came from two other men.[4]

The researchers used the postcards to qualitatively examine the types of chains that are created. Generally, the package quickly reached a close geographic proximity, but would circle the target almost randomly until it found the target’s inner circle of friends.[3] This suggests that participants strongly favored geographic characteristics when choosing an appropriate next person in the chain.

==========

(Natureの論文の冒頭)

"... Ordinarily,
the connection topology is assumed to be either completely
regular or completely random. But many biological, technological
and social networks lie somewhere between these two extremes.
Here we explore simple models of networks that can be tuned
through this middle ground: regular networks rewired to introduce
increasing amounts of disorder. We find that these systems
can be highly clustered, like regular lattices, yet have small
characteristic path lengths, like random graphs. We call them

small-world networks, by analogy with the small-world
phenomenon (popularly known as six degrees of separation)."

Regular network
  regular ring lattice (Fig. 1, left)

Random network (Fig. 1, right)

rewiring probability: p
  p = 0 -> regular
  p = 1 -> random
  0 < p < 1 -> intermediate region

characteristics of network
  L(p): characteristic path length
    number of edges in the shortest path between two vertices, averaged
    over all pairs of vertices
    (最短距離の平均)

  C(p): clustering coefficient
    vertex v has kv neighbours.
    at most kv(kv-1)/2 edges can exist between them.
    Cv denotes the fraction of these allowable edges that actually exist
    (あるノードの隣接ノードの間にリンクがある割合)

  n: number of vertices
  k: number of neighbours

  assumption: sparse, connected graph
  n >> k >> ln(n) >> 1
  Fig. 1: n=20, k=4, ln(20)=3

  (p->0)
  L ~ n/2k >> 1
  large world

  C ~ 3/4
  highly clustered

  (p->1)
  L ~ Lrandom ~ ln(n)/ln(k)
  small world

  C ~ Crandom ~ k/n << 1
  poorly clustered

Fig. 2
  There is a broad interval of p:
    L(p) is almost as small as Lrandom
    C(p) >> Crandom
  "small world network"

  short cut is important

Table 1
  examples of small world networks

Gnutella
  Pure P2P
    中央サーバなし (no central servers)
  unstructured
  ファイル交換 (used for file sharing)

History of gnutella
  2000-03-14
  Nullsoft社が最初のプログラムを作成

  (developed by Nullsoft)
  public beta testing の翌日に削除
  (deleted on the next day of its public beta testing started)

  reverse engineering によるプロトコル解析
  (protocol analysis by reverse engineering)
  多数のプログラムが作成された
  (many compatible programs developed)

  http://ja.wikipedia.org/wiki/Gnutella
  http://en.wikipedia.org/wiki/Gnutella

用語(glossary)
servent
  gnutella client のこと
  server + client の造語

descriptor
  gnutella protocol で交換されるメッセージ
  (message exchanged by servents)
  5 types (Ping, Pong, Query, QueryHit, Push)

---
gnutella protocol

接続 (connection)
  相手serventの (IP address, port) を入手
  (get IP address and port number of the peer servent)
    初期接続の問題 - いろいろな手段
    (the initial connection problem - several ways)
      ファイルにあらかじめ入っている
      (stored statically in the configulation file)
      特定のウェブページに接続先リストがある
      (some websites publish lists for initial connection)
      前回接続した相手をファイルに保存しておく
      (record servents which have been connected successfully in
       previous activation)

  TCP connection を張る (establish a TCP connection)

  そのあとのやりとり (after the TCP connection established,)

    send 「GNUTELLA CONNECT/<protocol version string>\n\n」
    receive 「GNUTELLA OK\n\n」

  gnutella connection が設定された (was established)

  ここからは descriptor の交換
  (after that, descriptors are exchanged on the connection)

接続拒否 (reject connection)
  servent は接続を拒否してよい
  (servents may reject incomming connections)
    現在の接続数が多過ぎる
    (already accepting too many connections)
    protocol version の違い
    (protocol version mismatch)
    ブラックリスト (blacklist)
---
descriptor の形式 (format)
  two parts:
    descriptor header (fixed length)
    payload (variable length)

その前に…
endian / byte order
  2以上のバイト数(2,4など)をもつフィールドの, パケット中でのバイトの順序

(ordering of bytes for multi-byte data in messages)
たとえば 10000 = 0x2716 は 16bit の整数に収まる.
(for example, 10000 = 0x2716 may be sent in two formats:)
これを送信するとき
(A) 0x27 0x16 上の桁から送信する: Big Endian
(B) 0x16 0x27 下の桁から送信する: Little Endian
プロトコルの規定に含めておく必要あり
(the ordering should be defined in the protocol definition)

CPUの種類によって, メモリ上でのバイト順が違うのが原因
(CPU architecture defines the ordering)
たとえば Intel のプロセッサは Little Endian
同じCPUが読み書きするのなら問題にならない
(no conversion necessary for the same processor)

たいていのインターネットのプロトコルでは Big Endian になる
(most internet protocols use Big Endian)
これを network byte order という

メモリ上にある変数をパケットバッファにコピーするときに問題になる
(conversion is necessary when storing/loading values in packet buffer)

プログラムにおいては, バイト順を入れ替えてコピーする必要あり
(network programs should use proper ordering)
(第一回 note 01, server.c L.25)
"serv_addr.sin_port = htons(8888);"
htons: バイト順を入れ替えるマクロ
Host order TO Network order, Short
CPUの種類によって定義が変えてある;
ソースプログラム上では同一でOK
(the macro definition is dependent on the CPU)

descriptor header
23 byte

Descriptor ID (16 byte)
Payload Descriptor (1)
TTL (1)
Hops (1)
Payload Length (4)

Descriptor ID
A 16-byte string uniquely identifying the descriptor
on the network

UUID (Universally Unique IDentifier)
or GUID
そのホストのMAC address, random number, system clock などから生成
(generated from the MAC address, random number, system clock of the host)

descriptorの識別に重要な役割
(important for the identification of descriptors)
重複のチェック (check duplication of a descriptor)
routing

Payload Descriptor
5 types of descriptorの識別 (identification)
00 - Ping
01 - Pong
40 - Push
80 - Query
81 - QueryHit

TTL (Time To Live)
この descriptor を転送する残りのホップ数
(remaining routing hops for this descriptor)
次のservantに転送する時に -1 する
(decreased when routed to the next servent)

ネットワーク上のgnutella traffic を制御する重要な値
(important to control gnutella traffic)

Hops
この descriptor が転送された回数
(number of hops this descriptor has been routed)
TTL(0) = TTL(i) + Hops(i)

★なぜ TTL と別に保持しているのか? (why not unify with TTL?)
(おそらく)このフィールドがないと, 受信したノードは送信元が
「どれくらい離れているか」がわからない
返信の初期TTLをそれを使って適切に設定する

Payload Length (32bit)
payload は header のすぐ次から
(payload immediately follows the header)
次の descriptor の開始位置を示す唯一の指標
(indicates the beginning of the next descriptor)
c.f. there is no message boundary on TCP connection

---

descriptor の各論 (details)

Ping
gnutella network 上の他の servent を探すため
(to find other servents on the gnutella network)
flooding による転送
(routed by flooding)

no payload (payload length = 0)

Ping を受けた servent は Pong を返す
(servents received Ping will answer with Pong)

Pong
Ping への応答 (answer to Ping)
servent に関する情報を返送
(with various information of the servent)

descriptor header 内の descriptor id field は
Ping の descriptor id field に一致させる
(descriptor id of Pong == that of Ping)
Ping の来た経路を逆にたどって転送
(routed along the route of Ping reversely)

payload 14 byte
Port (2)
IP Address (4)
Number of Files Shared (4)
Number of Kilobytes Shared (4)

Port, IP Address
servent の接続受け入れアドレス
(access point of the servent)
Number of Files Shared, Kilobytes Shared (of the servent)
その servent が公開しているファイルの個数, サイズ

かならずしも自分に関する情報でなくてもよい
(might be the information of other servents)
他の servent の情報

得られた情報の使い道 (utilizing the information collected)
「知っている servent」を蓄積しておく
(record the information in the "known servent" database)
新規接続を行う
(using the infomation in Pong, new connection is possible)