A Survey of Peer-to-Peer Content Distribution Technologies
- Additional Slides -

Stephanos Androutsellis-Theotokis and Diomidis Spinellis
ACM Computing Surveys, December 2004

Presented by
Sangyoun Paik
Introduction

- Structured Architectures: CAN, Chord, Tapestry
  - Distributed Hash Table based P2P overlay network
  - Lookup(key) -> identifier (e.g. IP address)

- 3 Questions from previous presentation
  1. CAN: How node joins
  2. Chord: How efficient Lookup is achieved by Finger Table
  3. Tapestry: Time complexity of Routing
CAN

- Keys hashed into d-dim space
- Maps keys to location
- Node stores a zone of hash table
- Key that maps (0.1,0.2) is stored at Zone B

Routing
- Choose the neighbor nearest to the destination
- A->B->E
CAN: Node joins

Bootstrap Node:
Node that provides initial configuration information to newly joining nodes

New node identifies a node already in CAN using Bootstrap node
CAN : Node joins

Pick a random point in space

new node
CAN : Node joins

Routes to \((x,y)\) from Node I and discovers node J

new node
CAN : Node joins

Split Node J’s zone in half, half of it is assigned to the new node
Chord: Protocol

- Chord Protocol
  - Assign each node and key an m-bit identifier
  - Node: hash the IP address
  - Key: hash the key
  - Identifiers are ordered on an identifier circle modulo $2^m$
  - Key k is assigned to $1^{st}$ node whose identifier is equal to/follows k.
    - Successor node

$m = 3$
### Chord – Simple Lookup

- **Simplest Chord Lookup (O(N))**
  - Each node only needs to know its successor on identifier circle
  - Query for identifier id is passed around identifier circle until successor(id) is found
  - Result returned along reverse path

```java
// ask node n to find the successor
// of id
n.find_successor(id)
if (id < (n, successor))
    return successor;
else
    // forward the query around the
    // circle
    return successor.find_successor(id);
```
Chord – Lookup using Finger Table

- Finger Table
  - Acceleration of Lookups
  - Each node has finger table with m entries
    - \( n.\text{finger}[i] = \text{successor}(n + 2^{i-1}) \)
Chord – Lookup using Finger Table

- Improved Lookup (Look up for id)
  - If id falls between n and successor(n), successor(n) is returned
  - Else, lookup is performed at node, which is the node in the finger table of n that most immediately precedes id
- Each node has finger entries at power of 2 intervals, each node can forward a query at halfway -> O(log N)
Tapestry: Routing

- Each Node maintains a Neighbor map
- Each entry in Neighbor map corresponds to a pointer to another node
- Path taken by a message originating from node 67943 and destined for node 34567.  
  - Decimal digits of length 5
- Routing takes approximately $\log_B N$ hop
  - a network of size $N$
  - Identifiers of base $B$ (e.g. $B=10$)
  - $O(\log N)$