2

Structured P2P networks

DHT - Distributed hash table

lookup(id)

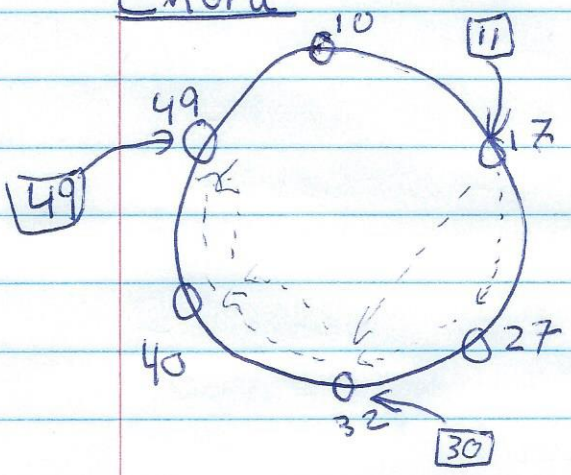
Assign id to
- Peers
- resources

Same id space

Assign resources to peers with close id

Problem is now: Find peer with id → then resource found

Chord



- Does not use locality
- Simple

Important: Good hash function needed

id = hash(ip:port)

Linear search = O(n)

Finger tables search = O(log n)

27+1	32
27+2	32
27+4	32
27+8	40
27+16	49

- Knows own neighbourhood well

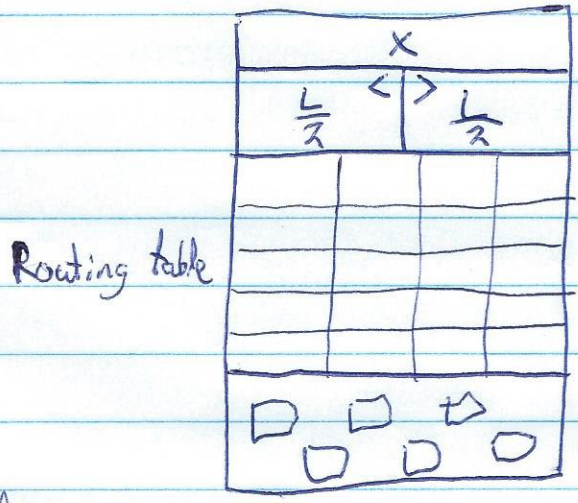
Joining:
- calculate hash
- Contact pred/succ
- move keys

Failures handled with successor list

Pastry

- API
  - route (msg, key)
  - deliver (msg, key)
  - forward (msg, key, nextid)

Uses locality information (own metric, assumes triangle inequality)



L = leaf set size. Configuration Parameter

Entries chosen with locality in mind

Row x has LCP ≥ x

Column x: next digit is x.

Neighbourhood set

Used to optimize routing table

Searching

O(log n) expected : worst case O(n)

- check it in leafset
- route to LCP (removes one digit)
- if unlucky, route to closest id (can lead to O(n))

Join

- Find lokal A (broadcasting)
- Z = lookup(x) via A
- Routing peers sends their state to A
- X state
  - leaf set from Z
  - Neighbourhood set from A
  - Routing table candidates from route - pick according to locality

Fail

- Checks liveness
  - Leaf dies → contact outermost leaf for new leaf
  - Routing entry dies → Contact peer in same row
    (First row in his routing table are candidates)
    → if no matches, use next row and contact peers
  - → Neighbour dies → contact other neighbour