One Hop Lookups for Peer-to-Peer Overlays

Anjali Gupta, Barbara Liskov, Rodrigo Rodrigues

Laboratory for Computer Science, MIT

Peer-to-Peer Systems

- Large scale dynamic network
- Overlay infrastructure :
    - Scalable
    - Self configuring
    - Fault tolerant
- Every node responsible for some objects
- Find node having desired object
- Challenge : Efficient Routing

Routing in Current P2P Systems

- Routing state size logarithmic in the number of overlay nodes
- Membership changes frequent
- Small routing table → Less bookkeeping
- Logarithmic overlay hops per lookup, high latency
- Amortized cost unacceptable for storing and retrieving small objects

Our Thesis

It is feasible to :

- Keep full routing state on every node
- Route in one overlay hop
- Achieve high lookup success rate

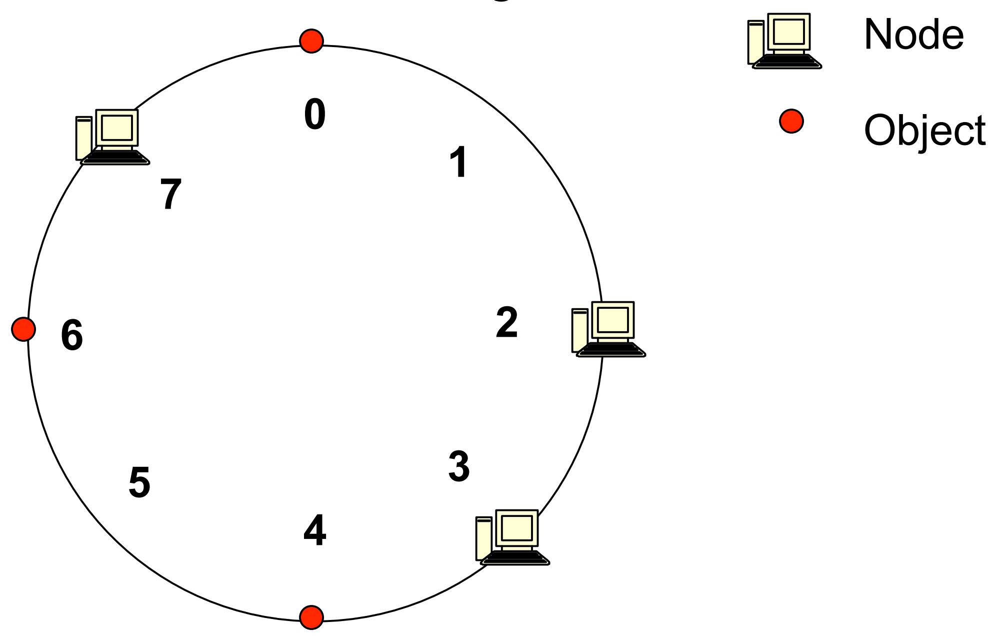
Thus we can :

- Enable a large class of applications for peer-to-peer systems

Outline

- Structured Peer-to-Peer Overlay
- Goals
- Design
- Analysis
- Future Work

Structured Overlay



Objects and nodes have identifiers

A specific node is responsible for each object

Different data placement algorithms, e.g., consistent hashing

Successor node

Dynamic Membership

- Nodes join and leave
- Gnutella study (Saroiu et al) :
  average node session time 2.9 hours
- Rate of change   Number of nodes
- For 100,000 nodes, approximately 20 membership changes per second

Goals

Routing Goal

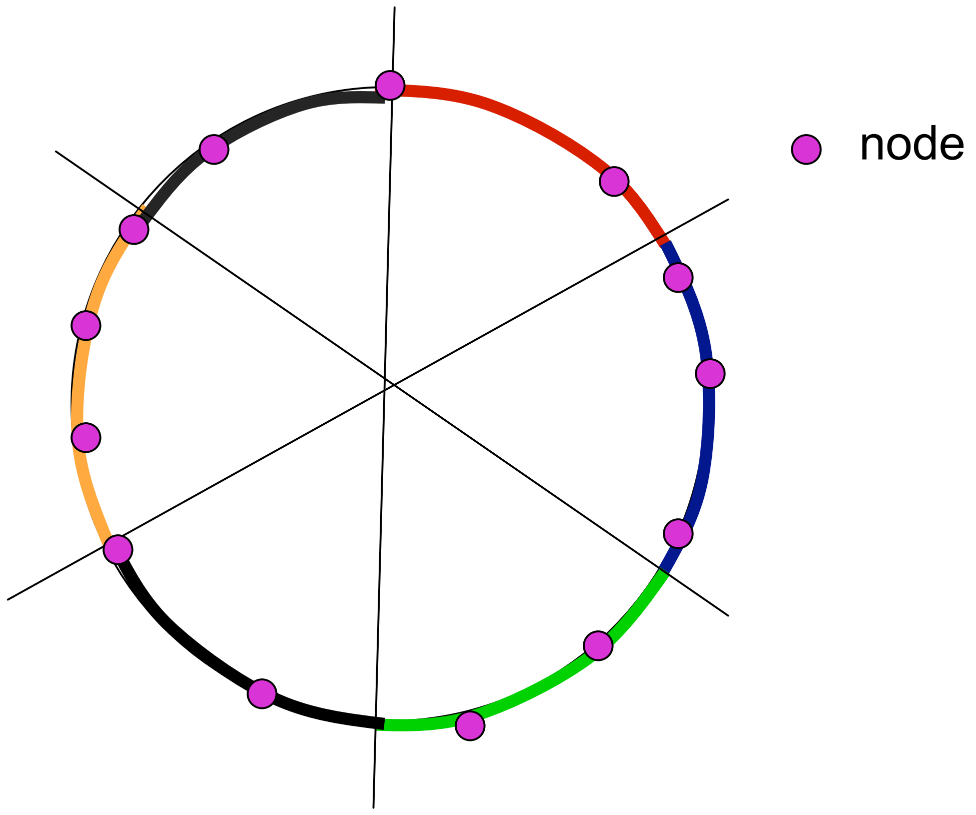
- How does node N find object O?
  - Finds successor S(O) by looking in its own table
  - Sends a message to S(O)
  - If S(O) is current successor of O, responds with success message
- Lookup success:
  - Object found in first attempt
- Achieve high lookup success rate, e.g., 99%

Design Goals

- Information about a node joining or leaving the system must reach all nodes rapidly
- Bandwidth requirement should be feasible
- Should be scalable
- Hierarchical Scheme!

Design

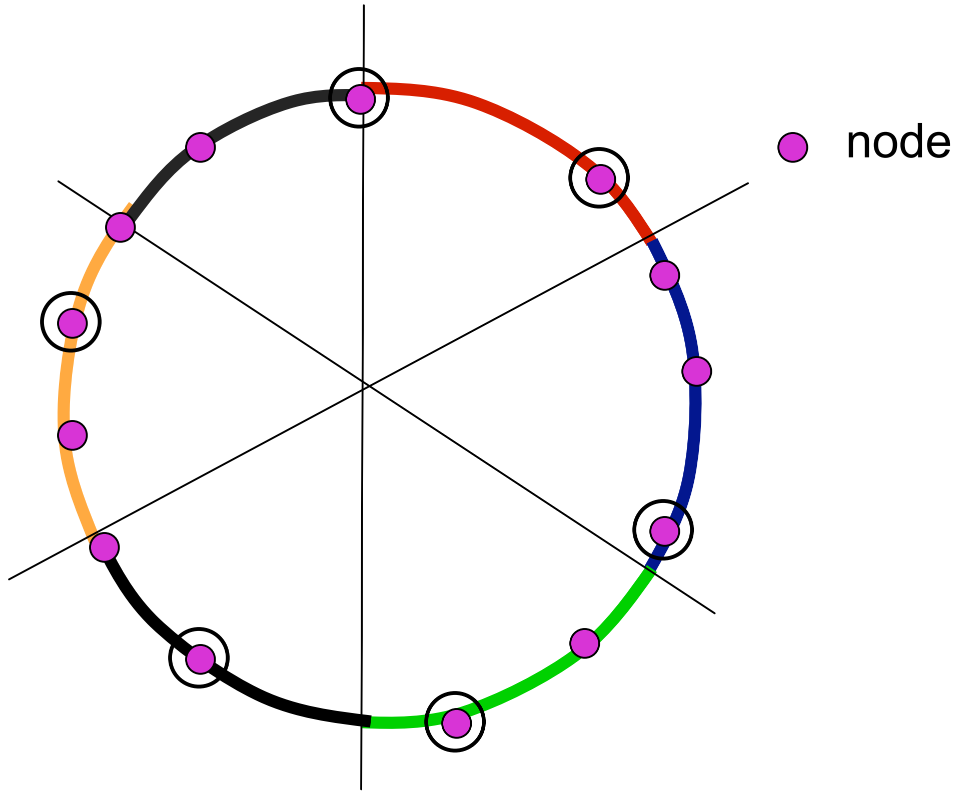
Hierarchical Scheme



Ring is divided statically into slices

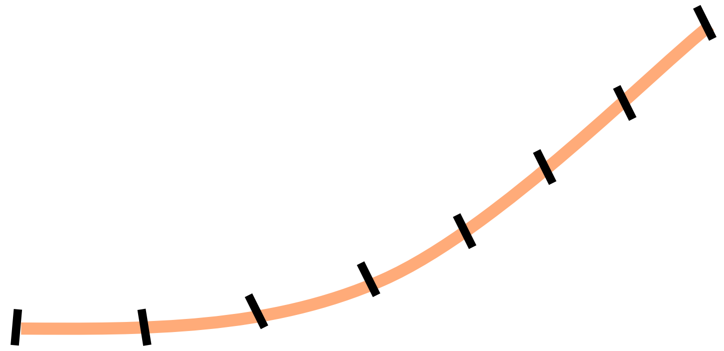
(i.e., slices are just identifier intervals)

Hierarchical Scheme



Successor of midpoint of slice is slice leader

Hierarchical Scheme



Each slice is divided statically into units

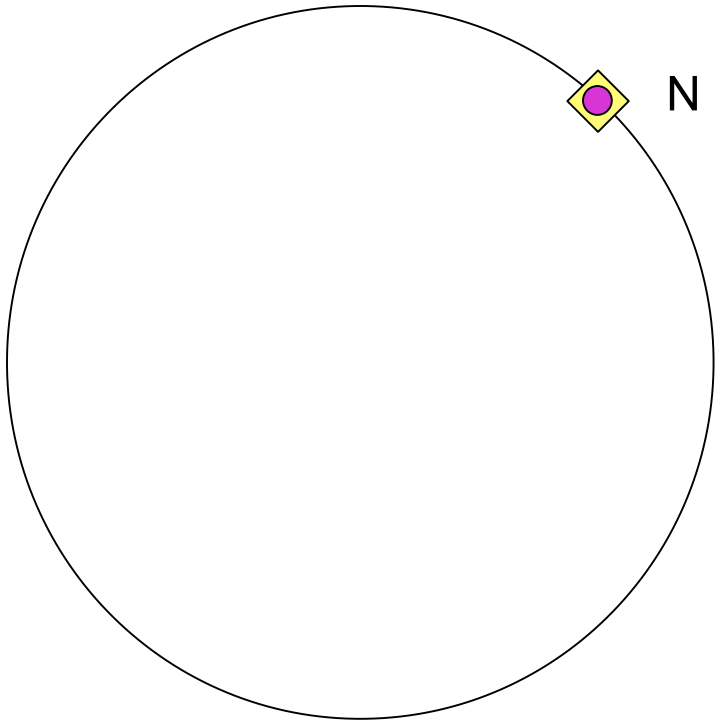
Again, units are just intervals

Successor of midpoint of unit is unit leader

Base Communication

- Node exchanges frequent keep-alive messages with predecessor and successor
- Detects change in successor : event
- Recent event log
- Piggyback event log on keep-alive messages
- Is this fast enough?

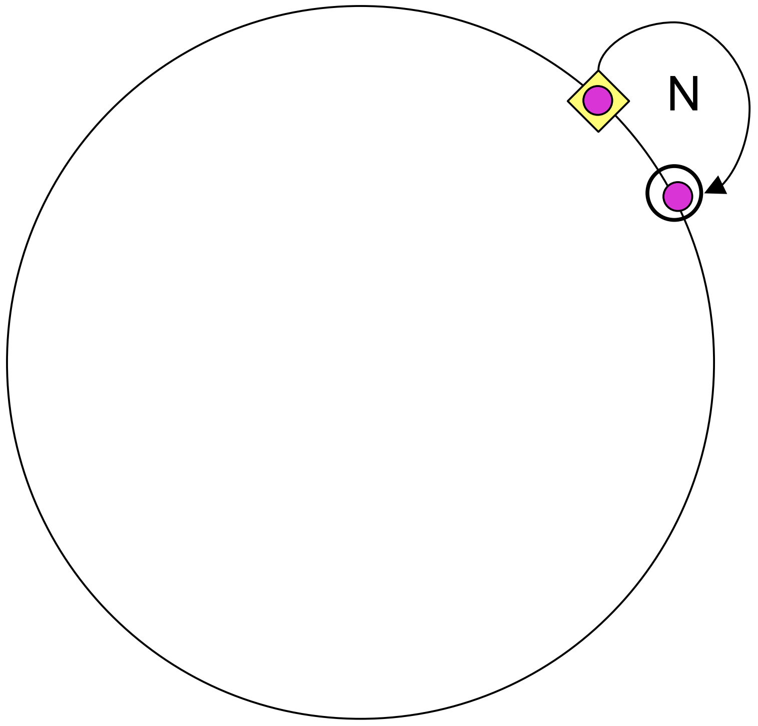
Flow of Information : Inter-slice



Step 1: Event detected by node N

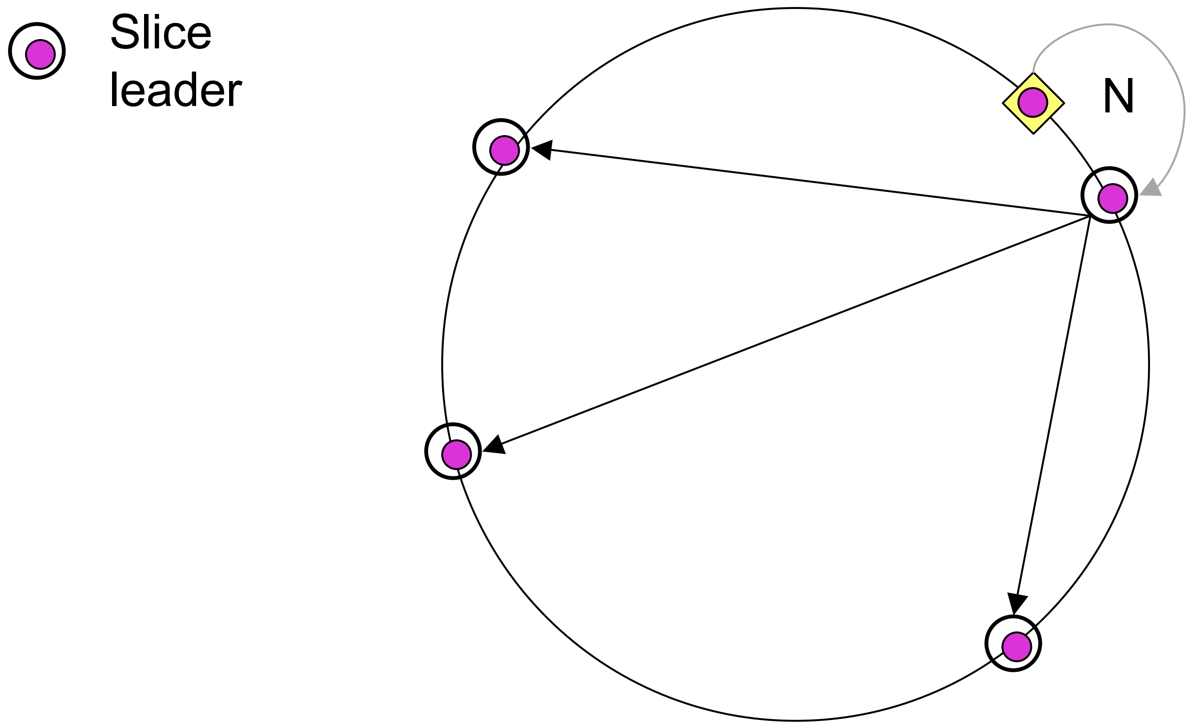
Flow of Information : Inter-slice

◉ Slice leader



Step 2: N notifies its slice leader

Flow of Information : Inter-slice

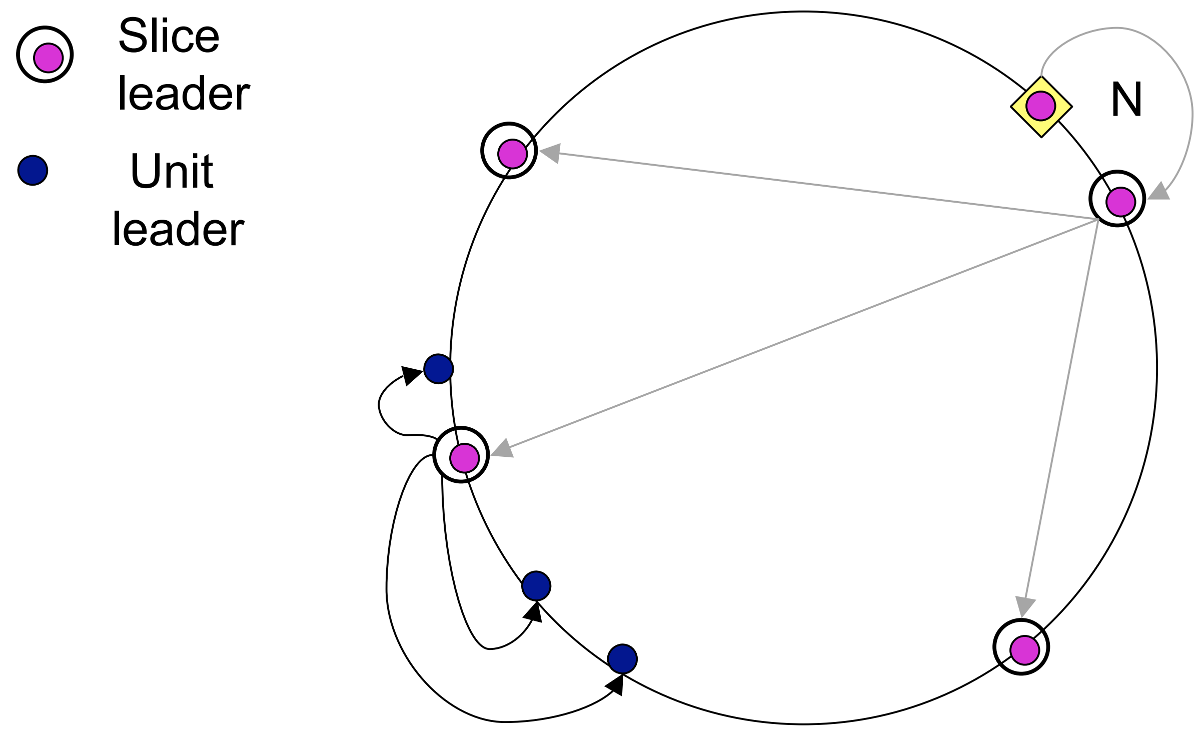


Step 3: N’s slice leader collects events for some time, then notifies other slice leaders

Is this fast enough?

- Slices may be large!
- Piggybacking on keep-alive messages can take a long time

Flow of Information : Intra-slice



Step 4: Slice leaders notify their respective unit leaders periodically

Analysis

Speed of Propagation

- Units kept small to spread information quickly
- Possible frequency of communications :
  - Slice leaders communicate with each other once every 30 seconds
  - Slice leader communicates with its unit leaders once every 5 seconds
  - Nodes exchange keep-alive messages every second

Speed of Propagation

- If (expected) unit size is 20, the time taken to reach farthest nodes is:
  - Node to Slice leader : 0 s
  - Slice leader to other slice leaders : 30 s
  - Slice leader to unit leaders : 5 s
  - Unit leader to edge of unit : 10 s
- Total time : 45 seconds!

Time Constraint

- Recall : Goal of 99% lookup success rate
- If f : acceptable lookup failure rate

  n : number of overlay nodes

  r : rate of membership change
- Then: All nodes must be notified about an event within

$$\frac{f \cdot n}{r} \sec$$

- e.g.: For f = 0.01, n = 100000, r = 20 changes/second

  time interval = 50 seconds

Bandwidth Use

1. A slice leader tells another slice leader only about events in its own slice
2. A slice leader never sends the same notification twice to any slice leader
3. Event notification is sent by a node to its neighbor only if :
    1. The neighbor is in the same unit
    2. It has not already sent this notification to this neighbor

Bandwidth Use

- Number of slices (k) chosen to minimize total bandwidth use. We prove that

$$k \quad \sqrt{n}$$

- For an overlay having 100,000 nodes:
  - Per node: 4 kbps (up, down)
  - Per unit leader: 4 kbps (up, down)
  - Per slice leader: 240 kbps (upstream), 4 kbps (down)

Slice Leaders

- Important to choose slice leaders correctly
- Super-node scheme
- Sufficient number of well-provisioned nodes
- Incentives for maintaining ring: Application dependent

Fault Tolerance

- Unit leader failure : easy recovery
- Slice leader failure
    - Contact unit leaders
    - Contact other slice leaders

Related Work

- Design of a Robust P2P system
  - R. Rodrigues et al
- Kelips
  - I. Gupta et al
- Controlling the Cost of Reliability in P2P Overlays
  - R. Mahajan et al

Future Work

- System implementation almost complete
- Experiments to see actual system behavior
- Systems larger than a million nodes
  - Two hop scheme