Bamboo
– An Open Source DHT

Sean Rhea, Dennis Geels, Timothy Roscoe,
and John Kubiatowicz
University of California,
Berkeley and Intel Research, Berkeley

Presenter : Saranya Thirumoolan
Course : Peer to Peer Networks [22c:196]

Agenda

- Introduction
- What is Bamboo ?
- Features
- Reactive vs Periodic Recovery
- Message Time out Calculation
- Proximity Neighbor Selection
- Future Work and References

Introduction

- DHT – Distributed Hash Table
- Maps a set of identifiers to a set of nodes
- Helps in locating rare objects better than in unstructured networks
- DHT Known for :
  - Incremental scalability in No. of Nodes
  - Extremely high availability
  - Low Latency
  - High Throughput

Introduction

- Problems in DHT
- Cannot perform well under high churn rates
- Reasons
  - Reactive vs Periodic Recovery
  - Message Timeout Calculation
  - Proximity Neighbor Selection
- Short session time – affects performance
- Increase latency can partition network

What is Bamboo ?

- Open Source DHT
- Implements algorithms of DHT, Pastry, Chord
- Modifies existing algorithms, in order to handle churn better.
- Java based
- Free Code available for download
- Documentation and support available

What is Bamboo ?

- Implemented in ModNet
- ModNet
  - In-network Queuing
  - Cross traffic
  - Packet loss
- Implemented under 1 network topology and a simple churn

Agenda

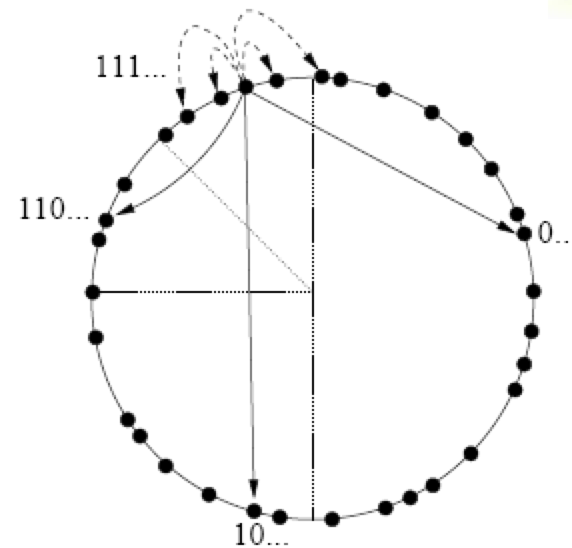
- Introduction -
- What is Bamboo ? -
- Features
- Reactive vs Periodic Recovery
- Message Time out Calculation
- Proximity Neighbor Selection
- Future Work and References

Features

- Geometry and Routing ϵ Pastry
- Geometry = Pattern in which the neighbors are connected in the overlay network
- Difference from Pastry – maintains the same geometry in spite of churn
- 2 sets per node
  - Leaf set – Set of 2k neighbors before and after the main node
  - Identifiers of nodes which share the longest successive prefix positions with same digit.

Features – Continued

- Node 01267 and 22267 have no digits in common
- Node 01267 and 01345 have 2 digits in common and l = 2
- R_l [i]
  - where l = column
  - [i] = the row.



	0	1	2	3
0				
1	22267			
2			01345	
3				

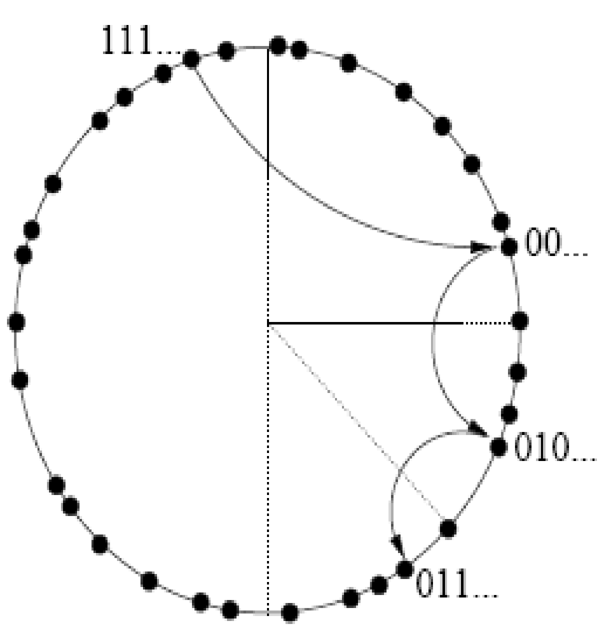
Features - Routing

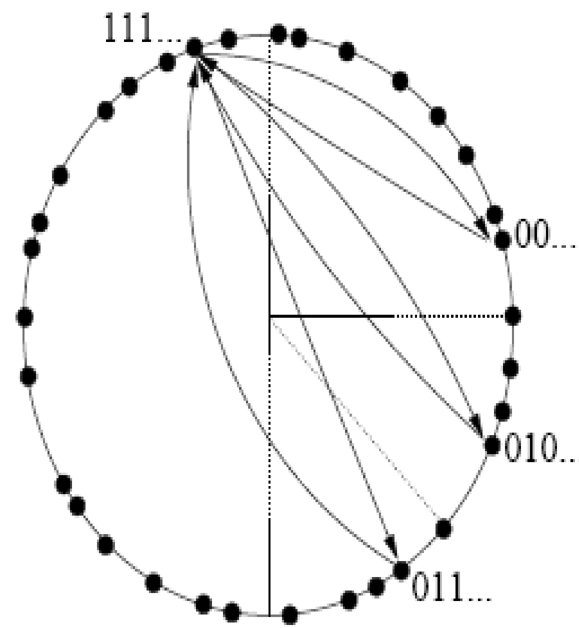
$$
\begin{array}{l}
\textbf{if } (L_{-k} \le D \le L_k) \\
\quad \text{next\_hop} = L_i \text{ s.t. } |D - L_i| \text{ is minimal} \\
\textbf{else if } (R_l[D[l]] \ne \textbf{null}) \\
\quad \text{next\_hop} = R_l[D[l]] \\
\textbf{else} \\
\quad \text{next\_hop} = L_i \text{ s.t. } |D - L_i| \text{ is minimal}
\end{array}
$$

- O(Log N) Steps
- Even if 30% of links are broken, there are still connected paths between all nodes in a network of 65,536 nodes

Features - Routing

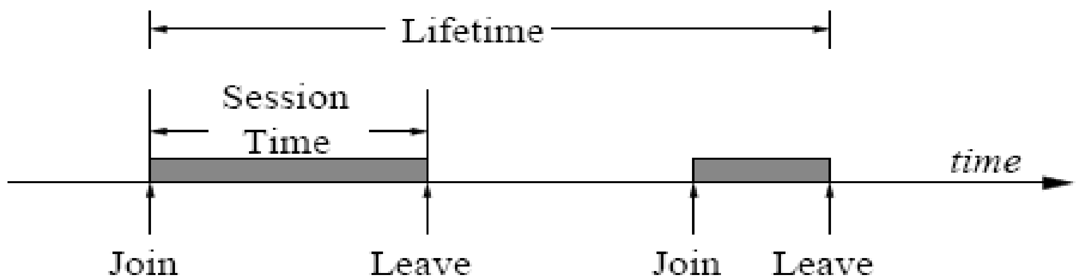
- Recursive Routing & Iterative Routing





Features - Continued

- High degrees of Churn



- Availability of a node = (Sum of Session times) / Lifetime
- Failure if
  - The node mentioned is not available
  - An intermediate node fails before forwarding the request

Agenda

- Introduction -
- What is Bamboo ? -
- Features -
- Reactive vs Periodic Recovery
- Message Time out Calculation
- Proximity Neighbor Selection
- Future Work and References

Reactive vs Periodic Recovery

- Reactive Recovery
  - For every node appearing and disappearing, send the differences in the leaf set to all the nodes in the leaf set. O(k^2)
- Periodic Recovery
  - The entire node set is shared with one of the nodes in the leaf set picked at random. O(Log k)
  - Periodic and Bandwidth saving
  - Currently used in Chord / Bamboo

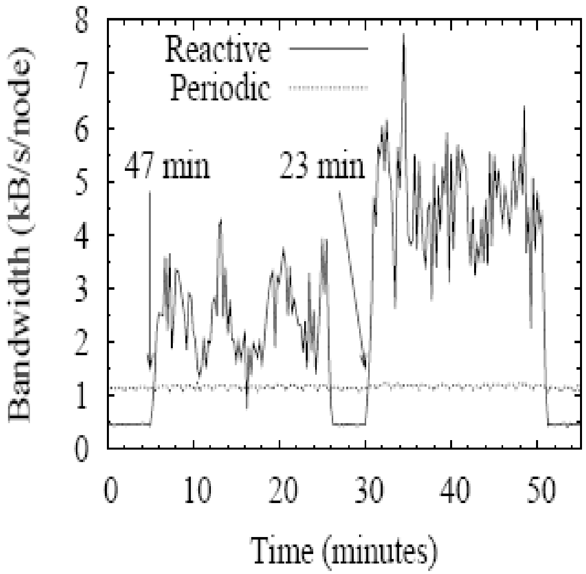
Reactive vs Periodic Recovery

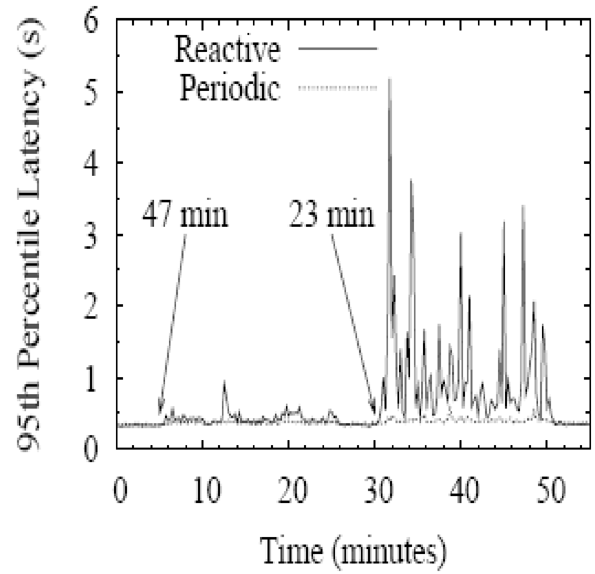
- Positive Feedback cycle
  - When a node does not get answered due to its congestion, but thinks the neighbors have failed.
- Dissipate failure detection and recovery to avoid positive feedback cycle
- Conclude failure after 15 consecutive timeouts
- Disadv – Might lead to existence of failure nodes
- Stop routing to a neighbour after 5 timeouts

Reactive vs Periodic Recovery

- A node C sends msgs to B, until A which is in between joins.
- A starts getting noticed.
- Even if A fails, C knows B is backup.
- Under low churn
  - Reactive recovery is good
  - Periodic is waste
- Under high churn
  - Reactive recovery not effective
  - Periodic is good

Reactive vs Periodic Results





Agenda

- Introduction -
- What is Bamboo ? -
- Features -
- Reactive vs Periodic Recovery -
- Message Time out Calculation
- Proximity Neighbor Selection
- Future Work and References

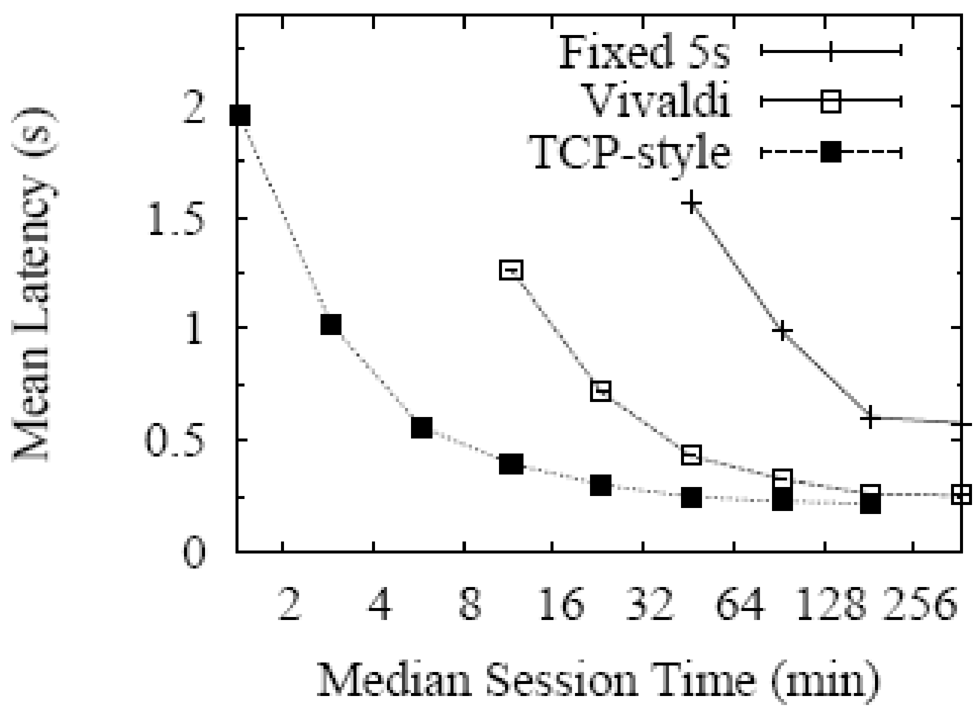
Message Timeout Calculation

- Timeout value must be selected accurate
- Lower = Query might not have reached
- Higher = unnecessary waiting
- 3 Techniques
  - TCP Style
  - Virtual co-ordinates
  - Fixed 5 seconds

Message Timeout Calculation

- TCP Style
  - Recursive Looping – hardly any contact nodes
  - Log N nodes, pinged for availability
  - Stored in history
  - RTO = AVG (observed avg RTT) + 4 * VAR (variance)
- Virtual Co-ordinates
  - Iterative Routing
  - Timeout α Distance between virtual co-ordinates
  - RTO = v (predicted) + 6 * α (avg error) + 15
- Fixed 5 Seconds

Message Time Calculation



Agenda

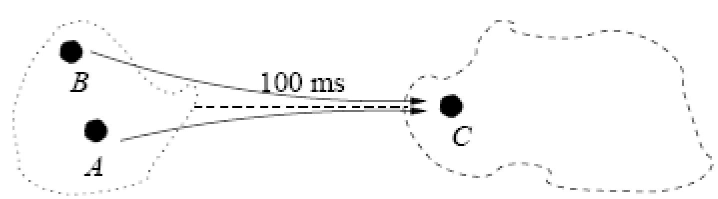
- Introduction -
- What is Bamboo ? -
- Features -
- Reactive vs Periodic Recovery -
- Message Time out Calculation -
- Proximity Neighbor Selection
- Future Work and References

Proximity Neighbor Selection

- Choosing the closest node among all the potential nodes for a routing table entry
- If Leaf set is perfect = O(N)
- If Routing Table is perfect = O(Log N)
- Churn – Re-run the algorithms to find out the closest node
  - Find the closest node
  - Find the Latency
  - If new node is not already present in Routing table
    - Add
  - Else
    - Check with latency of existing node and replace if less.

PNS - Techniques

- Global Sampling / Global Tuning
  - Node of same # of prefix will be chosen at random.
  - All selected nodes probed.
  - Disadvantage :
    - 2 far of neighbors will take a lot of time to discover

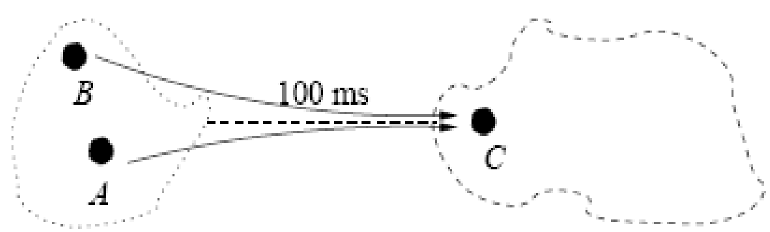


PNS - Techniques

- Neighbor’s Neighbors
  - Ping all L level neighbors of all the nodes at level L.
  - Pick the one with least latency
  - A → B, b → C, A → C
  - Use this technique to walk through the graph
  - 2 Far off neighbors will not be liked by anyone

PNS - Techniques

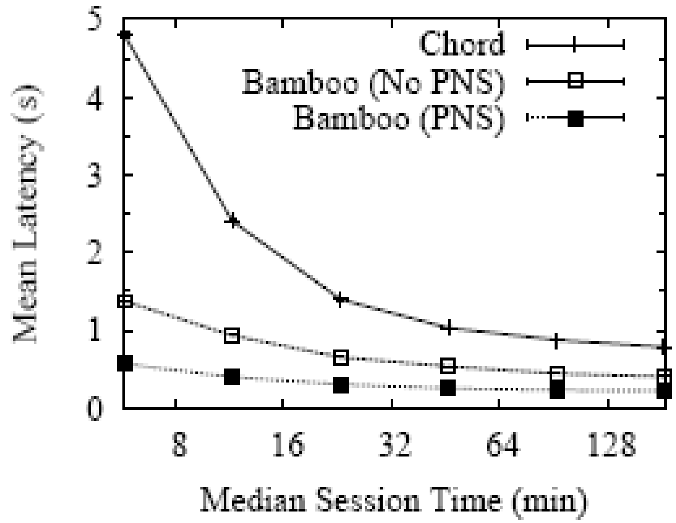
- Inverse Neighbor
  - Probe nodes which have our nodes as neighbors
  - A → C, B → C, means A & B are neighbors.
  - Helps to discover close by neighbors in the following case

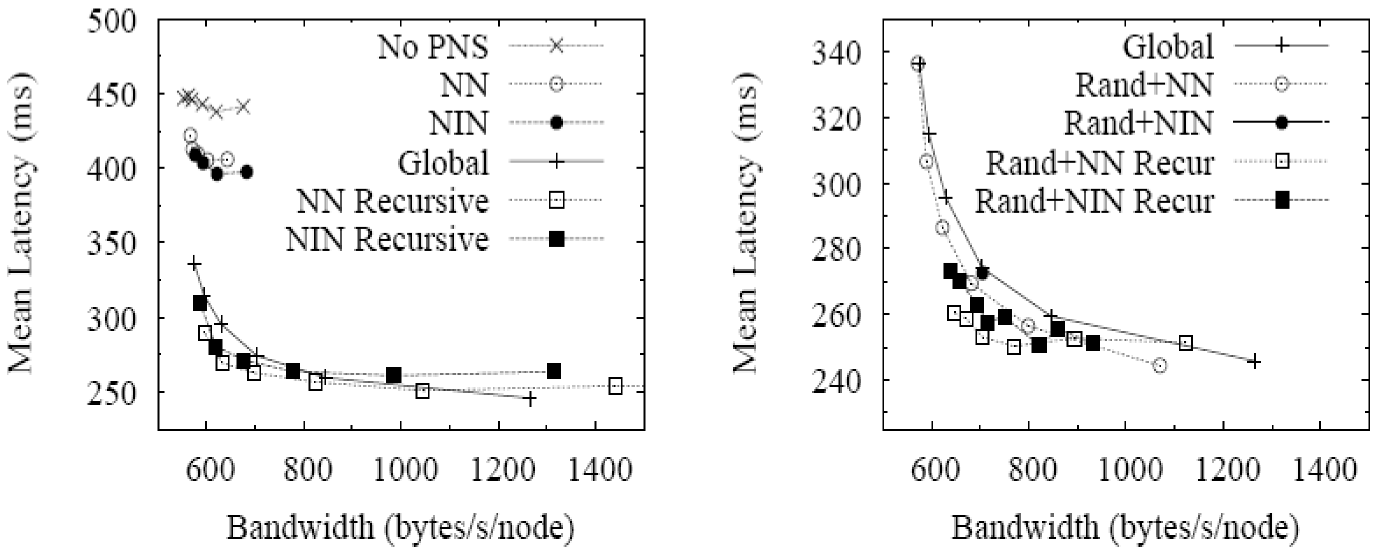


PNS - Techniques

- Recursive Sampling
  - A node in the highest L level taken
  - Its inverse neighbors are pinged and all but k-closest are discarded
  - Repeat this for all values of L recursively
  - These new nodes are potential nodes for the routing table.
  - Only 3 messages at a time.

PNS - Results





(a) (b)

Agenda

- Introduction -
- What is Bamboo ? -
- Features -
- Reactive vs Periodic Recovery -
- Message Time out Calculation -
- Proximity Neighbor Selection -
- Future Work & References

Future Works

- Alternate routing table algorithms
- Compare Iterative and Recursive routing, since current implementation focuses on recursive routing.
- Other network topologies and churn levels
- Churn rates, more natural, from observed values
- Security and Anonymity

References

- http://www.bamboo-dht.org/
- Handling Churn in DHT - Sean Rhea, Dennis Geels, Timothy Roscoe, and John Kubiatowicz University of California, Berkeley and Intel Research, Berkeley

Agenda

- Introduction -
- What is Bamboo ? -
- Features -
- Reactive vs Periodic Recovery -
- Message Time out Calculation -
- Proximity Neighbor Selection -
- Future Work & References –

Questions ?