CAN and Tapestry

Awalin Shopan
CMSC 818

Overview

• CAN is a distributed system that maps keys onto values
• Uses d-dimensional space for key hashing
• Interface:
  – insert(key, value)
  – retrieve(key)

Two dimensional case

In this 2 dimensional space a key is mapped to a point (x,y)
1) Discover some node "I" already in CAN

2) Pick random point in space

Routing

Construction
3) I routes to (x,y), discovers node J

4) split J’s zone in half... new owns one half

Insertion affects only $O(d)$ other nodes

- Use zone takeover in case of failure or leaving of a node
- Nodes send neighbors periodic update msg at discrete time interval $t$
  - Along with its zone coordinate and neighbour list
- If a node does not send alive msg in time $t$, its neighbour with the smallest zone size takes over its zone. The neighbours negotiate with TAKEOVER msg.
Zone reassignment: node 2 leaves

Uniform Partitioning
- Instead of splitting directly splitting the node occupant node
  - Compare the volume of its zone with neighbors
  - The one to split is the one having biggest volume

Zoning

Partition tree

TAPESTRY
- Structured p2p overlay
- CAN and CHORD do not exploit locality feature.
- DOLR: decentralized object location and routing
  - Route to node or object replicas.
Mapping

Nodes are assigned NodeID.
Different application can coexist in a node. So msg also contains Application ID.
Application specific end-points have Globally Unique ID.
Objects have Object ID.

Basic Routing

• Each Identifier has its **ROOT**.
• Node N is the root of Identifier G if its NodeID=Identifier G.
  – If exact match not found, use Surrogate [closest one in ID space]
• Routing table: contains list of <node ID, IP> . Of its neighbours.
• NodeIDs match larger prefixes with the Identifier as routing proceeds.

DOLR API

• **Publish Object**: make an object available on the local node.
• **UnpublishObject**: remove its location mapping.
• **RouteToObject**: route a msg to a location of the object with given ID.
• **RouteToNode**: Route msg to the application (with given id ) running on the exact node.

Neighbor Map

Node **325AE**

<table>
<thead>
<tr>
<th>Level</th>
<th>1</th>
<th>2</th>
<th>9</th>
<th>3259*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level 1</td>
<td>1****</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Level 2</td>
<td>31***</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Level 3</td>
<td>321**</td>
<td>322**</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Level 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Publish Object
• Publishing object location pointers throughout the network facilitate efficient routing.
Resiliency

- Soft state based republishing
- Fault tolerance: nodes keep secondary links, if the primary fails, make the secondary as primary and take another as secondary
- Nodes use periodic beacons:
  - Detect outgoing link failure
  - Detect node failure
Thank you