GSCI 1380: Day 10
Today
Tapestry
Conclusion
Time
* why real time is bad
* logical clock (used in YugedB/CockroachDB)
* Vector clocks (used in Amazon Dynamo)

Better Choice

\[(2-3) \mod 4 \rightarrow (-1) \mod 4 \rightarrow 3\]

Assumption: 112, 120, 333

Node = 110

Positions where ??? on go in?
Add a new Node (111)

1. Find Root (112)
2. Build route Table
3. Inform "next-to-know" nodes working tapestry that you've joined the network (closer)
4. Optimize route Table fast tapestry
5. Inform "next-to-know" nodes that you've joined the network

"Need to know" → a current node in the network with a missing cell that this new node fills in.

Prefix = (111, 112) (2)

Table 120

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table for 333

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The only place 111 can go is here but root also meets the criterion and would likely go there first.
(1) Find root

(2) Get prefix

(3) Get route table as initial seed for table

(4) Acknowledge Multicast()

for p = prefix; p ≥ 0; p -=

for each node in nodes{
  back-pointer.push(node, get-back-pointers(p))
}

local table.add(back-pointer);

nodes = merge(nodes, back-pointer)
This is a recursive call from node to node where level changes between lookups.

If the lookup happens at a row (say 1) at Node A but there is only one entry in the row, i.e., Node A is the only entry then you have two options:

1. Recursively call Node A & go down a row.
2. Simply go down a row.
<table>
<thead>
<tr>
<th># of servers</th>
<th>Consistent hash</th>
<th>Chord</th>
<th>Tapestry</th>
</tr>
</thead>
<tbody>
<tr>
<td>uses</td>
<td>LB</td>
<td>$\log_2(N)$</td>
<td>$B(\log_b N)$</td>
</tr>
<tr>
<td>table-size</td>
<td>$O(1)$</td>
<td>$O(\log_2(N))$</td>
<td>$\Theta(\log_b N)$</td>
</tr>
<tr>
<td># of servers</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Azure's Service Fabric

for storage of microservice data.

first server is random

(frontend)

Websave

Storage

"Chord"
Time

NTP

S

C

T - AD

S2

y'

del ($500)

add ($500)

12:00

12:01

5 minutes behind

11:58
Conclusion

1. Discussed common problems
   a. Closers
   b. findNextHop
   c. Join( )

2. Compared different consistent hash implementations

3. Started a discussion on Time