DHT Geometries

Tim Smith (tsmith@cs)

University of Toronto
Department of Computer Science

2007/10/18
Gummadi's Questions

- How does DHT geometry/flexibility affect:
  - static resilience
  - path latency
  - local convergence?
Gummadi's Analysis

- Flexibility comparison:

<table>
<thead>
<tr>
<th>property</th>
<th>tree</th>
<th>hypercube</th>
<th>ring</th>
<th>butterfly</th>
<th>xor</th>
<th>hybrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neighbor Selection</td>
<td>$n^{\log n/2}$</td>
<td>1</td>
<td>$n^{\log n/2}$</td>
<td>1</td>
<td>$n^{\log n/2}$</td>
<td>$n^{\log n/2}$</td>
</tr>
<tr>
<td>Route Selection (optimal paths)</td>
<td>1</td>
<td>$c_1(\log n)$</td>
<td>$c_1(\log n)$</td>
<td>1</td>
<td>1</td>
<td>$c_2(\log n)$</td>
</tr>
<tr>
<td>Route Selection (non-optimal paths)</td>
<td>-</td>
<td>-</td>
<td>$2c_2(\log n)$</td>
<td>-</td>
<td>$c_2(\log n)$</td>
<td>1</td>
</tr>
<tr>
<td>Natural support for sequential neighbors?</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>Default routing: no, Fallback routing: yes</td>
</tr>
</tbody>
</table>

- Ring looks pretty good.
Gummadi's Results

• How do DHT geometries affect static resilience?
  – Failed hosts vs. failed paths:
Gummadi's Results

- How do DHT geometries affect static resilience?
  - Failed hosts vs. “Path stretch”
Gummadi's Results

- Idea: ring geometry stands up well because it keeps track of sequential neighbors.
- What if we add sequential neighbors to other geometries?
  - What if we add more sequential neighbors to the ring geometry?
Gummadi's Results

• 16 Sequential neighbors
  – Failed hosts vs. failed paths:
Gummadi's Results

• 16 Sequential neighbors
  – Failed hosts vs. “path stretch”:

  – Note the increased range – ring paths now up to 160% longer
Gummadi's Results

• How do DHT geometries affect static resilience?
  – The ring geometry outperforms all others.
  – Support for sequential neighbors increases static resilience, especially with the ring geometry.
Gummadi's Results

How do DHT geometries affect path latency?

- Two ways to reduce path latency:
  - Proximity Neighbor Selection (PNS)
    - Choose neighbors based on proximity (as measured by ping time)
  - Proximity Route Selection (PRS)
    - Choose next hop based on proximity
    - Neighbors chosen arbitrarily, according to identifier ranges (ring), bit settings (XOR, Tree, etc)
Gummadi's Results

- Aside: how do we find our nearest neighbors?
Gummadi's Results

• Aside: how do we find our nearest neighbors?
  – Ideally, for each neighbor, choose neighbor in selection range with lowest latency.
    • What's the problem with this?
Gummadi's Results

• Aside: how do we find our nearest neighbors?
  – Ideally, for each neighbor, choose neighbor in selection range with lowest latency.
  – Problem: this means we will ping everyone in the DHT.
Gummadi's Results

• Aside: how do we find our nearest neighbors?
  – In reality, we will sample some number $K$ neighbors at random, and pick the one with the lowest latency.
  – Gummadi chooses $K = 16$ here.
Gummadi's Results

- How do DHT geometries affect path latency?
  - With ideal PNS:
    - PNS very close to Internet-speed routing!
    - PRS not so much!

- PNS very close to Internet-speed routing!
- PRS not so much!
Gummadi's Results

• How do DHT geometries affect path latency?
  – With PNS(16) (i.e. $K = 16$):

• PNS(16) still works very well!
Gummadi's Results

• How do DHT geometries affect path latency?
  - PRS provides some improvement over arbitrary/fixed neighbor selection
  - Ideal PNS provides roughly Internet-speed routing
  - PNS(16) is a good approximation of ideal PNS
  - PNS(16) + PRS provides only a small improvement over PNS(16)
Gummadi's Results

● Why is PNS so much better than PRS?
Gummadi's Results

• Why is PNS so much better than PRS?
  – Again, it is a matter of *flexibility*.
  – PNS can pick from \([2^i, 2^{i+1}]\) nodes when selecting neighbor \(i\) (with the next-hop chosen deterministically).
  – PRS can only pick from its first \(i\) neighbors when choosing the next hop (with all neighbors chosen deterministically).
  – Thus PNS can select from \(2^i\) nodes, PRS only \(i\)
Gummadi's Results

- How do DHT geometries affect local convergence?
  - Measured by number of exit points from "isolated domains" - domains of nodes with low latency to each other, but large latency from the network in general
  - The more exit points, the more times this "high latency boundary" has been crossed
  - Crossing the boundary is not good!
Gummadi's Results

• How do DHT geometries affect local convergence?
  – Isolated domain size vs. # of exit points:

  ![Graph showing isolated domain size vs. number of exit points](image)

• PNS is looking good again!
  – But we need to use PNS(16)...
Gummadi's Results

• How do DHT geometries affect local convergence?
  - Isolated domain size vs. # of exit points:

• PNS(16) doesn't look so hot now
  - Maybe we really need PNS(16)+PRS after all.
Gummadi's Results

- How do DHT geometries affect local convergence?
  - PRS alone is not enough
  - Ideal PNS is ideal!
  - PNS(16) is as bad as PRS
  - PNS(16)+PRS is ideal for isolated domains > 4096 nodes
Gummadi's Results

- The constraints a DHT geometry puts on the design of its algorithms affects flexibility.
Gummadi's Results

- The constraints a DHT geometry puts on the design of its algorithms affects flexibility.
- Flexibility in neighbor and route selection is important for static resilience, path latency, and local convergence.
Gummadi's Results

- The constraints a DHT geometry puts on the design of its algorithms affects flexibility.
- Flexibility in neighbor and route selection is important for static resilience, path latency, and local convergence.
- Some DHTs are *inflexible* – hypercube, tree, butterfly, and “hybrid”. 
Gummadi's Results

• The constraints a DHT geometry puts on the design of its algorithms affects flexibility.

• Flexibility in neighbor and route selection is important for static resilience, path latency, and local convergence.

• Some DHTs are *inflexible* – hypercube, tree, butterfly, and “hybrid”.

• Ring and XOR are flexible – they allow implementation of both PNS and PRS.
More Questions

• What should $K$ be set to?
• How else can we improve path latency?
• How can we improve throughput?
Dabek's Results

- What should $K$ be set to?
  - $K$ vs. lookup latency:
Dabek's Results

• What should $K$ be set to?
  – $K$ vs. lookup latency:

• Not much benefit after $K = 20$
Dabek's Results

• How else can we improve path latency?
  – Lookup latency with iteration vs. recursion:
Dabek's Results

• How else can we improve path latency?
  – Lookup latency with iteration vs. recursion:

    ![Graph showing cumulative probability over latency for iterative and recursive methods.]

• Recurse!
Dabek's Results

• How else can we improve path latency?
  – “Recursion [eliminates latency by] immediately forwarding lookups before acknowledging the previous hop.”
Dabek's Results

• How can we improve throughput?
  – Idea: TCP is holding us back.
  – Replace TCP with a custom transport:

![Graph showing cumulative probability against throughput (KB/second)]
Discussion

- Any questions?
Discussion

• Why don't we just use ring for everything?
Discussion

- Is path latency more important than path bandwidth?
  - How would path bandwidth be optimized?
Discussion

- What were the desirable design characteristics identified?
  - Do we have heuristics now, or just fuzzy words like “flexibility” and “geometry”??
Discussion

• Do “inflexible” geometries have any saving graces?
  – I.e. are there any cases in which they are desirable?