Examining The Tradeoffs Of Structured OVERLAYS In A Dynamic Non-transitive Network

Steve Gerding  Jeremy Stribling
{sgerding, strib}@csail.mit.edu
Motivation

• P2P overlays are a hot topic in networking research
• However, overlay performance research is still young
• Relatively unexplored areas:
  • Comparing several overlays in a fair setting, with a unified metric
  • Examining their behavior under real, pathological conditions
  • Determining how parameter tuning affects performance
• Important for system designers and wide area deployment
Our Goal

- Compare the performance of several structured P2P overlays under real world network conditions
- Explore the effects of parameter tuning for individual overlays

Accomplished by:
- Gathering and analyzing data about real world network conditions
- Using this data to compare the overlays in simulation
- Analyzing the simulation results and drawing conclusions
Presentation Overview

- Related work
- Real world dataset: PlanetLab
- Overlays in brief: Chord, Tapestry, Kademlia, Kelips
- Experimental methodology
- Results
- Discussion
- Future work
Related Work

- Gummadi et. al.: Effect of routing geometry on resilience, proximity
  *The impact of DHT routing geometry on resilience and proximity, SIGCOMM 2003*

- Rhea et. al.: App-level bmarks to encourage quality implementations
  *Structured peer-to-peer overlays need application-driven benchmarks, IPTPS 2003*

- Liben-Nowell et. al.: Chord stabilization traffic, with churn
  *Analysis of the evolution of peer-to-peer systems, PODC 2002*

- Xu: Routing state vs. network diameter: log(n) asymptotically optimal
  *On the fundamental tradeoffs between routing table size and network diameter, Infocom 2003*

- Countless structured and unstructured P2P overlays
The PlanetLab Dataset

- Topology data obtained from the PlanetLab federated testbed
- Extracted from PlanetLab All-Pairs-Pings data (http://pdos.lcs.mit.edu/~strib/pl_app)

- Why is this interesting?
  - Global-scale testbed
  - Non-transitive links
  - Time-varying latency data
  - Real-world rates of churn (node failure and recovery)
The PlanetLab Dataset

Observed properties of the PlanetLab testbed:

- Size of datasets
  Fully-connected: 159
  Non-transitive: 248

- Non-transitivity
  9.9% of combinations are non-transitive

- Mean round trip time
  Fully-connected: 117.39 ms
  Non-transitive: 118.46 ms

- Churn rate
  MTTF: 321.1 hours
  MTTR: 2.7 hours

(Blind submission, SIGMETRICS 2004)
# Overlays

<table>
<thead>
<tr>
<th>Chord</th>
<th>Tapestry</th>
<th>Kademlia</th>
<th>Kelips</th>
</tr>
</thead>
</table>

[Image of a network diagram]
Properties of Chord (Stoica et. al., SIGCOMM 2001):

- Ring/Skiplist geometry
- Separates correctness (successors) and performance (finger table)
- $\log(n)$ state, $\log(n)$ hops

Parameters Explored:

<table>
<thead>
<tr>
<th></th>
<th>Chord</th>
<th>Tapestry</th>
<th>Kademlia</th>
<th>Kelips</th>
</tr>
</thead>
<tbody>
<tr>
<td># successors</td>
<td>4 – 32</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Finger base</td>
<td>2 – 128</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Finger stabilization</td>
<td>2 – 32 min</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SuccList stabilization</td>
<td>1 – 32 min</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recursive routing</td>
<td>Yes / No</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Properties of Tapestry (Zhao et. al., UC Berkeley TR 2001):

- Tree-like geometry
- Rtg. table used for both correctness and performance
- Recursive routing
- $\log(n)$ state, $\log(n)$ hops

Parameters Explored:

<table>
<thead>
<tr>
<th></th>
<th>Chord</th>
<th>Tapestry</th>
<th>Kademlia</th>
<th>Kelips</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID Base</td>
<td>2 - 128</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stabilization</td>
<td>2 – 32 min</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Backups per entry</td>
<td>1 – 4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Backups used in lookups</td>
<td>1 – 4</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Overlays

Properties of Kademlia *(Maymounkov & Mazières, IPTPS 2002)*:

- **XOR routing metric**
- Lookups refresh routing state
- Iterative routing
- \( \log(n) \) state, \( \log(n) \) hops

Parameters Explored:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>( k ) (bucket size)</td>
<td>8 – 32</td>
</tr>
<tr>
<td>( \alpha ) (parallel lookups)</td>
<td>1 – 5</td>
</tr>
<tr>
<td>Stabilization timer</td>
<td>2 – 32 min</td>
</tr>
<tr>
<td>Refresh rate</td>
<td>2 – 32 min</td>
</tr>
</tbody>
</table>
Properties of Kelips (Gupta et. al., IPTPS 2003):

- Nodes hashed into $n^{\frac{1}{2}}$ groups
- Keep contacts in each other group
- Use p2p gossip state maintenance
- $O(n^{\frac{1}{2}})$ state, 2 hops

(Some of the) Parameters Explored:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gossip interval</td>
<td>0.125 – 24 min</td>
</tr>
<tr>
<td>Contacts per group</td>
<td>2 – 8</td>
</tr>
<tr>
<td>New item gossip count</td>
<td>0 - 4</td>
</tr>
<tr>
<td>Routing entry timeout</td>
<td>5 – 40 min</td>
</tr>
</tbody>
</table>
Experimental Methodology

- **p2psim**, a discrete event simulator ([http://pdos.lcs.mit.edu/p2psim](http://pdos.lcs.mit.edu/p2psim))
  - Simulates network delay

- Nodes generate lookups for random keys every 116 seconds
  - As observed by Saroiu et. al. for Kazaa traffic
    
    *An analysis of content delivery systems, OSDI 2002*

- Observed tradeoff between bandwidth and latency
  - Background maintenance traffic
  - Timeouts incurred during lookups
Baseline Results – Chord (Recursive)
Baseline Results - Tapestry

![Graph showing Mean Lookup Latency vs Total Traffic for different Base values](image)

- X: Base 2
- ▲: Base 4
- ⋆: Base 16
- △: Base 32
- □: Base 64
- □: Base 128

Y-axis: Mean Lookup Latency (milliseconds)
X-axis: Total Traffic (bytes per node per second)
Baseline Results - Kademlia

![Graph showing Mean Lookup Latency vs Total Traffic for different parallel lookups and values of k.

- 1 Parallel Lookup
- 2 Parallel Lookups
- 3 Parallel Lookups
- 4 Parallel Lookups
- 5 Parallel Lookups

The graph plots Mean Lookup Latency (milliseconds) on the y-axis against Total Traffic (bytes per node per second) on the x-axis. The data points are labeled with k values for each parallel lookup configuration.]

- k=8
- k=16
- k=32

[449 50 100 150 200 250]
[386 0 1 2 3 4 5]
[334 276 219 162]
[162 278 294 263]
Baseline Results - Kelips

![Graph showing baseline results for different contact group sizes with varying total traffic.](image-url)
Baseline Results - All

![Graph showing Mean Lookup Latency (milliseconds) vs Total Traffic (bytes per node per second) for various P2P lookup protocols: Chord (Iterative), Chord (Recursive), Tapestry, Kademlia, and Kelips. The x-axis represents total traffic in bytes per node per second, ranging from 0 to 30. The y-axis represents mean lookup latency in milliseconds, ranging from 0 to 500. Each protocol is represented by a different color line, with Chord (Iterative) in red, Chord (Recursive) in blue, Tapestry in green, Kademlia in orange, and Kelips in purple. The graph illustrates the performance of these protocols under varying traffic loads.]
Churn Results - All

![Graph showing mean lookup latency vs total traffic for different protocols: Chord (Iterative), Chord (Recursive), Tapestry, and Kelips. The x-axis represents total traffic (bytes per node per second), and the y-axis represents mean lookup latency (milliseconds). The graph illustrates the performance of each protocol under varying traffic loads.](image)
Non-transitive Results - All

- Chord (Iterative)
- Chord (Recursive)
- Tapestry
- Kademlia
- Kelips

<table>
<thead>
<tr>
<th>Total Traffic (bytes per node per second)</th>
<th>Mean Lookup Latency (milliseconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>25</td>
<td>25</td>
</tr>
<tr>
<td>30</td>
<td>30</td>
</tr>
</tbody>
</table>
Non-transitive + Churn Results - All

![Graph showing Mean Lookup Latency (milliseconds) against Total Traffic (bytes per node per second) for different protocols: Chord (Iterative), Chord (Recursive), Kelips, and Tapestry.](image)
Discussion

- Performance of a particular protocol can vary widely
  - Careful tuning of parameters greatly improves performance
- Low rate of churn on PlanetLab has little effect on most protocols
  - Optimal configuration:
    - Large number of neighbors (base)
    - Low maintenance traffic (stabilization)
- Non-transitivity has a greater effect
  - Recursive routing a big win
  - Strictness of Chord hinders its performance
Future Work

• By next Friday
  • Analysis of overlays in the presence of variable-latency links
  • Data for Kademlia in churn scenario

• Future research topics
  • More overlays (Koorde, one-hop, etc.)
  • Effects of link failures
  • Effects of asymmetric links
  • Scaling simulation up to thousands of nodes
  • Adaptive, self-tuning parameters
Summary

• Our goal: Explore the effects of real world conditions and parameter tuning on the performance of structured overlays

• Real world data was collected from the PlanetLab testbed

• Illustrated tradeoffs within and between four overlay protocols

• Non-transitivity has a large effect on performance

• Recommendations for system designers:
  • Choose an appropriate overlay for target environment
  • Carefully tune parameters for that overlay
Why Non-transitivity Breaks Chord