Minimizing Churn in Distributed Systems

P. Brighten Godfrey, Scott Shenker, and Ion Stoica
UC Berkeley
Computer Science Division

–Anjali Sridhar
CS598 – Brighten Godfrey
Outline

- Peer to Peer Networks
- Churn and Overlay Networks
- Motivation to reduce churn in Peer to Peer networks
- Existing Solution
- Proposed Solution
- Results
Peer to Peer Networks

- Peer-to-peer (P2P) computing or networking is a distributed application architecture that partitions tasks or work loads between peers.
- Peers are equally privileged, equipotent participants in the application.
- Eg: Napster, Gnutella
Recap - Churn and Overlay Networks

- Churn rate is the rate of turnover of nodes in a collective over a specific period of time.
- Overlay Network is a network built on top of another network by virtual or logical links. Each of these links may be composed of one or more physical links.
Motivation

- Price of Churn –
  - dropped messages
  - data inconsistency
  - increased user experienced latency
  - decreased service quality

- Selecting a reliable set of nodes to host replicas of data
- Choosing nodes in a tree where failure is most costly
Related Work

- Three types of predictors for predicting nodes which will be online for a greater period of time
- Non-equitable distribution of storage due to bias towards highly available nodes
Node Selection Strategies

- Stability– Predictive – based on individual node characteristics e.g. past uptime
- Stability–Agnostic – ignore any information
- Fixed – never replaces a failed node
- Replacement – always replaces a failed node
- Example :
  1. Predictive Fixed – Planet lab nodes
  2. Predictive Replacement – Dynamically minimize churn based on longest current uptime
Churn Simulations

- **Model** – Nodes may be “up” or “down”. The “up” nodes may be “in use” or “available”. There are $k$ nodes selected from a total of $n$ nodes using the below strategies where $k = \alpha n$

- **Selection Strategies** – Predictive Fixed, Agnostic Fixed, Predictive Replacement, Agnostic Replacement

- **Traces Collected From** – PlanetLab, Web Sites, Microsoft PC’s, Skype superpeers, Gnutella peers
RR and PL

- Random Replacement (RR) is a kind of Agnostic Replacement strategy where you replace a failed node with a uniform-random available node.
- Preference List (PL) is also a kind of Agnostic Replacement strategy where you replace a failed node with the highest ranked node in a preference list.
**Fixed Random** (*Agnostic Fixed Strategy*): Pick $k$ uniform-random nodes

**Fixed Descent** (*Predictive Fixed Strategy*): Discard the 50% of nodes that were up least during the observation period. Pick $k$ random remaining nodes

**Fixed Longest Lived** (*Predictive Fixed Strategy*): Pick the $k$ nodes which had greatest average session time

**Fixed Most Available** (*Predictive Fixed Strategy*): Pick the $k$ nodes that spent the most time up

**Longest Uptime** (*Predictive Replacement Strategy*): Select the node with longest current uptime

**Max Expectation** (*Predictive Replacement Strategy*): Select the node with greatest expected remaining uptime, conditioned on its current uptime

**Optimal** (*Predictive Replacement Strategy*): Select the node with longest time until next failure. This requires future knowledge, but provides a useful comparison. It is the optimal replacement strategy
Passive PL (Agnostic Replacement Strategy): Given a ranking of the nodes, after an in-use node fails, replace it with the most preferable available node.

Active PL (Agnostic Replacement Strategy): Same as Passive PL, except when a node becomes available that’s preferable to one we’re using, switch to it, discarding the least preferable in-use node.

Random Replacement (Agnostic Replacement Strategy): Pick k random initial nodes. After one fails, replace it with a uniform-random available node.
**Why RR is better than PL?**

- In RR the longer that a node has been active the greater probability of it being in the “in use” set already.
- The longer a node has been active the greater the probability of it being available in the future. This is only true in typical failure patterns.
- RR performs best when there is skewed session time of nodes
Applications

- DHT neighbor selection in Chord
- Multicast – overlay multicast tree, RON
A Distributed Hash Table is a decentralized Hash Table which allows you to perform Lookup, Select and Delete on (Key,Value) pairs.

- Load Balancing, Look up and Insert efficiencies and churn are some of the performance bottlenecks to consider.
- Churn causes packet losses when there is latency in discovering that an overlay link has failed.
Chord

Finger Table at N80

<table>
<thead>
<tr>
<th>i</th>
<th>f[i]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>96</td>
</tr>
<tr>
<td>1</td>
<td>96</td>
</tr>
<tr>
<td>2</td>
<td>96</td>
</tr>
<tr>
<td>3</td>
<td>96</td>
</tr>
<tr>
<td>4</td>
<td>96</td>
</tr>
<tr>
<td>5</td>
<td>112</td>
</tr>
<tr>
<td>6</td>
<td>16</td>
</tr>
</tbody>
</table>

Say \( m=7 \)

\[ i \text{th entry at peer with id } n \text{ is first peer with id } \geq n + 2^i (\text{mod } 2^m) \]
Results

Randomized Topology has 29% fewer failed requests when avg n = 850
Multicast

- Multicast Overlay Networks— Messages broadcast from one node to only nodes belonging to a particular group
- The single source multicast tree consists of a root that never fails
- When a node fails, each of its descendants experiences an “interruption”
- The three strategies for selecting the parent among m random suitable nodes – Longest Uptime, Minimum Depth and Minimum Latency
Multicast—Results

(a) Interruption rate in Gnutella trace
(b) Interruption rate in Skype trace
(c) Mean node depth in Skype trace
References

- I. Gupta. CS425–Fall2010.Lecture 10
Questions