A Self-tuning DHT for RELOAD

draft-maenpaa-p2psip-self-tuning-00

Jouni Mäenpää
Gonzalo Camarillo
Jani Hautakorpi

Background (1/2)

- **Peers in a P2PSIP overlay network run a DHT to organize the overlay**
- **When configuring a DHT, some network characteristics need to be known in advance**
  - **Churn rate**
  - **Network size**
- **This information is used to configure static values for**
  - **Routing table size**
  - **Neighborhood set size**
  - **Stabilization rate**

Background (2/2)

- **Problem: what if the characteristics change?**
- **If operating conditions become worse than expected**
  - **Routing tables become inaccurate**
  - **The network may become partitioned**
- **If operating conditions become better than expected**
  - **Bandwidth is wasted**
- **It is not possible to achieve a low failure rate and low communication overhead by using fixed parameters**

Self-tuning DHT (1/2)

- **Takes into account the evolution of network conditions and adapts to them**
- **Implemented as a new topology plugin for RELOAD**
  - **Based on the Chord DHT algorithm**
- **A stabilization routine is used to keep routing information consistent with overlay topology**
  - **Successor stabilization**
  - **Finger stabilization**
  - **Predecessor stabilization**

Self-tuning DHT (2/2)

- **Uses periodic stabilization in contrast to reactive stabilization**
  - **In large-scale (>1000 peers) or high-churn overlays, reactive stabilization can result in congestion collapse [1]**
- **Each peer collects statistical data about the network**
  - **This data is used to dynamically adjust DHT parameters**
- **Benefits**
  - **No need to tune DHT parameters manually**
  - **The system adapts to changing operating conditions**
  - **Low failure rate and low stabilization overhead**

Next steps?

Questions?

Additional Information (1/2)

- **Step 1: Estimate overlay size**
  - **Use the density of peer identifiers in the neighborhood set [2,3]**
- **Step 2: Estimate leave rate**
  - **Peers are assumed to leave the overlay according to a Poisson process [3]**
  - **Use departures in routing table and neighborhood set to produce an estimate of the leave rate [3]**
  - **Each peer maintains a history of the last K departures**

Additional Information (2/2)

- **Step 3: Estimate join rate**
  - **Peers are assumed to join the overlay according to a Poisson process [2]**
  - **Peers exchange information about their uptimes**
  - **Use the age of peers in the routing table and neighborhood set to calculate an estimate of the join rate [2]**
- **Step 4: Calculate the stabilization interval**
  - **A Chord network in a ring-like state remains in a ring-like state as long as peers send $O(\log^2(N))$ messages before N new peers join or N/2 peers fail [4]**

References

- **[1] Rhea, S., Geels, D., Roscoe, T., and J. Kubiatowicz, "Handling churn in a DHT", In Proc. of the USENIX Annual Techincal Conference June 2004.**
- **[2] Ghinita, G. and Y. Teo, "An adaptive stabilization framework for distributed hash tables", 20th International Parallel and Distributed Processing Symposium April 2006.**
- **[3] Mahajan, R., Castro, M., and A. Rowstron, "Controlling the cost of reliability in peer-to-peer overlays", In Proceedings of the 2nd International Workshop on Peer-to-Peer Systems Feb. 2003.**
- **[4] Liben-Nowell, D., Balakrishnan, H., and D. Karger, "Observations on the dynamic evolution of peer-to-peer networks", In Proc. of the First International Workshop on Peer-to-Peer Systems March 2002.**