A scalable content-addressable network (CAN)

SIGCOMM 2001

Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, Scott Shenker

Dept. of Electrical Eng. & Comp. Sci. AT&T Center for Internet Research at ICSI
UC Berkeley Berkeley, CA, USA

What is a “scalable content-addressable network”? 

• Network: distributed system
• Content-addressable:
  – Hash table: maps keys onto values
  – P2P: “R.E.M. - Drive.mp3” \(\rightarrow\) 128.178.15.30
• Scalable: functionality on Internet-like scales

Contents

• Design of CAN:
  – Routing
  – Construction
  – Node departure
• Design improvements
  – Shorter path lengths, increased fault tolerance
  – CAN performance
• Summary

Design of CAN

• What does CAN?
  – Insertion, lookup, and deletion of \((key, value)\) pairs
• d-dimensional Cartesian coordinate space (d-torus)
• Every node owns a distinct zone
• Map key \(K_j\) onto a point \(P_j\) using a uniform hash function
• \((K_j, V_j)\) is stored at the node \(N_i\) that owns the zone with \(P_j\)
Routing in CAN

• How to find a point \( P_1 \) at \((0.7/0.6)\)?
  – Node maintains routing table with neighbors
  – Follow the straight line path through the Cartesian space

Node departure

– Someone has to take over the zone
– Explicit handover of the zone to one of the neighbors
– Merge to “valid” zone if possible
– If not possible: two zones are temporary handled by the smallest neighbor
– Failure: Immediate takeover algorithm
  • Neighbors initialize takeover timers
  • Neighbor with smallest timer takes over zone

Construction of CAN

Design Improvements

• Increasing the dimensions
  – Reduces the routing path lengths
  – Small increase of coordinate routing table
• Multiple independent coordinate spaces (realities)
  – \( r \) realities
  – Each node is assigned to \( r \) different zones, one on every reality
  – \( r \) independent neighbor sets
    • \( \Rightarrow \) higher routing fault tolerance
    • \( \Rightarrow \) route to neighbor closest to the destination
  – \( r \) replicates of every key-value pair
• Overloading zones:
  – More than one peer in the same zone
  – Node knows peers in its own zone and one node of each neighboring zone
  – Split zone if there are a certain number of peers
  – Divide or replicate key-value pairs of the zone
  – \( \Rightarrow \) improved fault tolerance, reduced per-hop latency
Summary

- **Completely distributed:** no centralized control or configuration
- **Scalable:**
  - Path length in hops with $d$ dimensions: $O(d(n^{\frac{1}{d}}))$
  - Per-node state: $O(d)$
- **Fault-tolerant:** nodes can route around failures
- **Global knowledge:**
  - Mapping of keys