Dynamic Network Topologies: Chord [SML+ 03] and Koorde [KK 03]

Greg Plaxton
Internet Algorithms, Fall 2003
Department of Computer Science
University of Texas at Austin

Chord [SML+ 03]

- Arrange all 2^b b-bit IDs on a ring ($b = 128$, say)
- Each node chooses a random ID; collisions unlikely
- Each object stored in the DHT is hashed to a random ID
- Each node x is responsible for objects with IDs in the interval between the predecessor of x and x (excluding the predecessor of x)
- Each node maintains a finger table

The Chord Finger Table

- The ith finger of a node x is the first node succeeding x by at least 2^{i-1} positions on the ring
- The number of distinct fingers is $\Theta(\log n)$ whp
- Maximum node indegree is $\Theta(\log^2 n)$ whp

Lookup

- Number of messages per lookup $\sim \frac{1}{2}\log n$ expected, $O(\log n)$ whp
  - The constant factor can be improved by increasing the number of fingers, e.g., by having a finger for each power of $1+\varepsilon$ offset instead of each power of 2

Load Balance

- Maximum fraction of the namespace "owned" by a single node is $\Theta(\frac{\log n}{n})$ whp
  - By simulating $O(\log n)$ virtual nodes at each physical node, this fraction can be improved to $\Theta(\frac{1}{n})$ whp
  - But this increases the expected degree of each node to $O(\log^2 n)$

Join

- Pick your ID and look it up to find you successor
- Node i updates its fingers periodically by looking up ID $i + 2^j$ modulo 2^d for each j
  - The total cost of these lookups is $O(\log^2 n)$ expected and whp

Leave

- Passive approach
- Some fingers may become invalid
  - This is a temporary problem since fingers are periodically recomputed
  - The lookup protocol still works since fingers are just an optimization, i.e., successor pointers alone suffice to perform lookups (albeit slowly)

Dynamic Behavior of Chord [LBK 02]

- In practice, a large Chord network is rarely in an "ideal" state, since nodes are constantly joining and leaving
- Any peer-to-peer network needs to expend $\Omega(n \log n)$ messages per half-life in order to remain connected
  - A dynamic version of Chord is presented that matches this lower bound to within a polylogarithmic factor
- Understanding the dynamic behavior of peer-to-peer systems is an important area for future research

Fault Tolerance

- Modify Chord so that each node keeps track of $O(\log n)$ successors instead of just one
- Modify the lookup algorithm to use an appropriate successor pointer whenever the desired finger node is down
- Even if each node independently crashes with probability $\frac{1}{2}$, each lookup (of an object at a live node) succeeds within $O(\log n)$ messages whp

Koorde [KK 03]

- A modified version of Chord based on de Bruijn graphs, one type of bounded degree hypercubic topology
- In a d-dimensional de Bruijn graph, there are 2^d nodes, each of which has a unique d-bit ID
  - The node with ID i is connected to nodes $2i$ and $2i+1$ modulo 2^d
  - Can route to any destination in d hops by successively "shifting in" the bits of the destination ID

Koorde Neighbors

- A node with ID i maintains pointers to two other nodes:
  - The successor of i
  - The predecessor of node $2i$ modulo 2^d, where d denotes the number of bits in an ID, e.g., 128
- Koorde emulates the de Bruijn lookup path by visiting the predecessor of each de Bruijn ID on that path
  - Sometimes it is necessary to follow additional successor pointers in order to maintain this invariant
  - Still, the total number of messages per lookup is $O(\log n)$ whp

Non-Constant Degree Koorde

- The d-dimensional de Bruijn can be generalized to base k, in which case node i is connected to nodes $k \cdot i + j$ modulo k^d, $0 \leq j < k$
- The diameter is reduced to $\Theta(\log_k n)$
- Koorde node i maintains pointers to k consecutive nodes beginning at the predecessor of $k \cdot i$ modulo k^d
  - Each de Bruijn routing step can be emulated with an expected constant number of messages, so routing uses $O(\log_k n)$ expected hops
  - For $k = \Theta(\log n)$, we get $\Theta(\log n)$ degree and $\Theta(\frac{\log n}{\log \log n})$ diameter

Fault Tolerance

- Koorde node i maintains pointers to:
  - A block of $\Theta(\log n)$ successors as in Chord
  - A block of nodes consisting of $\Theta(\log n)$ nodes before, and $\Theta(\log n)$ nodes after, position $i \cdot k$ modulo 2^d
- Even if each node independently crashes with probability $\frac{1}{2}$, each lookup (of an object at a live node) succeeds within expected $O(\frac{\log n}{\log \log n})$ messages