Peer-to-Peer and Overlay Networks
Structured networks: CAN

Ricardo Lopes Pereira
ricardo.pereira@inesc-id.pt

IST

29-10-2012

1Images from A Scalable Content Addressable Network by Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, Scott Shenker
1 Overview

2 Maintenance
   - Joining the network
   - Node departure

3 Latency optimization
   - Goals
   - Reducing the number of hops
   - Reducing the hop latency

4 Performance optimizations
Characteristics

- Implements DHT
- Operations: insert, lookup, remove
- Identifier space is a multidimensional torus. Multidimensional space where the extremes of each dimension are connected
- Each node is responsible for a part of the identifier space
- Completely distributed
- Scalable
- Fault tolerant. There are many possible paths
Identifier space

- d-dimensional Cartesian space
- Identifiers split into several parts, corresponding to the coordinates of each dimension
- Connected extremes: d-Torus
- Each node responsible for a subset of the space
- The entire space is covered by the nodes
- Hash function uniformly distributes objects in the space
Search

- Each node maintains information about the nodes responsible for the zones neighbouring its.
- Nodes forward message to the neighbour closest to the destination.

1's coordinate neighbor set = \{2,3,4,5\}
7's coordinate neighbor set = \{\}
For a d-dimensional space, divided into n equal zones:

- Nodes maintain information on 2d neighbours
- State does not vary with the number of nodes
- Average path length is: \((d/4)(n^{1/d})\)
- For a fixed \(d\), the path length varies with the number of nodes: \(O(n^{1/d})\)
- If \(d = (\log_2 n)/2\), the number of neighbours is \(O(\log n)\) and the path length is \(O(\log n)\)
In order to access the network, node needs to discover nodes which already belong to the network. CAN provides mechanisms to discover nodes which are part of the network:

- A DNS name will resolve to one or mode bootstrap hosts
- These hosts maintain a list of nodes which belong to the CAN network
Joining the network

1's coordinate neighbor set = \{2,3,4,5\}
7's coordinate neighbor set = \{\}

1's coordinate neighbor set = \{2,3,4,7\}
7's coordinate neighbor set = \{1,2,4,5\}
Joining the network

- Node randomly picks a point in the identifier space
- Sends JOIN message to that point (to the node responsible for that zone)
- Node which receives the message divides its zone into two
- Division is performed in the same order on dimension, so that zones may be regrouped latter (ex: xyxy ou xyzxyz)
- `<key, value>` stored in the half handed to the new node will have to be transferred to the new node
- New node builds its neighbour table (using part of the neighbours of the other node and the other node)
- Old node removes some of its neighbours, replacing them for the new node
- Both of them communicate the changes to their neighbours (process performed regularly)
Orderly departure

- Node which is about to leave warns its neighbours
- If possible, the zone is regrouped with one of its neighbours
- Otherwise, the neighbour with the smallest zone will assume temporary responsibility for both zones
- Node which is about to leave transfers data to the new node responsible for that zone
Node or network failure

- Each node periodically sends its list of neighbours and zone dimensions to its neighbours.
- The prolonged absence of a message indicates a node failure.
- When a node detects a failure, it will start a timer proportional to the size of its zone.
- When the timer expires, it will send a TAKEOVER message to the neighbours of the failed node.
- Neighbours which receive the message cancel their timer. If its zone is smaller than that of the sender, it will reply with a TAKEOVER message.
- The node with the smallest zone will take over the zone of the failed node.
- Data kept by the failed node is lost.
- Nodes which post data must periodically republish it.
Concurrent failures

- It’s possible that a node should fail simultaneously with a significant part of its neighbours.
- In this situation, in order to avoid inconsistencies, before performing the recovery process, neighbours start a search procedure (expanding ring) in order to determine the new neighbours.
- Should a node lose all its neighbours, while recovery proceeds, it may send messages by performing flooding to find another node closer to the destination.
Responsibility for multiple zones

- Departures may cause some nodes to become responsible for more than one zone.
- Periodically, an algorithm is run, trying to assign the responsibility of one of the zones to another node:
  - Node determines dimension of the last division
  - If that neighbour has the same volume, they’re combined
  - Otherwise it forwards the request to the neighbour, with tries to do the same, until a node is released and takes responsibility for the zone.
Motivation

- Each overlay hop may correspond to several IP hops, causing a significant delay for message delivery.
- Total average delay is the product of the number of hops by the average hop delay.
Overview
Maintenance
Latency optimization
Performance optimizations

Goals
Reducing the number of hops
Reducing the hop latency

Increasing the number of dimensions

- Increasing the number of dimensions reduces the average number of hops
- It results in a slight increase of the state information at each node
- Each node has more neighbours, gaining resistance against neighbour failures
Increasing the number of realities

- Each reality is a CAN network. We may create several, having each node become responsible for a different zone in each reality.
- Multiplies state and maintenance.
- Data is stored in every reality. Provides fault tolerance.
- Node performs a search using the reality in which it is closer to the key, thus reducing the path length.
Dimensions vs realities

- For the same state requirements, increasing the dimensions provides the shortest paths.
- Increasing realities provides greater resistance to failures and provides data redundancy.
Forwarding RTT

- We may minimize the RTT of each hop, eventually at the cost of using a few more hops.
- Each node measures the RTT to each of its neighbours.
- Forward queries to the node which presents the best progress to RTT ratio.
Multiple nodes per zone

- Use more than one node per zone (maximum of 3 or 4)
- Node state will also contain its peers (provided name!)
- Zone is only divided when it reaches the maximum number of nodes. If that happens, peers are divided by the 2 zones
- Periodically, each nodes requests the list of peers of a neighbouring zone. It measures the RTT to each one and selects the fastest one to be its neighbour
- The content of the zone may be divided or replicated among the peers of that zone
  - Replication increases availability but also the amount of data stored at each node (increases the zone volume)
  - Replication requires mechanisms to ensure consistency
- Advantages: shorter paths (fewer zones), reduce hop latency (multiple choices), greater resistance to failures (several nodes per zone)
Use of multiple hash functions

- Apply several hash functions, storing the data at each of the locations
- Queries may be sent in parallel
- Queries may be sent to the closest key
- Greater availability as there are several copies
- Disadvantages: increases the amount of data at each node, increases the amount of messages when parallel queries are used
Use of landmarks for building topology

- Before joining the network, node measures RTT to a set of $m$ fixed nodes.
- It orders the $m$ landmarks by increasing RTT ($m!$ possibilities).
- Each dimension is successively divided into $m-d$ slices (xyzxyz).
- Node chooses random value within the assigned area.
- Advantages: simulation (using fixed RTTs) reveals that latency is significantly reduced.
- Disadvantages: node distribution is no longer uniform.
Uniform partitioning

- Instead of dividing a random zone, node determines which node has the largest zone and divides that one
- Zone areas become more homogeneous
- Advantages: balances load (stored data)
Replication

- Nodes with lots of queries may replicate content at their neighbours
- Popular content is “spread”
- Nodes with replicas may either respond or forward the query, as not to limit the cache to the periphery
- Replicas expire
Caching

- Each node may cache the replies it received
- When forwarding a query, it will respond with the value on cache
- Cache expires