CS 427 MP3

Karan Shah (khshah3)
Martin Wozniewicz (wozniew1)

We used the Hashing approach for splitting up the ring of machines. The reason for this was because it requires less reshuffling when a machine joins or leaves. Instead of shuffling all of the data in order to equally space out the members when the member list changes, all we need to worry about is the machine in question's successor or predecessor (depending on whether it is joining or leaving).

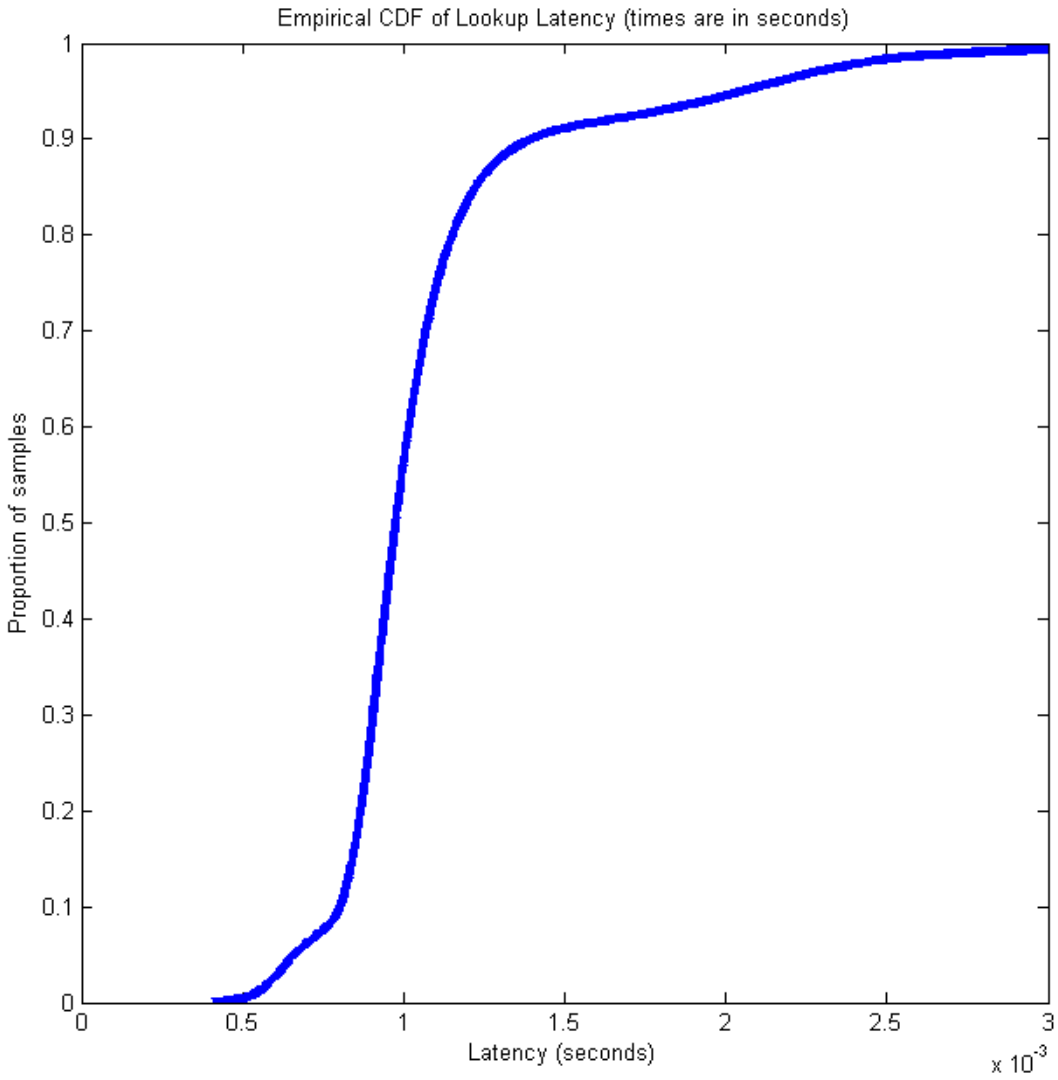
When a machine A joins, it receives data sequentially from its successor B, starting at B's lowest key. So, in effect, A is "sliding" up until it reaches its position determined by the hash value of its identifier.

Similarly, when A leaves, it sends its key/value pairs from highest-key to lowest, "sliding" back down until all of its data belongs to its successor (then the machine can disconnect).

To handle stale gossip data, if a machine is in the process of sending data and gets a request for something it has already sent to another, it will simply respond with the new owner's information. For instance, if the ring is A -> B -> C -> A, and B is leaving, it is sending its keys to A. If C requests a key that B no longer has, B sends A's address to C, and C updates B's location in its own representation of the ring ("sliding" it down to the position of that key, since it must be below it).

The entire time, all machines are randomly gossiping their rings to one another in order to keep the system in sync and hopefully avoid these conflicts in the first place.

Below is the CDF of latencies when running on 4 machines. Mean is 0.0013 seconds.


Empirical CDF of Lookup Latency (times are in seconds)
Proportion of samples
Latency (seconds)
x 10^-3
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
0.5
1
1.5
2
2.5
3