Riak

Background

Riak is another type of distributed database system that hopes to optimize on great write and read throughput. It has many similarities to Cassandra (such as the replication schema, and the partitioning schema), which I have covered previously, so it seems most reasonable to discuss the main differences between it and Cassandra.

Data Modeling

Unlike Cassandra which is a wide column store (and has primary and clustering keys for secondary indexes), Riak is just a basic key-value store where the value can be literally anything (allows for more complex things to be stored like images or a deeply nested json string).

Generally speaking, to do queries on data for something other than a key requires adding extra pieces of metadata to each key-value pair for a secondary index or integrating with the Riak search index (we will discuss search indexes in the future).

Conflict Resolution

Both Riak and Cassandra use a leaderless replication system, which inevitably will lead to conflicting concurrent writes.

Recall: Cassandra uses last write wins, which loses data and is based on unreliable timestamps.

Instead, Riak keeps dotted version vectors associated with all objects in a database allowing it to detect concurrent writes and store them all as siblings. The next application server that reads this field is supposed to take all of the siblings into consideration and merge them in a reasonable way for the next write.

CRDTs

If you don’t want to potentially take upon the burden of having to merge database siblings, Riak has good support for CRDTs for structures like:

- Counters
- Sets
- Maps

Conclusion

Riak is yet another high throughput database which achieves its goals through partitioning, multi-master replication, LSM trees, and read repair + anti-entropy.

Unlike Cassandra, it is a key value store, which means that generally you should only use it if single key accesses are important, as opposed to range queries over a single partition (secondary indexes in Riak are not ideal).

Compared to Cassandra, the main benefits of using Riak are greater flexibility in what can be stored as values, as well as no data loss on conflicting writes, either via the use of application code for merging siblings or CRDTs.