Memcached and Redis

Background

When people speak of using Redis or Memcache, it is generally for highly performant data accesses such as needed in caching. This is because they both rely on using random access memory (RAM) as their main data stores, which is capable of reading and writing at orders of magnitude faster than a traditional hard disk.

However, as we will see, there are significant differences between the services.

Memcached

An extremely simple service to run a bare minimum distributed string to string hashmap across many servers. Best for single key accesses and sets.

Features:

- Partitioning via a consistent hashing ring
- Least recently used cache (entries can also be invalidated with an expiration)
- Compare and set functionality
- No failure handling or replication measures built in, need to manually make sure using some other service that all clients agree on the same list of memcached servers

(Single Node) Redis

Redis is like a modification of Memcached (is also an LRU distributed hashmap with expirations), that allows it to be much more multifunctional.

Features:

- Other data types such as strings, sets, maps, lists (with built in atomic operations)
- Transactions on a single partition
- Range queries on a single partition
- Disk persistence via checkpointing or a write ahead log

Redis Cluster

Distributed version of Redis meant to provide high availability and consistency.

Features:

- Single leader replication with automatic failover
  - As a result, some writes can be lost if leader fails before replicating
  - Gossip protocol with heartbeats (which also convey which nodes hold which partitions) used to determine when leader is no longer up
  - Other master replicas vote on new master, election occurs when a quorum is received
- 16,384 hash ranges, which can be moved between nodes
  - This is like the pre-generated partition ranges we discussed back in the partitioning video!

Memcached Redis Comparison

Ultimately, when designing systems these are both very viable solutions, but it looks like Redis is effectively just Memcache with a bunch of features. Why would you ever not want to use Redis Cluster?

- Need strongly consistent data (no lost writes on failover)
- Want alternate replication patterns such as master-master (leaderless)
- Prefer to use a coordination service for configuration/partition management as opposed to gossip

Conclusion

In real large scale systems, caching is used whenever possible to reduce database load and latency of the application. While caching is hugely beneficial, it certainly increases system complexity, especially when ensuring consistent data on writes. Nonetheless, it is generally a must-have and Redis and Memcached are two valid ways of implementing it.