SUMMARY

DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., and Vogels, W. 2007. Dynamo: amazon's highly available key-value store. *SIGOPS Oper. Syst. Rev.* 41, 6 (Oct. 2007), 205-220.
Date: 1 February 2010

The paper describes Dynamo; Amazon's key-value store, that achieves high availability and scalability even in case of data center failures, through data partitioning and replication. Dynamo guarantees to provide its clients a certain level of availability[1]; measured by the upper bound on the latency under a particular workload. Obviously, the trade-off is in its weakened support for consistency, which is *eventually* resolved at the application level. Yet, Dynamo's object versioning facilities aim to reduce the burden on the applications to some extent. Finally, in Dynamo, queries are assumed to be reads and writes on single items uniquely identified by some primary key.

Dynamo uses a variant of *consistent hashing* to distribute keys among consecutive peer nodes in a ring overlay. When a node is introduced or removed, key re-allocation takes place with the neighboring nodes. Eventually, all of the nodes in the ring will be notified of the new arrangement through a *gossip-based* propagation protocol. Each node in the ring is responsible for replicating its data on $n-1$ successor healthy nodes. For a given key, every node can determine the list of nodes responsible for storing the key-value pairs; hence the system is essentially a *zero-hop distributed hash table*. In Dynamo, a write always proceeds as long as it is propagated to at least W out of the N peers responsible for eventually storing the data. Consequently, there may be multiple versions of the same object on different peers. In this respect, the states (or versions) of the key-value pairs are maintained by *vector clocks*. If version branching occurs, all of the conflicting versions are returned to and reconciled by the client. Finally, for durability; when a node is down and cannot participate in a write, the replica that would have normally lived on it, together with the write, are propagated to a temporary node. When the original node awakens, the content will be transferred back.

In conclusion, it was observed that Dynamo could survive the peak season workloads with $300ms$ latency[2]. However, by compromising durability and using local caching, this number was reduced. Different strategies were applied to ensure uniform load distribution among the peers. Finally, on the average, the percentage of requests that needed reconciliation was observed to be around 0.06%.

SUMMARIZED BY: Gunes Aluc

[1] expressed in Service Level Agreements (SLAs)
[2] calculated over 99.9% of the operations