COPS: Critique

Abhishek Srivastava
Student Id: 861307778

CS202 Advanced Operating Systems
March 3, 2017

Summary

The paper presents issue with existing distributed data stores who sacrifice strong consistency to provide low latency operations by using "eventual" consistency in place of "causal" consistency model.

Main Points

Authors proposes scalable *causal consistency model with convergent conflict handling* which they call *causal+ consistency* for wide area storage with COPS & COPS-GT. Causal Component ensures consistency of data stored between different operations. Convergent conflict handling takes care of replicas so that they do not diverge from original and conflict while updating data with same key are dealt at all places identically.

For current distributed data storages Authors also proposes few properties (**ALPS**): **Availability**: All operations must be completed with no blocking or error returned, **Low Latency**: Operations issued by client must be performed "quickly", **Partition tolerance**: Data store continues to operate even with different network partitions and **High Scalability**: Adding resources increase performance and storage capacity, along with **Strong Consistency**: Distributed systems should provide a single consistent data.

COPS & COPS-GT system provide ALPS properties with causal+ consistency. They has two main components: *Key-Value store* and *Client library*.

- Key-Value store is similar to traditional tuple store but differ by storing versions of written values and their dependencies and provides linearizable operations on keys. They provide interfaces: *get_by_version*, *put_after* and *dep_check*. COPS clusters maintain their local key-store copy but scale it using consistent hashing and perform chain replication for fault tolerance in asynchronous manner to other nodes.

- Client Library consist on four API's: *createContext*, *deleteContext*, *put*, *get* or *get_trans*. They also differ with traditional interface by using context id as argument for each get and put client operations to track their causal dependencies. get_trans operation provides consistent view of multiple specified key-value pairs. Garbage collection is used to reduce the state requirements to store all the dependencies by removing those dependencies which are committed to all the replicas. To reduce the number of dependency checks while replications client library maintains list of nearest dependencies and use it for its optimization.

The evaluation done by the Authors shows that COPS and COPS-GT provide a low-latency, high throughput and scalable distributed data storage system.

Limitations/Critique

The Drawbacks of COPS and COPS-GT can be that they are less efficient for the write-heavy workload environments. They are also not robust enough for datacenter failures handling and long network partitions.

Proposed Extension

COPS and COPS-GT can be extended to large scale data which are very dependent and have large dependencies. For those systems replication will take much more time. Also better methods for dependency checking for a key-value can be designed which can increase the efficiency especially for COPS-GT get_trans method.