COMPSCI 590S Systems for Data Sciencea **Fall 2016**

Lecture 24: DynamoDB

Lecturer: Emery Berger *Scribe(s): Arjun Sreedharan,Janani Krishna*

24.1 DynamoDB

DynamoDB is a fully managed proprietary NoSQL database service that is offered by Amazon as part of the standard Amazon Web Services. It uses synchronous replication across multiple data centers for high durability and availability. It can be described as an *always on* key-value storage. It's targeted mainly at applications that need an *always writeable* data store.

Amazon does not give away free GPU runtime because it fears people will use them to mine bitcoins. Bitcoin is a cryptocurrency that utilizes public-key cryptography. As task of mining bitcoins have become increasingly difficult, miners use custom chips called ASICs (Application Specific Integrated Circuit) for mining.

24.1.1 The Model

In Dynamo, availability is achieved by sacrificing consistency under certain failure scenarios. It's safe to assume that there is always a small number of servers and network components that are failing at any given time. If you recall CAP theorem (Consistency, Availability, Partition tolerance) - all three cannot be achieved together. Therefore, Amazon stresses heavily on availability and adopts the notion of *Eventual Consistency*. Eventual Consistency here is driven by the client and this is based on the mechanism of versioning aided by vector clocks.

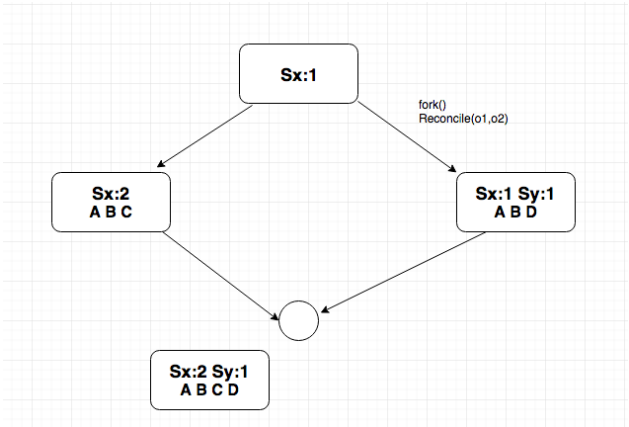

Figure 24.1: Reconciliation Mechanism

The two main operations offered by Dynamo are: *get(key)* and *put(key, context, value)*. When a client wants to update the value associated with a key, it must specify which version it is updating. This is done by passing the context it obtained from an earlier read operation. On a *get()*, if Dynamo has multiple versions

of values for the key and if it cannot automatically resolve them, then it will return all values with its corresponding contexts (versions) to the client. It's up to the client to reconcile the different versions.(fig 1)

For fault tolerance, it follows the concept of Consistent hashing.

24.1.2 Anti-entropy

Anti-entropy refers to the process of coping with decays in a distributed setting. It involves comparing all replicas of each block of data that exist (or is expected to exist) and refreshing each replica if required. The naive way to compare and refresh each replica would involve large data transfer between the replicas which is undesirable. Therefore, we use a data structure named Merkle tree to limit the amount of data transferred when synchronizing. A Merkle tree is a hash tree where leaves are hashes of the values of individual keys. Parent nodes higher in the tree are hashes of their respective children. To compare data in replicas, Dynamo compares their Merkle trees. The comparison starts at the root. If the hashes are the same, the data is already synchronized. Otherwise, the subtrees that is different is compared. This goes on until the leaf node that's not synchronized.(fig 2)

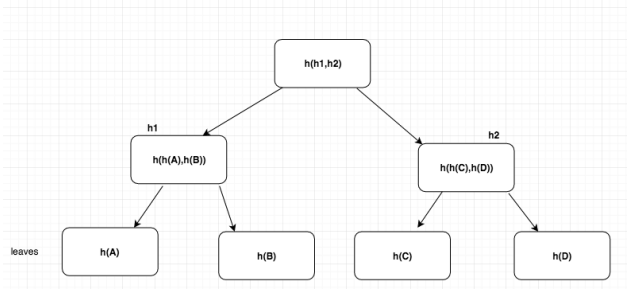

Figure 24.2: Merkle Tree