Nosql Veritabanları

16 Ekim 2010
Özgür Web Günleri
Yeditepe Üniversitesi

CAP Theorem

- Aynı anda aşağıdaki üçü bir arada olamaz!
  - Consistency (Aynı anda tüm birimlerde aynı veri)
  - Availability (Bazı birimlerde sorun olsa bile çalışmaya devam ederler)
  - Partition Tolerance (Önemli miktarda mesaj kaybı olsa bile sistem çalışmaya devam eder)

  Bunlardan sadece ikisi aynı anda elde edilebilir.

Dynamo Paper

Table 1: Summary of techniques used in *Dynamo* and their advantages.

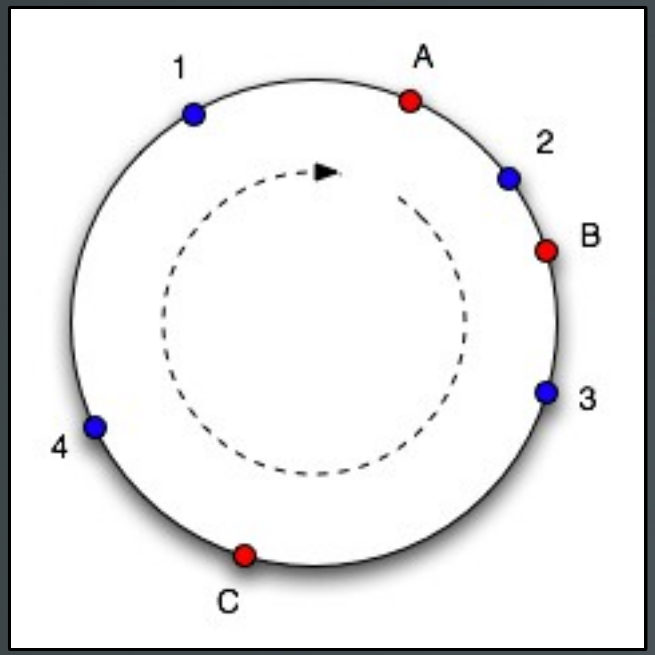
Problem	Technique	Advantage
Partitioning	Consistent Hashing	Incremental Scalability
High Availability for writes	Vector clocks with reconciliation during reads	Version size is decoupled from update rates.
Handling temporary failures	Sloppy Quorum and hinted handoff	Provides high availability and durability guarantee when some of the replicas are not available.
Recovering from permanent failures	Anti-entropy using Merkle trees	Synchronizes divergent replicas in the background.
Membership and failure detection	Gossip-based membership protocol and failure detection.	Preserves symmetry and avoids having a centralized registry for storing membership and node liveness information.

Basit Terimler

- N: Tüm sistem içinde sağlıklı/tercih edilmiş birimlerin sayısı
- R: Okuma anında gereken minimum birim sayısı
- W: Yazma anında gereken minimum birim sayısı
- Coordinator: Okuma veya yazma işlemlerini yapmak üzere belirlenmiş olan birim

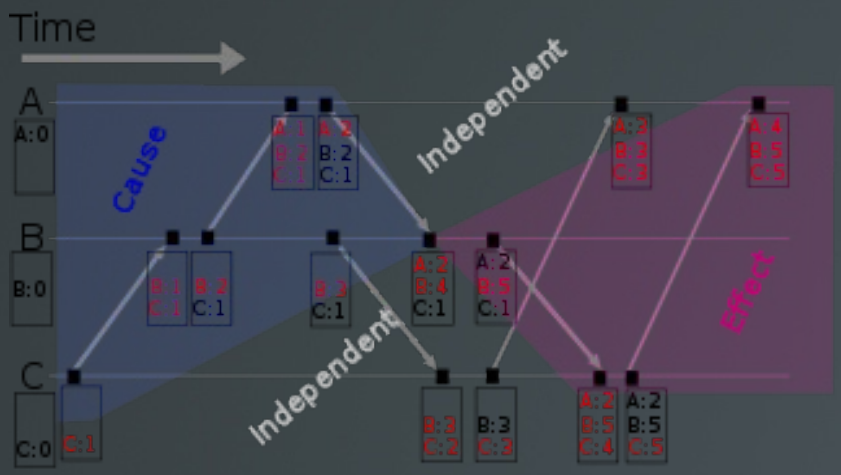
Consistent Hashing

- Hash Table içinde sadece değişiklik kadar K/n key remap edilir.
- K=key sayısı, n=birim (node sayısı)



Vector Clocks

- Yazılan verinin versiyonlarını tutmak için kullanılır.
- Yazılan veriye timestamp eklenir.
- En son veri dikkate alınır.



Vector Clock Erlang uyarlaması

http://bitbucket.org/justin/riak/src/tip/apps/riak/src/vclock.erl

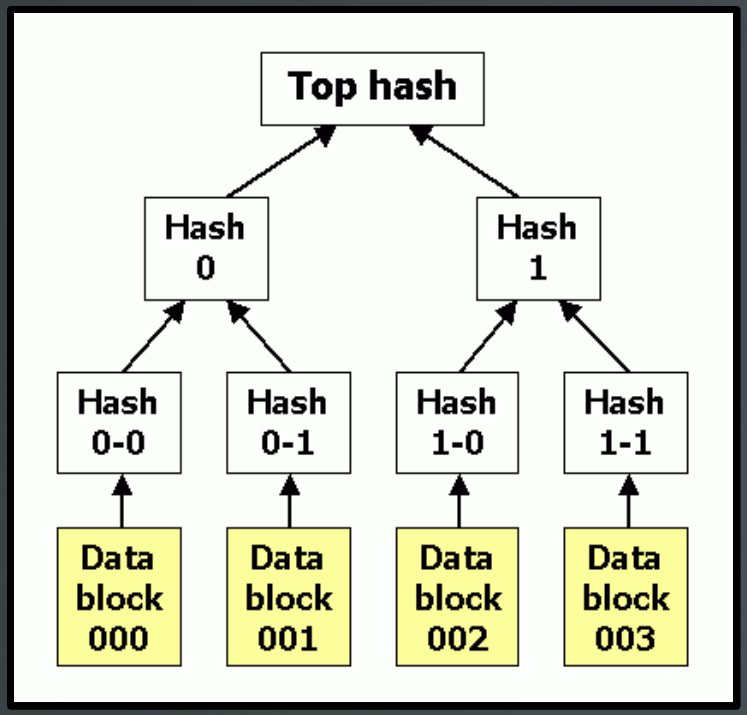
Sloppy Quorum & Hinted Handoff

- Tüm okuma ve yazma işlemleri ilk N adet sağlıklı birimde (node) yapılır. Consistent hashing ring içindeki bazı birimler atlanır.

- Eğer bir birim geçici olarak down olduysa veri sağlıklı bir birime aktarılır ve veriye bir ipucu “hint” eklenerek orijinal alıcısına gönderilmesi gerektiği belirtilir.

Anti-Entropy & Merkle Trees

- Replikaların senkronize edilmesi için kullanılır.
- Replikalar arasındaki farkı en hızlı şekilde tespit etmek ve transfer edilecek veriyi minimize etmek?
- Tüm tree indirilmeden, sadece ilgilenilen kısımla çalışılır.
- Eğer iki birim arasında, köke bağlı iki tree aynıysa bunları senkronize etmek gerekmez.



Gossip Protocol

- Dedikodu mantığıyla çalışır.
- Sanal birimler, aralarında paylaştıkları üyelik bilgilerinin değiştiğini buradan öğrenirler.

Önemli Kavramlar

- Paxos
- Chord
- multi-version concurrency control (MVCC)
- B+ trees
- B trees
- ACID (atomicity, consistency, isolation, durability)

Nosql Türleri

- Kolon Tabanlı (Column Based)
- Belge Depolama (Document Store)
- Key/Value
- Eventually Consistent Key Value Store
- Graph
- Object
- Grid Database
- XML

Hangisi Ne Zaman?

- Örnek data yapısı yarat
- Data türünü tercih et. (hash/column/tuple/JSON)
- Per record data size hesapla,
- Zamansal veri büyümesini hesapla
- Donanım planını kontrol et
- İşletim sistemini kontrol et.

  -> SONRA
- Veri tipi karmaşık değil, K/V, EC K/V
- Veri tipi karmaşık, Document, Column, EC K/V
- Eventual Consistency isteniyor, Column, Document, EC K/V
- Veri lineer olarak ve hızla artıyor, veri tipine göre dağıtıklık özelliği barındıran biri
- Fazla veri (>4TB) analizi ön planda, map/reduce, sistem orkestrasyonu gibi yardımcı unsurlar var mı?
- W or R intensive? Tek tek incele!

Column Families

- Hadoop / HBase=> API: Java / any writer, Protocol: any write call, Query Method: MapReduce Java / any exec, Replication: HDFS Replication, Written in: Java
- Cassandra=> API: many Thrift languages, Query Method: MapReduce, Written in: Java, Concurrency: eventually consistent
- Hypertable=> API: Thrift (Java, PHP, Perl, Python, Ruby, etc.), Protocol: Thrift, Query Method: HQL, native Thrift API, Replication: HDFS Replication, Concurrency: MVCC, Consistency Model: Fully consistent
- Amazon SimpleDB

Document

- CouchDB=> API: JSON, Protocol: REST, Query Method: MapReduceR of JavaScript Funcs, Replication: Master Master, Written in: Erlang, Concurrency: MVCC
- MongoDB=> API: BSON, Protocol: lots of langs, Query Method: dynamic object-based language, Replication: Master Slave, Written in: C++,Concurrency: Update in Place.
- Terrastore=> API: Java & http, Protocol: http, Language: Java, Querying: Range queries, Predicates, Replication: Partitioned with consistent hashing, Consistency: Per-record strict consistency
- RavenDB=> .Net solution. Provides HTTP/JSON access

Key Value

- Riak=> API: JSON, Protocol: REST, Query Method: MapReduce term matching , Scaling: Multiple Masters; Written in: Erlang, Concurrency: eventually consistent (stronger then MVCC via Vector Clocks)
- Chordless=> API: Java & simple RPC to vals, Protocol: internal, Query Method: M/R inside value objects, Scaling: every node is master for its slice of namespace, Written in: Java, Concurrency: serializable transaction isolation
- Redis=> API: Tons of languages, Written in: C, Concurrency: in memory and saves asynchronous disk after a defined time. Append only mode available. Different kinds of fsync policies. Replication: Master / Slave

Key Value - devamı

- Scalaris=> Written in: Erlang, Replication: Strong consistency over replicas, Concurrency: non blocking Paxos, in memory.
- Tokyo Cabinet / Tyrant
- Berkeley DB=> API: Many languages, Written in: C, Replication: Master / Slave, Concurrency: MVCC, License: Sleepycat, Berkeley DB Java Edition: API: Java, Written in: Java, Replication: Master / Slave, Concurrency: serializable transaction isolation
- Mnesia=> ErlangDB

Eventually Consistent K/V

- Amazon Dynamo
- Voldemort=> Amazons Dynamo türevi open source.
- Dynomite=> Written in Erlang. With "data partitioning, versioning, and read repair, and user-provided storage engines provide persistence and query processing".
- KAI=> Amazons Dynamo türevi open source.

Grid Database

- GigaSpaces=> Popular SpaceBased Grid Solution
- Hazelcast=> P2P Data Grid Solution on java.util.*, On a 100 Noce EC2 Cluster (Yaratıcısı Talip Öztürk)

Eventually Consistent K/V

- Amazon Dynamo
- Voldemort=> Amazons Dynamo türevi open source.
- Dynomite=> Written in Erlang. With "data partitioning, versioning, and read repair, and user-provided storage engines provide persistence and query processing".
- KAI=> Amazons Dynamo türevi open source.

Sorular

Kaynakça

- http://www.allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf
- http://www.spiteful.com/2008/03/17/programmers-toolbox-part-3-consistent-hashing/
- https://nosqleast.com/2009/#speaker/sheehy
- http://en.wikipedia.org/wiki/Chord_%28DHT%29
- http://en.wikipedia.org/wiki/Paxos_algorithm
- http://nosql-database.org/
- http://news.ycombinator.com/item?id=1470521
- http://www.julianbrowne.com/article/viewer/brewers-cap-theorem
- http://dbmsmusings.blogspot.com/2010/04/problems-with-cap-and-yahoos-little.html
-

kunthar@gmail.com