Cloud data store services and NoSQL databases

Ricardo Vilaça
Universidade do Minho
Portugal
Context

- Traditional RDBMS were not designed for massive scale.
- Storage of digital data has reached unprecedented levels.
- Centralized storage and processing making extensive and flexible data partitioning unavoidable.
- Emergence of Cloud Computing paradigm/business model.
Cloud data store services

- Elasticity. Scale at any moment.
- Reduce the need to have IT specialized staff to implement reliable and scalable data management solutions.
- Reduce the power demands and consumption on data centers.
Introduction

Cloud data store services

• Dependency on Internet connectivity.

• Dependency on cloud provider.

• Data security (privacy).

• Drawbacks and benefits have to be carefully balanced.
Amazon SimpleDB

• Amazon SimpleDB is one of the Amazon Cloud services, Amazon Web Services.
• Web service that allows to store and run simple queries on structured data.
• Requires no schema, automatically indexes data, and provides real time lookup.
• Highly scalable system optimized for real time and read only applications.
Cloud data store services and NoSQL databases

Amazon SimpleDB

• Data is organized into domains.

• Domains are collections of items that are described by attribute-value pairs.

• A domain is like a spreadsheet.

• However multiple values can be associated with each attribute.

• Each item can have its own unique set of associated attributes.
Cloud data store services and NoSQL databases

Amazon SimpleDB

- CreateDomain
- DeleteDomain
- ListDomains
- PutAttributes
- GetAttributes
- DeleteAttributes
- Select: =, !=, <, >, <=, >=, starts-with, and, or, not, intersection and union.
• Google's App Engine is a toolkit that allows developers to build scalable apps. Python or Java.

• App Engine Datastore API provides a query engine and transactional storage.

• Includes a data modeling API and a SQL-like query language.
Google’s App Engine Datastore API

- An entity has one or more properties and a unique key.
- The entity properties can be a reference to another entity.
- Every entity is of a particular kind, a group of entities that can be returned by a query.
- Datastore API allows to define data models, and create instances of those models.
- Entities are organized into entity groups, sets of one or more entities that can be manipulated in a single transaction.
Cloud data store services and NoSQL databases

Google’s App Engine Datastore API

- A query object interface, and a query language called GQL (Python).
- Query language to filter, order, etc. entities in a procedural style: property conditions, `filter()`; ancestor conditions, `ancestor()`; and ordering, `order()`.

```
SELECT * FROM <kind>

[WHERE <condition> [AND <condition> ...]]

[ORDER BY <property> [ASC | DESC] [, <property> [ASC | DESC] ...]]

[LIMIT [<offset>,] <count>]

[OFFSET <offset>]

<condition> := <property> {< | <= | > | >= | = | != } <value>

<condition> := <property> IN <list>

<condition> := ANCESTOR IS <entity or key>
```
• SQL Azure database

• Windows Azure storage
  • Blob
  • Queue storage
  • Table storage
Large Scale Tuple Stores

- Scalable, elastic and dependable systems.
- Amazon’s Dynamo, Yahoo’s PNUTS, Google’s Bigtable, Facebook’s Cassandra
- Solve internal data management problems and support their current or future Cloud services.
- A simple tuple store interface, that allows applications to insert, query, and remove individual elements.
Requirements

• Scale both in the sheer volume of data that can be held but also in how required resources can be provisioned dynamically and incrementally.

• Rely on distributed systems designed from the beginning to be elastic and highly available.

• The CAP theorem states that under network partitions it is impossible to achieve both strong consistency and availability.
Dynamo and PNUTS

• Amazon’s Dynamo
  • Properties of both databases and distributed hash tables (DHTs)
  • Building block of some of the Amazon Web Services, such as S3.
  • Several distributed systems concepts in a production system.

• Yahoo’s PNUTS
  • Hosted data management service that allows multiple applications to concurrently store and query data.
  • Is a component of the Yahoo!’s Sherpa, an integrated suite of data services.
Bigtable and Cassandra

- Google’s Bigtable
  - Being used internally at Google for web indexing, Google Earth and Google Finance.
  - Used to store Google’s App Engine Datastore entities.
- Facebook’s Cassandra
  - Initially developed by Facebook and now an Apache open source project.
  - Uses most of the ideas of the Dynamo architecture to offer a data model based on Bigtable.
DataDroplets

• Offers a simple application interface providing the atomic manipulation of tuples and the flexible establishment of arbitrary relations among tuples.

• Multi-tuple operations leverage disclosed data relations to manipulate sets of comparable or arbitrarily related elements.

• Provides additional consistency guarantees.
### Dynamo

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a</td>
</tr>
<tr>
<td>3</td>
<td>b</td>
</tr>
<tr>
<td>4</td>
<td>c</td>
</tr>
</tbody>
</table>

$K \rightarrow (V \times C)$
### Data Model

**PNUTS**

<table>
<thead>
<tr>
<th>Key</th>
<th>Attribute 1</th>
<th>Attribute 2</th>
<th>Attribute 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a</td>
<td></td>
<td>b</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>b</td>
<td>c</td>
</tr>
<tr>
<td>4</td>
<td>a</td>
<td>d</td>
<td></td>
</tr>
</tbody>
</table>

\[ K \rightarrow \mathcal{P}(\text{String } \times V) \]
Cloud data store services and NoSQL databases

BigTable

<table>
<thead>
<tr>
<th>Key</th>
<th>Column1</th>
<th>Column2</th>
<th>Column3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a v1</td>
<td>g v1</td>
<td>e v1</td>
</tr>
<tr>
<td></td>
<td>b v2</td>
<td>h v2</td>
<td>f v2</td>
</tr>
<tr>
<td></td>
<td>c v3</td>
<td></td>
<td>g v3</td>
</tr>
<tr>
<td>3</td>
<td>d v1</td>
<td></td>
<td>u v1</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>e v1</td>
<td>e v1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>f v2</td>
<td>f v2</td>
</tr>
</tbody>
</table>

\[ K \rightarrow (\text{String} \rightarrow (\text{String} \times \text{Long} \rightarrow \text{V} )) \]
### Data Model

#### Cassandra

<table>
<thead>
<tr>
<th>Key</th>
<th>Column1</th>
<th>Column2</th>
<th>Column3</th>
<th>Column4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>g</td>
<td>e</td>
<td>x</td>
<td>v</td>
</tr>
<tr>
<td>3</td>
<td>u</td>
<td>u</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>e</td>
<td>e</td>
<td>v</td>
<td>k</td>
</tr>
</tbody>
</table>

\[ K \rightarrow (\text{String} \rightarrow (\text{String} \rightarrow (\text{String} \rightarrow V ))) \]
### DataDroplets

#### Data Model

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a</td>
<td>Blue</td>
</tr>
<tr>
<td>3</td>
<td>b</td>
<td>Red</td>
</tr>
<tr>
<td>4</td>
<td>c</td>
<td>Oval</td>
</tr>
</tbody>
</table>

- Tags: Blue, Oval, Red, Rectangular

\[
\text{String} \rightarrow (K \rightarrow (V \times P\text{String}))
\]

© 2011 Ricardo Vilaça
• `get(key)`

• `put(key, context, value)`
• **get-any**(tableName,key), **get-critical**(tableName,key,version), **get-latest**(tableName,key)

• **put**(tableName,key,value)

• **delete**(tableName,key)

• **test-and-set-put**(tableName,key,value,version)

• **scan**(tableName,selections,projections)

• **rangeScan**(tableName,rangeSelection,projections)

• **multiget**(keys)
BigTable

- put(key,rowMutation)
- get(key,columns)
- delete(key,columns)
- scan(startKey,stopKey(columns))
Cassandra

- **put**(key,rowMutations)
- **get**(key(columns)
- **range**(startKey,endKey(columns)
- **delete**(key(columns)

© 2011 Ricardo Vilaça

Semana da LEI 2011
DataDroplets

- **put**(collection,key, value, tags)
- **get**(collection,key)
- **delete**(collection,key)
- **multiPut**(mapKeys)
- **multiGet**(keys)
- **getByRange**(min, max)
- **getByTags**(tags)
Cloud data store services and NoSQL databases

Dynamo

Architecture

Clients

DHT

Request Routing

Partition

Replication

Storage
Cloud data store services and NoSQL databases

Architecture

BigTable

Clients

Tablets Servers

Master Server

Lock Servers

GFS

Request Routing

Partition

Replication

Storage
Cassandra

Architecture

Clients

Request Routing
Partition
Replication
Storage

DHT

Partitioning

Storage

© 2011 Ricardo Vilaça
DataDroplets

- **Clients**
- **Soft-state layer**
- **DHT**
- **Persistent-state Layer**

- **Request Routing**
- **Partition**
- **Replication**
- **Storage**
- **Replication**
## Tradeoffs

<table>
<thead>
<tr>
<th></th>
<th>Dynamo</th>
<th>PNUTS</th>
<th>BigTable</th>
<th>Cassandra</th>
<th>DataDroplets</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Partition</strong></td>
<td>random</td>
<td>random and ordered</td>
<td>ordered</td>
<td>random and ordered</td>
<td>random, ordered, and correlation</td>
</tr>
<tr>
<td><strong>Consistency</strong></td>
<td>eventual</td>
<td>atomic or stale reads</td>
<td>atomic</td>
<td>eventual</td>
<td>atomic or stale reads</td>
</tr>
<tr>
<td><strong>Multiple versions</strong></td>
<td>version</td>
<td>version</td>
<td>timestamp</td>
<td>timestamp</td>
<td>none</td>
</tr>
<tr>
<td><strong>Replication</strong></td>
<td>quorum</td>
<td>async message broker</td>
<td>file system</td>
<td>quorum</td>
<td>sync or async</td>
</tr>
</tbody>
</table>
Large scale tuple stores

- Each system offers a different data modeling and a distinct expressiveness on operations.
- Flexible data models with dynamic attributes.
- Replace the traditional transactional serializability by eventual consistency.
- Specific trade-offs regarding consistency, availability and migration cost.
Large scale tuple stores

- Focus on applications that have minor consistency requirements and can favor availability.
- Increase complexity at the application logic.
- In most enterprises, it is hard to add this complex layer to the application.
DataDroplets

• Provides additional consistency guarantees and higher level data processing primitives.

• Smooths the migration path for existing applications.

• Fits the access patterns required by most current applications.
Open Issues

- Automatic elasticity.
- Multi-tenancy.
- More complex operations.
- Storage of massive data in commodity hardware.
Ongoing Cloud Research Projects

• **Stratus**: A Layered Approach to Data Management in the Cloud.

• **CumuloNimbo**: A highly scalable transactional multi-tier platform as a service