NoSQL Data Mangement Systems

A Very Subtle Introduction

Hassan Abedi
School of Computer Engineering - IUST

How to reach me:
Email: `hassan.abedi.t@gmail.com`

What NoSQL is and what it is not

NoSQL really is a generic term about data storage systems used in storing and retriving the type of data which isn't in the classical tabular form, like as it's a Realational Database. Actually some may claim is NoSQL not even about a type of database but more of a something that is not SQL. NoSQL trys to answer to three main needs in short:

- They provide **horizontal scaling**.
- They provide really high levels of **availability**.
- They manage to **store and handle more data types**(like key-values or graph) than RDBMS, so they maybe **more flexible**.

It's not bad to note, for ends mentioned above to manifest most NoSQL stores lack true ACID instead they manage to provide three main features[1]:

- **Basic availability**: Each request is guaranteed a responsesuccessful or failed execution.
- **Soft state**: The state of the system may change over time, at times without any input (for eventual consistency).
- **Eventual consistency**: The database may be momentarily inconsistent but will be consistent eventually.

Maybe the biggest difference they make is about data consistency, where most of NoSQL family loosend usual consistency promises a normal RDBMS holds to some sort of In-Future(Eventual) plans to make data consistent.In the moment they are finding significant and growing industry use in **Big Data and real-time web applications**. Some people may interpret NoSQL as "Not Only SQL" (more serious proponents of them would tell you it stands for "No to SQL"!;0) reminding that they may deployed besides a RDBMS or atleast may have SQL like query languages like their older cousins. Data replication is a classical way to get high availability, it's one of main features of NoSQLs, so always remember when someone says i store my data using a NoSQL database she means there is at-least one replica of her data stored somewhere!.
Most of NoSQLs compromise consistency (in the sense of the CAP theorem) in favour of availability and partition tolerance.

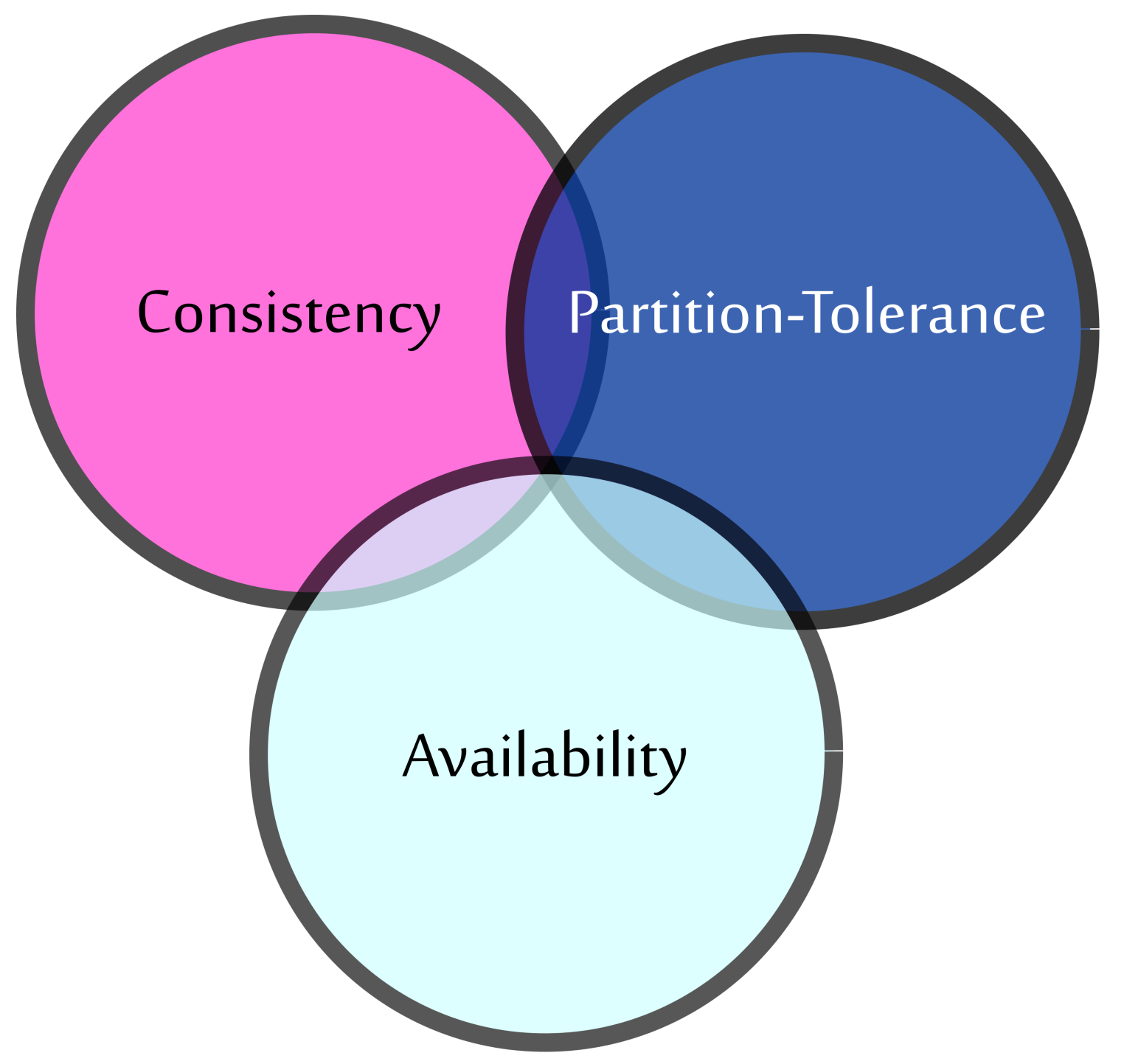

Figure 1: Brewer's CAP theorem states you can only have two out of three of C.A.P. at a moment in a distributed computing system.

Types of NoSQLs

NoSQL storage management systems usually depending the data that they are designed to store are categorized into different types enumerated below:

Column-oriented stores

The column-oriented databases store data as columns as opposed to rows that is prominent in RDBMS, you can add columns and don't worry about default values for cells in added rows. For these databases there are advantages when working with a subset of the available columns. For example, computing maxima, minima, averages and sums, specifically on large datasets, is where these column-oriented data stores outshine in performance. Similarly, when new values are applied for either all rows at once or with same column filters, these databases will allow partial data access without touching unrelated columns and be much faster in execution.
Examples:Apache Cassandra, Hbase, Google BigTable

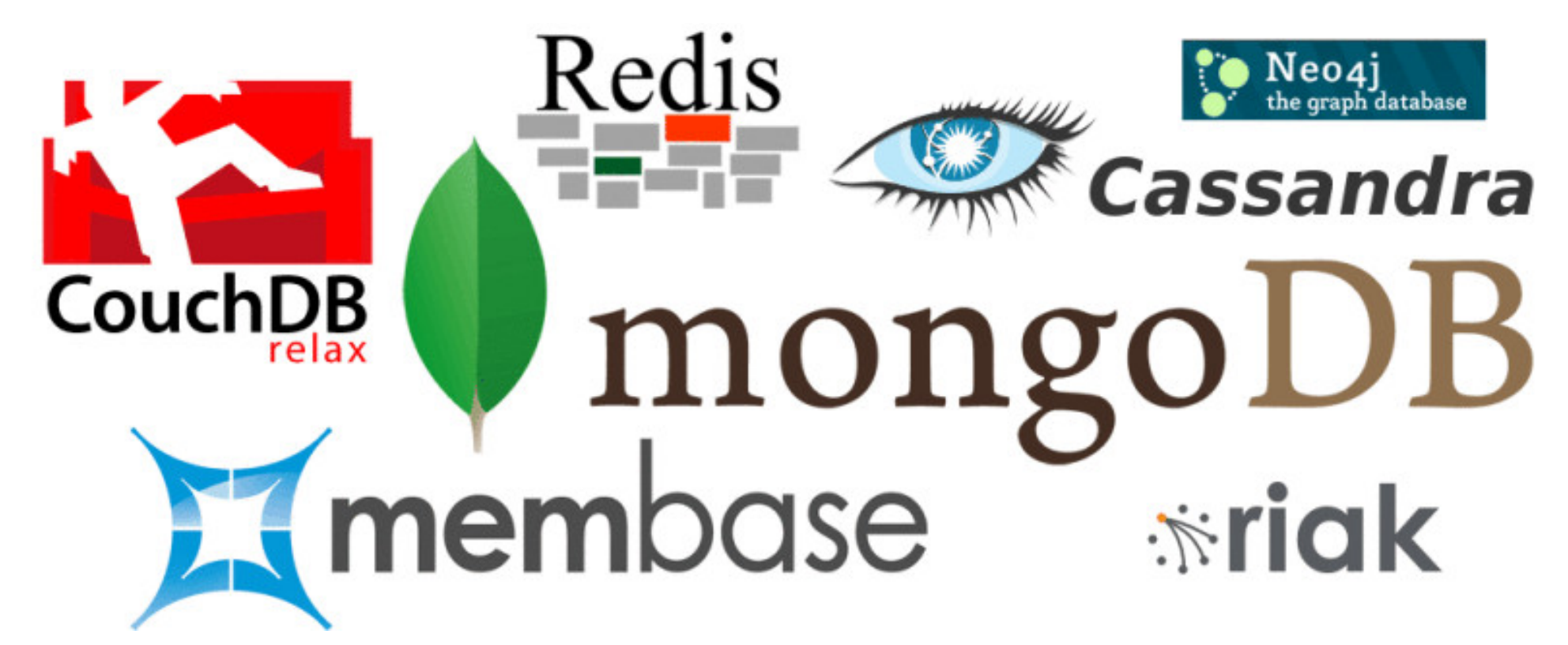

Figure 2: NoSQL use is growing every day

Document stores

Also referred to as document-oriented database, a document store allows the inserting, retrieving, and manipulating of semi-structured data. Most of the databases available under this category use XML, JSON, BSON, or YAML, with data access typically over HTTP protocol using RESTful API or over Apache Thrift protocol for cross-language interoperability. compared to RDBMS, the documents themselves act as records (or rows), however, it is semi-structured as compared to rigid RDBMS. they provide dynamic or changeable schema or even schema-less documents. because of the limitless flexibility provided in this model, this is one of the more popular models implemented and used.
Examples : MongoDB, CouchDB, Terrastore, BaseX

Key-value stores

A Key-value store is very closely related to idea of a map in computer science, they are actually distributed hash-tables that store data blobs in arbitrary sizes with unique keys over some distributed buckets. the key part of data usually can be chosen to be of type of string or numerical value but the value section of data - as said earlier - can be almost anything and is allowed to be very large in size. these databases reach very high performances and are very simpler to design and develop compared to others of their kin. they are optimized for querying against keys, As such they serve great in-memory caches. data usually is accessed via RESTful API(like in Document stores) and only the CRUD operations are supported by the system per-se most of the time.
Examples : Redis, Voldemort , Riak, MemcacheDB

Graph stores

Graph databases represent a special category of NoSQL databases where relationships are represented as graphs. the relationships represented may include social relationships between people, transport links between places, or network topologies between connected systems. they can be considered as special purpose NoSQL databases optimized for relation-heavy data. If there is no relationship among the entities, there is no use-case for graph databases. graph databases are the **most complex** type of NoSQL databases to build. there is no standard query language yet but graph traversal languages like **Gremlin** could be used to describe operations over stored graphs.
Examples : FlockDB, Neo4j, Titan

Features	Column	Document	Key-Value	Graph
Table-like schema	yes	no	no	yes
Complete update	yes	yes	yes	yes
Partial update	yes	yes	yes	no
Aggregates across rows	yes	no	no	no
Relationships among entities	no	no	no	yes
Cross-entity view support	no	yes	no	no
Query	yes	yes	no	yes
Batch data update	yes	yes	yes	no
Batch data fetch	yes	yes	yes	yes

Table 1: NoSQL common feaures comparison

Multi-type storages

There are some intriguing and exceptional example databases that can be categorized differently, mostly they can support multiple data storage plans. some examples are:

- **OrientDB**: Supports document storage, key-value as well as graph.
  The official website is http://www.orientdb.org.
- **ArangoDB**: Universal database with support for document store, key-value and graph models.
  Official website is http://www.arangodb.org.
- **Aerospike**: A hybrid between RDBMS and NoSQL store. it supports document store, key-value, graph as well as RDBMS.
  Source code can be found at https://github.com/JakSprats/Alchemy-Database.

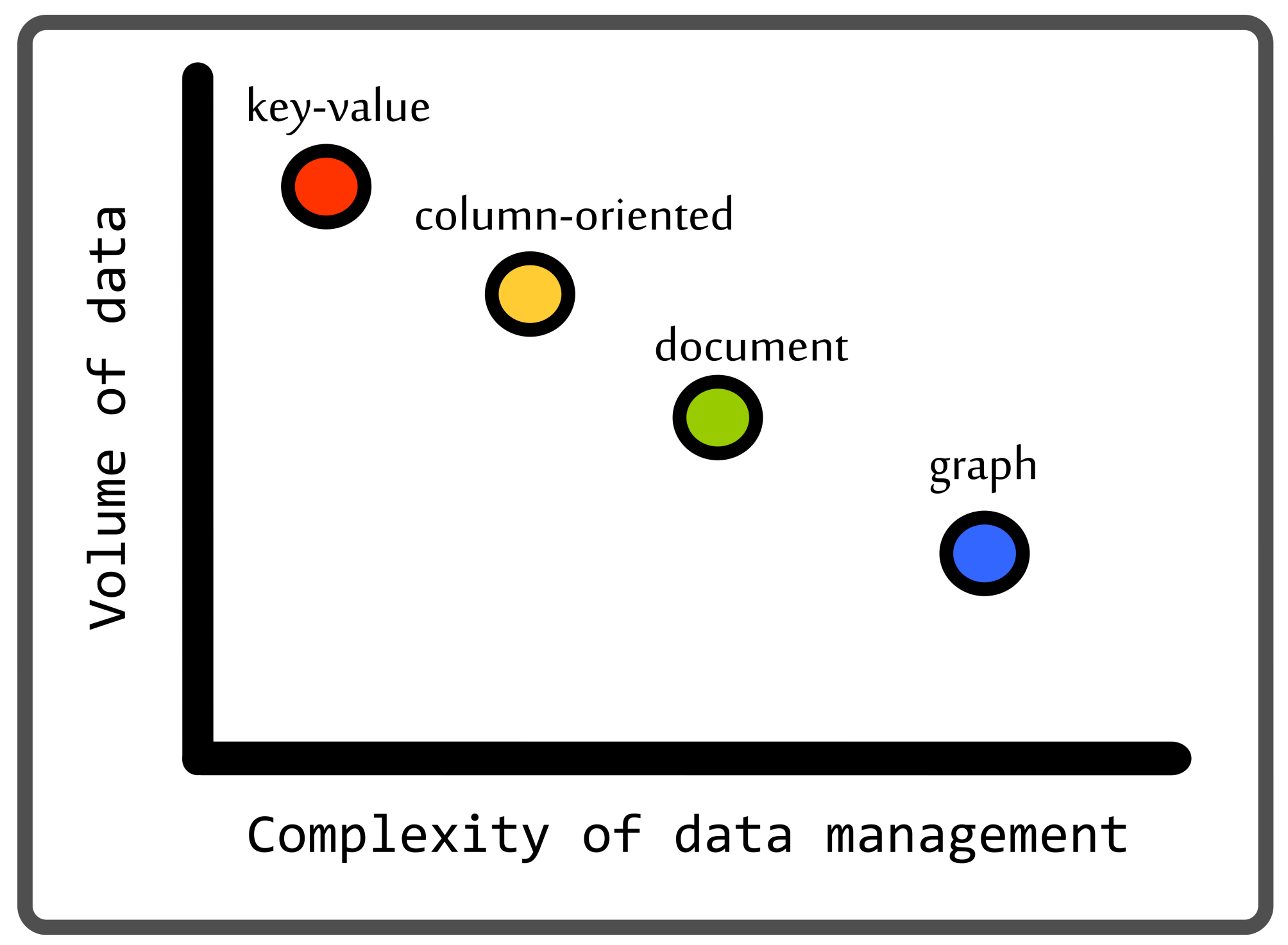

Figure 3: Data size vs Data complexity in NoSQL family

References

[1] http://en.wikipedia.org/wiki/cap_theorem, May 2014.
[2] http://en.wikipedia.org/wiki/nosql, May 2014.
[3] C. Mohan. History repeats itself: sensible and nonsensql aspects of the nosql hoopla. In *Proceedings of the 16th International Conference on Extending Database Technology*, pages 11–16. ACM, 2013.
[4] G. Vaish. *Getting Started with NoSQL*. Packt Publishing Ltd, 2013.