Tutorial 6: NoSQL & Spark

Li Yuan

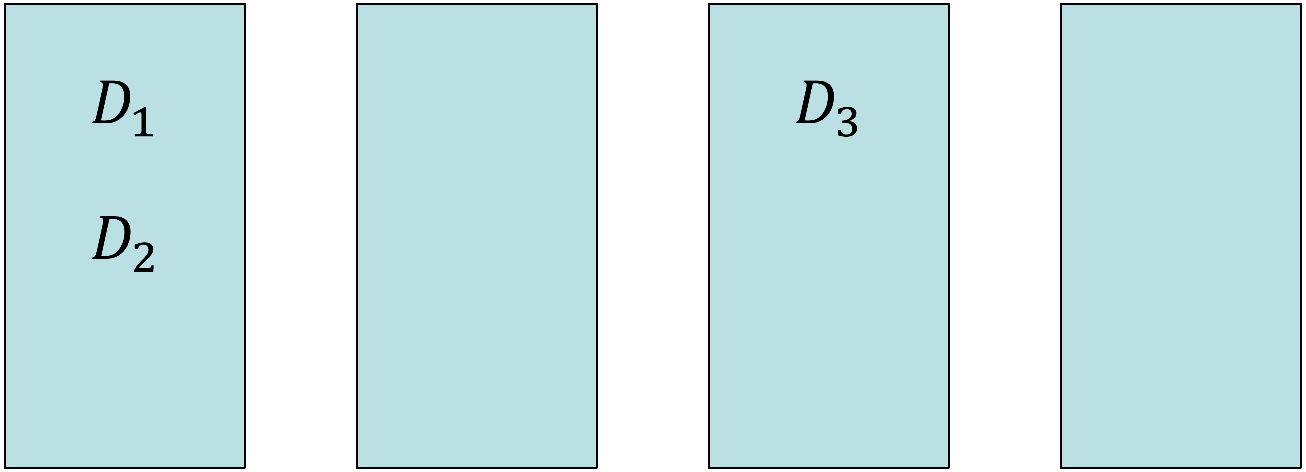
li.yuan@u.nus.edu

NUS
National University
of Singapore

Question 1a

MapReduce and the Google File System (GFS) were designed to work well together. What important optimization in MapReduce is enabled by having GFS expose block replica locations via an API?

Solution 1a



The MapReduce scheduler can arrange for Map tasks to execute on the same node that stores the data, avoiding a copy across the network.

Question 1b

List two features that are originally designed for relational databases and are now integrated into the MapReduce/Hadoop software stack.

- **High-level languages (like SQL)**
- **Index**
- **Column stores.**

Question 1c

What are the advantages of adding schema to MapReduce/Hadoop?

- **Improving the performance of parsing fields out of flat text files**
- **Schemas define a contract, decoupling logical from physical**

Question 1d

Why we need YARN since we already have reasonably efficient cluster management systems in MapReduce/Hadoop?

Yet-Another-Resource-Negotiator
YARN is a platform responsible for managing computing resources in clusters and using them for scheduling users' applications.

Support other distributed frameworks.

Question 2a

List three of the many development features in Spark that show the trend of convergence between relational databases and Spark.

- **schema**
- **query optimization**
- **high level languages**

Question 2b

In HDFS, each chunk is replicated for three times by default. In contrast, in Spark, RDD uses lineage for reliability. What are the major problems if Spark also uses replications for reliability?

Consume a lot of memory

Question 2c

Is it true that in the Spark runtime, RDD cannot reside in the hard disk?

False. RDD can also be in the disk if out of memory.

Question 3

NoSQL databases have been a hot research topic. Reading the below paper will help you to have an overview on the NoSQL databases. *Rick Cattell. 2011. Scalable SQL and NoSQL data stores. SIGMOD Rec. 39, 4 (May 2011), 12-27.*

After reading the paper, answer the following questions:

Question 3a

Compare ACID and BASE. Why do NoSQL systems choose BASE?

BASE = Basically Available, Soft state, Eventually consistent

ACID = Atomicity, Consistency, Isolation, and Durability

Solution 3a

Basically Available: This constraint states that the system does guarantee the availability of the data as regards CAP Theorem; there will be a response to any request. But, that response could still be ‘failure’ to obtain the requested data or the data may be in an inconsistent or changing state, much like waiting for a check to clear in your bank account.

Soft state: The state of the system could change over time, so even during times without input there may be changes going on due to ‘eventual consistency,’ thus the state of the system is always ‘soft.’

Eventual consistency: The system will *eventually* become consistent once it stops receiving input. The data will propagate to everywhere it should sooner or later, but the system will continue to receive input and is not checking the consistency of every transaction before it moves onto the next one. Werner Vogel’s article “Eventually Consistent – Revisited” covers this topic is much greater detail.

Solution 3a

Atomicity: Each transaction be "all or nothing”.

Consistency: Any transaction will bring the database from one valid state to another.

Isolation: Concurrent execution of transactions results in a system state that would be obtained if transactions were executed sequentially.

Durability: Once a transaction has been committed, it will remain so, even in the event of power loss, crashes, or errors.

Solution 3a

The idea is that by giving up ACID constraints, one can achieve much higher performance and scalability.

Question 3b

If we plan to use relational databases for supporting the following workloads, what are the problems? Or, why do we need specialized engines for each workload?

1) key-value stores,

2) document stores.

Performance and flexibility.

Question 3c

Case studies. Would you please give some examples/case studies on using the NoSQL databases mentioned in the paper, rather than relational databases?

MongoDB: High volume data feeds, Metadata management, content management.

SimpleDB: Web Services for Connected Systems, Low-Usage Application

Acknowledgement

Thanks to Li Qinbin for making these slides.

liqinbin@u.nus.edu