MapReduce Vs Spark

CS5229 - Big Data Analytics Technologies

Name: S. A. P. Jayatilake
Index No: 239321L

MapReduce

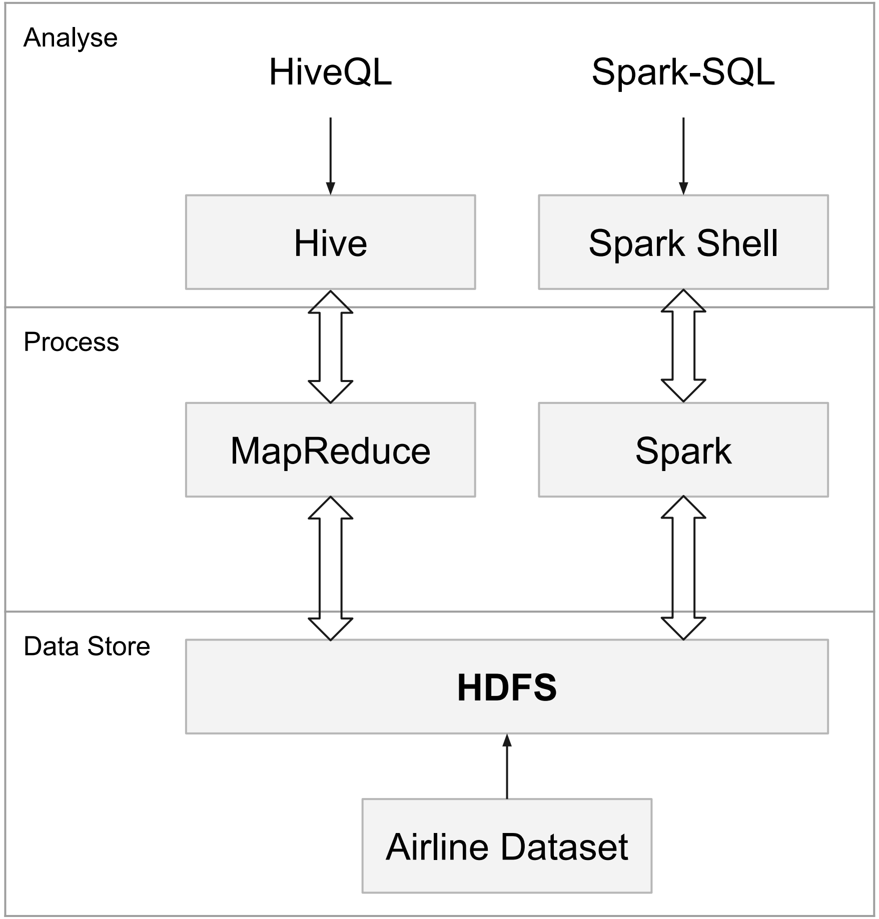
- A framework for processing large datasets with a parallel, distributed algorithm on a cluster.
- Introduced in December 2004 by Google
- MapReduce in Hadoop is a **divide and conquer** strategy
- Main advantages of MapReduce algorithm
  - Parallel processing → Fast processing
  - Data Locality (moving processing unit to the data) → Save network bandwidth
- Hadoop MapReduce simplifies writing parallel distribute applications by handling all of the logic
  - You have to provide only the Map and Reduce functions.
  - **Map**: maps data to sets of key-value pairs called intermediate results
  - **Reduce**: combines the intermediate results, applies additional algorithms, and produces the final output
- Multiple frameworks are available for MapReduce
  - E.g.: Hive: Open-source, SQL-like data warehouse solution which automatically generates Map and Reduce programs.

Apache Spark

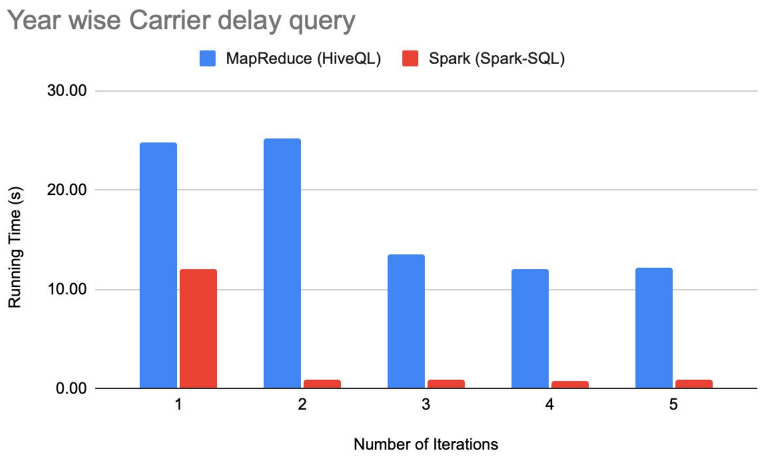
- An open-source, distributed processing framework that was created to address the limitations of MapReduce
- Uses **in-memory caching** and **optimized query processing**
- Supports **code reuse** across multiple workloads
  - Spark reuses data by using an in-memory cache to speed up ML algorithms that repeatedly call a function on the same dataset.
  - This is accomplished by creating DataFrames which are a collection of objects that are cached in-memory and reused in multiple Spark operations.
  - **⇒ dramatically lowers the latency**
- reduces the number of steps in a job
  - With Spark, only one step is needed, where data is read into memory, operations are performed, and the results are written back
  - **⇒ much faster processing.**

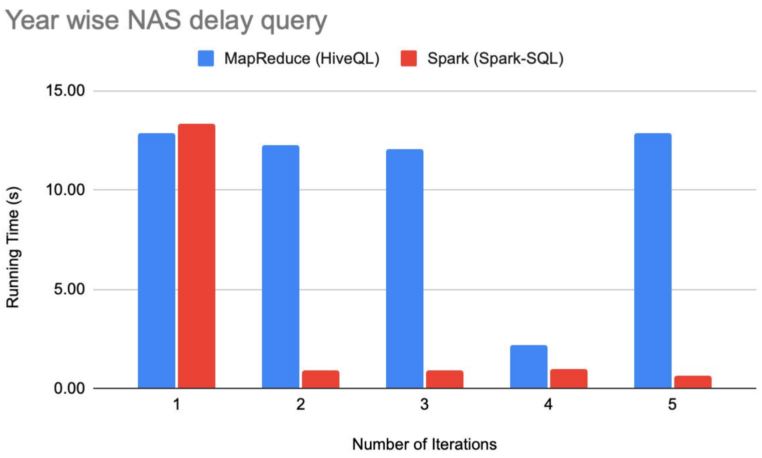
Demo

- Analyze the Airline Delay dataset using MapReduce and Spark
  - Year wise carrier delay from 2003-2010
  - Year wise NAS delay from 2003-2010
  - Year wise Weather delay from 2003-2010
  - Year wise late aircraft delay from 2003-2010
  - Year wise security delay from 2003-2010

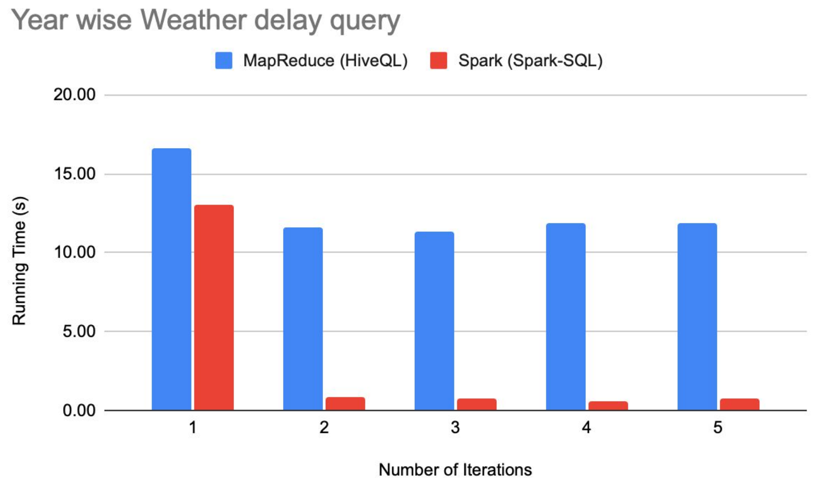


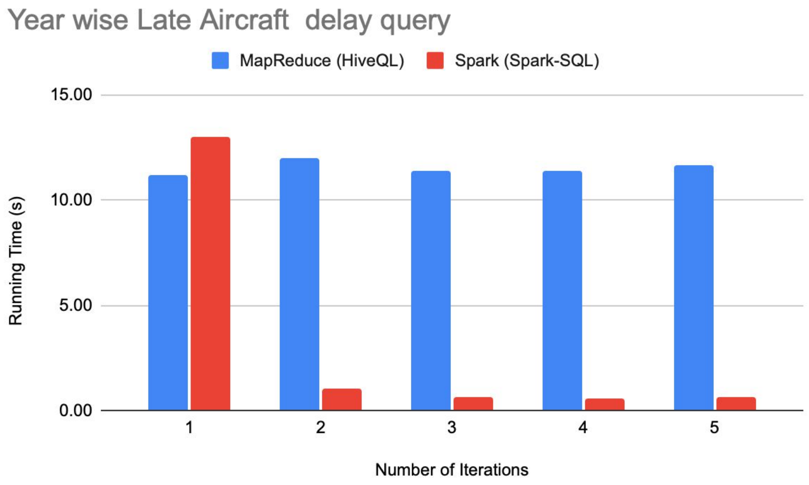
Results: Query Execution time for 5 iterations



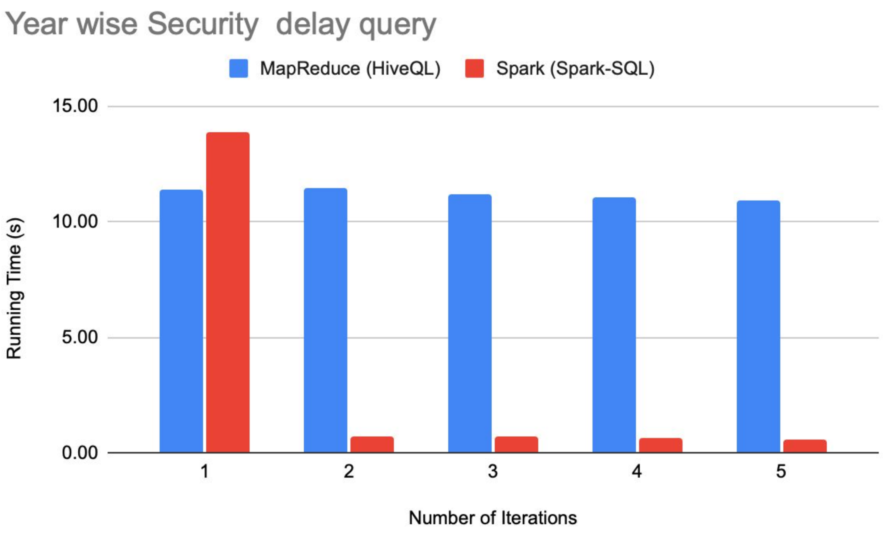


Results: Query Execution time for 5 iterations (Ctd.)

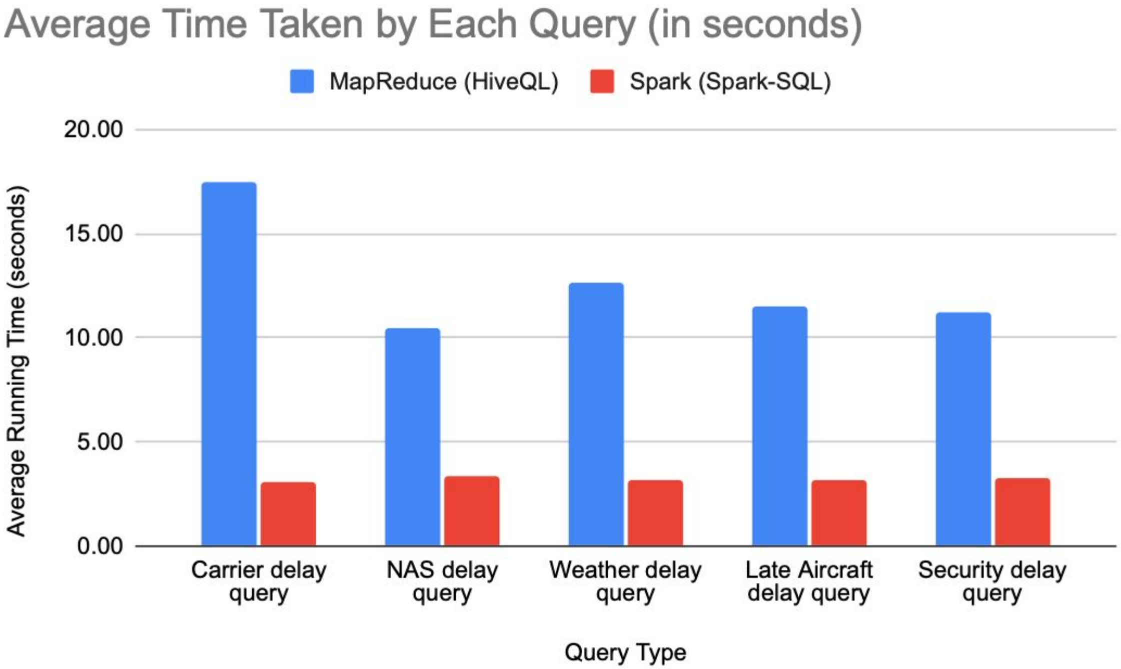




Results: Query Execution time for 5 iterations (Ctd.)



Results: Average Time Taken by Each Query



MapReduce Vs Spark

	MapReduce	Spark
Ease of Use	● more difficult to program ○ developed in Java and difficult to program. ○ no interactive mode ○ However, Hive provides a command-line interface.	● more user-friendly and features an interactive mode ○ contains APIs for Scala, Java, and Python and Spark SQL for SQL users ○ offers basic building blocks that allow users to easily develop user-defined functions ○ Can make use of Apache Spark interactive mode when running commands to get an instant response
Fast Processing	● does not provide data caching, ○ other services can assist it with little to no performance downturn since it terminates its operations the moment they are complete. ● has to persist data back to the disk after every Map or Reduce action.	● runs 100 times faster in memory and 10 times faster on disk than Hadoop MapReduce ○ since it processes data in memory (RAM). ● takes a lot of RAM to operate effectively. ○ Spark saves processes to memory and keeps them there if different instructions are not given. ○ If Spark is used with other resource-demanding services, its performance may be hampered notably. ○ Additionally, Spark's performance will suffer if the data sources are too large to fit fully in memory.

Conclusion

- When it comes to performance, MapReduce and Spark both have benefits.

- Spark is better solution for your big data requirements
  - if your data get accommodated with the amount of memory space you have or
  - if you have a dedicated cluster.

- MapReduce is a better solution
  - if you have a large volume of data that won't fit neatly into memory, and
  - you need your data framework for coordinating with other services.

Thank You!