Open source Google-style large scale data analysis with Hadoop

Ioannis Konstantinou

Email: ikons@cslab.ece.ntua.gr
Web: http://www.cslab.ntua.gr/~ikons

Computing Systems Laboratory
School of Electrical and Computer Engineering
National Technical University of Athens
Big Data

- Facebook: 20TB/day compressed
- CERN/LHC: 40TB/day (15PB/year)
- NYSE: 1TB/day
- 2009 Digital Universe: 800,000 Petabytes or 0.8 Zettabytes
- Moore's Law: Data doubles every 18 months
- 2020 prediction: 35 Zettabytes (44 times bigger than 2009)
What is Hadoop?

• It’s a distributed framework for **large-scale data processing**:

• Inspired by Google’s architecture: Map Reduce and Google File System

• Can scale to thousands of nodes and petabytes of data

• A top-level Apache project (since 2008) – Hadoop is open source

• Written in Java, plus a few shell scripts
Why Hadoop?

• Hadoop is designed to run on cheap commodity hardware
• Fault-tolerant hardware is *expensive*
• It automatically handles data replication and node failure
• It does the hard work – you can focus on processing data
When to use Hadoop?

• There is access to lots of commodity hardware
• The processing can be easily parallelized
• Need to process lots of unstructured data
  – Data intensive applications
• It is ok to run batch jobs (no need for interactive results)
Architecture

• HDFS: Distributed file system
  – Hard to store a PB
  – Based on Google FS
  – Fault-tolerant: handles replication, node failure, etc

• MapReduce: Data aware parallel computation framework
  – Even harder to process a PB
  – Based on a research paper by Google
Hadoop Distributed File System 1/2

• Master/Slave Architecture
• Files are split into one or more blocks and these blocks are stored in a set of DataNodes
• A Master NameNode
  – a master server that manages the file system namespace and regulates access to files by clients
  – determines the mapping of blocks to DataNodes
• Many DataNodes
  – Serve client read/write requests
  – Create/delete/replicate blocks
Hadoop Distributed File System 2/2

HDFS Architecture

Namenode

Metadata (Name, replicas, ...): /home/foo/data, 3, ...

Client

Read

Datanodes

Replication

Write

Rack 1

Block ops

Datanodes

Blocks

Rack 2

Metadata ops

Client
MapReduce 1/3

• A programming model
• A software framework
• for writing applications that
  – rapidly process vast amounts of data in parallel
  – on large clusters of compute nodes
MapReduce 2/3

• Problem is separated in two different phases, the Map and Reduce phase.

• **Map**: Non overlapping chunks of input data is assigned to separate processes (mappers) that emit a set of intermediate results

• **Reduce**: Map results are fed to a usually smaller number of processes called reducers that “summarize” their input in a smaller number of results
MapReduce 3/3

Diagram showing the flow of data through Map, Shuffle/Sort, and Reduce processes, leading to output.
When should I use it?

• Good choice for
  – Indexing log files
  – Sorting vast amounts of data
  – Image analysis

• Bad choice for
  – Figuring $\pi$ to 1,000,000 digits
  – Calculating Fibonacci sequences
  – MySQL replacement
Hadoop MapReduce

• Master/Slave architecture

• A JobTracker Master
  – Runs together with NameNode
  – Receives client job requests
  – Schedules and monitors MR jobs
    • Move computation near the data
    • Speculative execution

• Many TaskTrackers
  – Run together with DataNodes
  – Perform I/O operations with DataNodes
Typical problems

• Log and/or clickstream analysis of various kinds
• Marketing analytics
• Machine learning and/or sophisticated data mining
• Image processing
• Processing of XML messages
• Web crawling and/or text processing
• General archiving, including of relational/tabular data, e.g. for compliance
Use cases 1/3

- Large Scale Image Conversions
- 100 Amazon EC2 Instances, 4TB raw TIFF data
- 11 Million PDF in 24 hours and 240$
- Internal log processing
- Reporting, analytics and machine learning
- Cluster of 1110 machines, 8800 cores and 12PB raw storage
- Open source contributors (Hive)
- Store and process tweets, logs, etc
- Open source contributors (hadoop-lzo)
Use cases 2/3

• 100.000 CPUs in 25.000 computers
• Content/Ads Optimization, Search index
• Machine learning (e.g. spam filtering)
• Open source contributors (Pig)

• Natural language search (through Powerset)
• 400 nodes in EC2, storage in S3
• Open source contributors (!) to HBase

• ElasticMapReduce service
• On demand elastic Hadoop clusters for the Cloud
Use cases 3/3

- ETL processing, statistics generation
- Advanced algorithms for behavioral analysis and targeting
- Used for discovering People you May Know, and for other apps
- 3X30 node cluster, 16GB RAM and 8TB storage
- Leading Chinese language search engine
- Search log analysis, data mining
- 300TB per week
- 10 to 500 node clusters
Questions