

Elephants and Storms

**Using Big Data techniques
for Analysis of Large and
Changing Datasets**

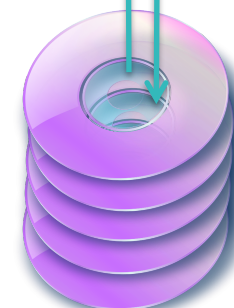
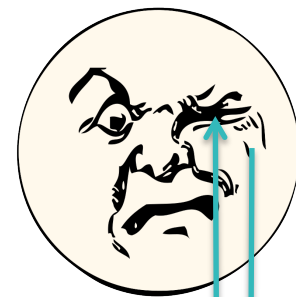
Geoff Low

Introduction

- Data, data, data.... BIG DATA
- MapReduce – batch-based processing
- Storm – stream-based processing
- The Lambda Architecture
- Case Studies

Data.... so much data

- Data is generated at an astounding rate:
 - In 2012, 2.5 quintillion bytes of data were generated
EVERY DAY
 - 90% of the worlds data was generated in the 2 years previous to 2012
 - **Rate** of data generation is expected to double by 2015
- Number of studies registered with clinicaltrials.gov in 2012
 - 19498 (registered btw 2012-01-01 and 2013-01-01)
 - Average number of subjects = 1590 (!!)
 - Assume
 - 1Mb of data per subject/study as capture
 - 2Mb of data per subject/study for tabulation
 - ➔ Conceivably 93 Tb of study data (plus audits, etc)

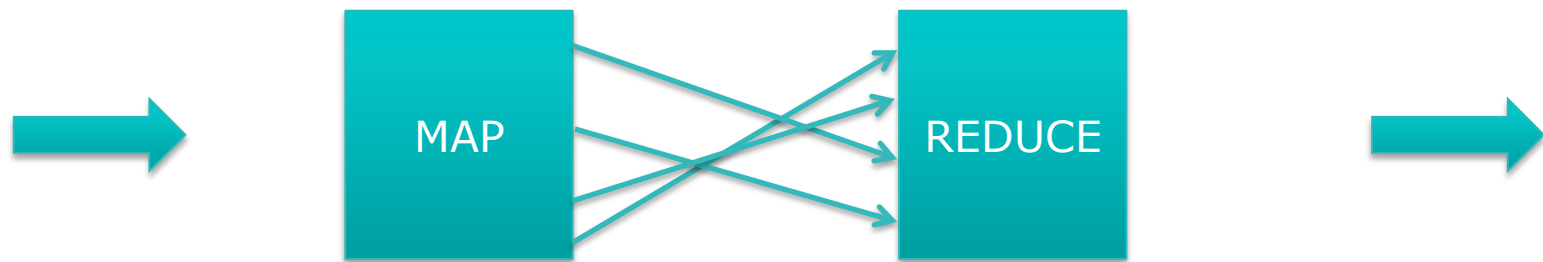


What is “Big Data”?

- A chameleon term
 - Many things to many people
- Is it....
 - Large quantities of data?
 - Dealing with large quantities of data?
 - A reason to get a new server?
- Werner Vogels (CTO Amazon)
 - “The collection and analysis of large amounts of data to create a competitive advantage.”
 - “When your data sets become so large that you have to start innovating how to collect, store, organize, analyze, and share it.”
- Two approaches for innovation (analysis)
 - Batch-based processing
 - Stream-based processing

MapReduce – batch-based processing

- “Programming model for processing large datasets using a distributed algorithm on a cluster”
- Originates in Google (published in 2004)
 - Building Indexes/Reverse Indexes
 - PageRank
- Two procedures
 - Map – sort or filter the data
 - Reduce – summarise the mapped data



MapReduce – batch-based Processing

- “Big Iron” versus “Big Data”



MapReduce - Implementations



- Apache Hadoop
 - Originated in Yahoo
 - Implements design from Google paper
- Execution details
 - Master node
 - Coordinates tasks, nodes, status, failure recovery
 - Mapper/Reducer nodes
 - Each node can be a mapper or a reducer (or both)
 - Ship data to/from mapper and reducer
 - Distributed File System
 - Hadoop File System (HDFS)

Example – Word Count

“I have ten books and I want to get a count of the number of times each unique word appears”

- Parallelisation

- I can read the ten books, and keep a tally
- I can pick ten of you to read a book each, return the tally to me and I sum it up
- I can pick ten of you to take a book each, and then you can split the book into sentences and send each sentence to another person to do the count, who return the tallies to me to sum up.

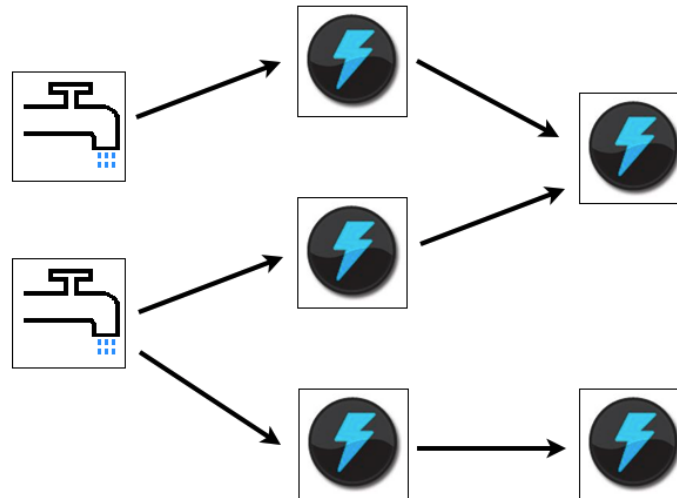
- MapReduce

- I assign a sentence to each of you
- Each time you read a word in the sentence, you put a token of paper in a unique bucket with the word on the front
- I assign each of you a Bucket
- You count the number of pieces of paper in the bucket



Storm – Stream-based Processing

- Stream-based processing operates continuously on a stream of data
- Storm was created at Twitter and is described as a real-time computation engine.
- Spouts, Bolts and Topologies



Stream-based processing - Example

- Listen for mentions of bands on the radio
- Each person is assigned to a radio station
 - Every time they hear their band mentioned they put a piece of paper in a bucket
- Listen for mentions of bands on the radio and track the gender of the mentioner
 - Every time the name is mentioned, put a piece of paper in a bucket (bearing M, F, UNK)
- At any point you can get the count, by counting pieces of paper in the different buckets.
- Have no way of generating aggregated metrics



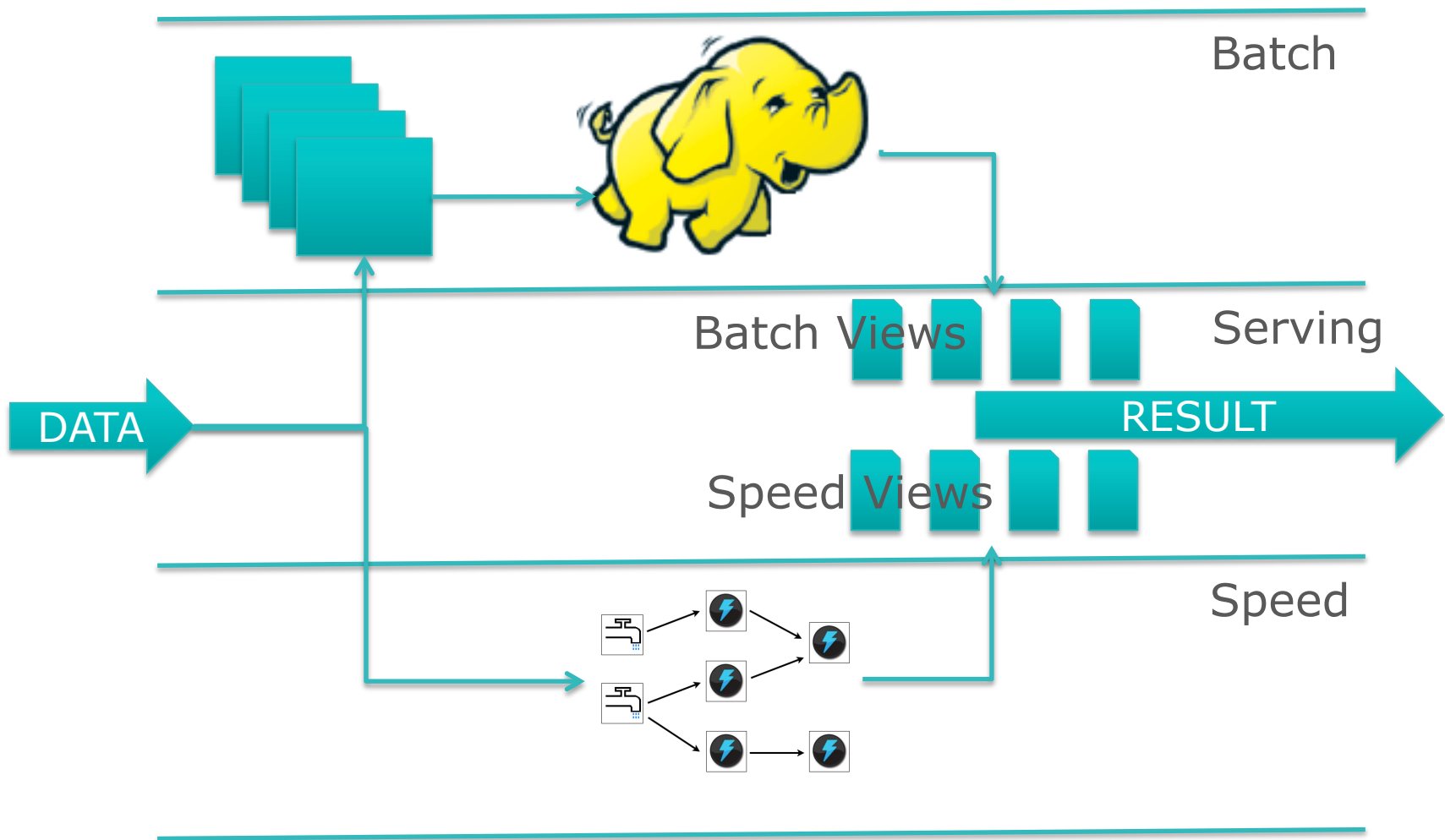
The Lambda Architecture

query = fn(all data)

The Lambda Architecture cont.

- We have two approaches:
 - Batch-based processing (THEN)
 - Stream-based processing (NOW)
- We want to be able to query on both NOW and THEN to give us results that represent everything we are interested in
- Lambda Architecture allows us to combine the two approaches into one
 - We have a single platform for writing queries across all our data
- So, how do we tie this together?

The Lambda Architecture



Benefits of the Lambda Architecture

- Best of both worlds
 - Batch-based processing for continual updating of pre-computed views
 - Stream-based processing to include data subsequent
- Source data remains unaltered
 - Not storing a processed view
 - Can generate any queries in the future, as all the data remains
- Resilient to hardware error
 - Distributed file systems, redundancy built in.
- Resilient to user error
 - Views (result of human input) are entirely disposable, if there is a mistake then the batch and speed views can be flushed.

So, what might stop us, and save us

- Completely different paradigm
 - Need to properly analyse your problem and think about how to approach it
- Programming skills are valuable for defining Mapper/Reducer/Bolt/Topology
 - SQL not directly supported
 - Many different languages supported (Clojure, Java, Python, etc)
- Very tunable
 - Cluster can be resized
 - Cluster can be reconfigured
- Very powerful
 - Terasort benchmark

Sample Use Case – Subject Identification

- Social media is a patient empowerment “powerhouse”
 - We should be able to help connect patients to studies
 - If invited to participate in a study:
 - 32% would be very willing to participate in a cancer clinical trial
 - 38% would be inclined to do so
 - Traditionally done through patient groups
- We can filter tweets using Storm looking for patterns
 - Such as indicatory diagnoses
- We can then attempt to filter matching tweets
 - Male
 - Female
 - Age group
 - Location
- Engage the tweeter

Sample Use Case – Whole Study Status

- Using a Distributed File System + Lambda Architecture
 - Can store all study data... for all studies (including audits, site information, investigator)
 - Compute standard looks across study data for all time
- Want to change the reporting metrics?
 - Change the program, rebuild all the metrics for all data
- Want to create new metrics?
 - Create the program, batch-views are updated
- Time-based metrics
 - Can derive metrics on an hourly, daily, weekly, monthly, etc basis

Conclusions

- However we want to refer to it, big data is a current problem
 - But it is a problem which we can innovate about
- We owe it to the patients to maximise the value of the data we collect (and have already collected)
 - Big data approaches expand the looks we can take with our data
- We may need to adopt some new approaches to our data
- Big Data techniques can be very effective for processing large quantities of data
 - Make sure the problem you are trying to solve suits it

Any Questions?

