Hadoop – A Perfect Platform for Big Data and Data Science
Agenda

- Data Explosion
- Data Economy
- Big Data Analytics
- Data Science
- Historical Data Processing Technologies
- Modern Data Processing Technologies
- Hadoop Architecture
- Key Principles Hadoop
- Hadoop Ecosystem
Presentation Goal

- To give you a high level of view of Big Data, Big Data Analytics and Data Science
- Illustrate how Hadoop has become a founding technology for Big Data and Data Science
Data Explosion
“Big” Data in the News
Data Creation

• Visually Illustrates how much data is generated per minute.

Source: http://schoollibrarybeyondsurvival.files.wordpress.com/2012/06/dataneversleeps_4fd61ee2eda5a.jpg
### Why so much data is being Generated today?

<table>
<thead>
<tr>
<th>1975</th>
<th>Today</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000 “online” users = End Point</td>
<td>2000 “online” users = Start Point</td>
</tr>
<tr>
<td>Static User Population</td>
<td>Dynamic user population</td>
</tr>
<tr>
<td>Business Process Automation</td>
<td>Business Process Automation</td>
</tr>
<tr>
<td>Highly structured data records</td>
<td>Structured, semi-structured and unstructured data</td>
</tr>
<tr>
<td>Data networking in its infancy</td>
<td>Universal high-speed data networks</td>
</tr>
<tr>
<td>Centralized computing (Mainframes and minicomputers)</td>
<td>Distributed computing (Network servers and virtual machines)</td>
</tr>
</tbody>
</table>
• **Existing OLTP Databases**
  – Organizations have several OLTP databases for the various products and services they offer

• **User Generated Data**
  – Many social networking, blogging sites allow for users to generate their own data
    • Blogs, tweets, links
    • Videos, audios

• **Logs**
  – Enterprise and Internet scale applications may have several servers that generate log files
    • Ex. Access log files

• **System generated data**
  – Many services inside an enterprise generate syslogs that may have to be processed
Data in Personal Computing

- Let’s compare my PC from 1984

<table>
<thead>
<tr>
<th></th>
<th>My PC in 1984</th>
<th>My PC today</th>
<th>Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Speed</td>
<td>1MHz</td>
<td>3GHz</td>
<td>3000</td>
</tr>
<tr>
<td>Ram</td>
<td>256K</td>
<td>4GB</td>
<td>15,000</td>
</tr>
<tr>
<td>Transmission Rate</td>
<td>30B/s</td>
<td>1MB/s</td>
<td>30,000</td>
</tr>
<tr>
<td>Hard Disk Capacity</td>
<td>1MB</td>
<td>1TB</td>
<td>1,000,000</td>
</tr>
</tbody>
</table>
Data Volumes are Growing
Data Economy
The Revolution in the Marketplace – The Shift

Hardware Was King

Software Becomes King

Data is the New King
What are Data Driven Organizations?

A data driven organization that acquires data, that processes data, and leverages data in a timely fashion to create efficiencies, iterate on and develop new products and navigate the competitive landscape.
Data Driven Organizations Use Data Effectively

Data Driven Organizations

Organizations that use Data to augment their Business
Big Data Business is Big Business

- Data Generators + Aggregators
- Data Aggregators + Enablers
- Big Data BI + Data Analytics sw + Tool Vendors
- Big Data Cloud Platform Software
- Big Data public Cloud Platform Providers (hw+sw)
- Big Data Hardware Manufactures

- Data as a Service

- Platforms (PAAS)

- Infrastructure as a Service (IaaS)
# Information Science Is Affecting Every Industry

<table>
<thead>
<tr>
<th>Biotech/Healthcare</th>
<th>Linguistics</th>
<th>Mining</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="http://www.youtube.com/watch?v=1-C0Vtc-sHw" alt="Biotech/Healthcare" /></td>
<td><img src="http://www.youtube.com/watch?v=VwgkT34g61w" alt="Linguistics" /></td>
<td><img src="http://www.cloudera.com/blog/2012/01/seismic-data-science-hadoop-use-case/" alt="Mining" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Finance</th>
<th>Journalism/Visualization</th>
<th>Education</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="http://www.youtube.com/watch?v=1-C0Vtc-sHw" alt="Finance" /></td>
<td><img src="http://www.youtube.com/watch?v=GWF2SU7UWs8" alt="Journalism/Visualization" /></td>
<td><img src="http://www.youtube.com/watch?v=uuUa4FEGvzo" alt="Education" /></td>
</tr>
</tbody>
</table>
Wake up - This is a Data Economy!

• We are in the midst of Information Science in the making.

• Not long ago data was expensive. There wasn’t much of it. Data was the bottleneck for much of human endeavor.

• No limit to how much valuable data we can collect!

• We are no longer data-limited, but insight limited. The people who know how to work with data are in short supply.
Big Data Analytics
What is Data?

**da-ta**  *noun pl but singular or pl in constr, often attributive* ˈdā-tə, ˈda- also ˈdā-

- Factual information (as measurements or statistics) used as a basis for reasoning, discussion, or calculation
- Information output by a sensing device or organ that includes both **useful** and **irrelevant** or **redundant** information must be processed to be meaningful
- Information in numerical form that can be digitally transmitted or processed

source: http://merriman-webster.com
Big Data Characteristics (Three Vs)

- **Volume**
  - Data volume on the rise
  - $44x$ increase from 2010 to 2020
    - Expected to go from 1.2 zetabytes to 35.2 zb

- **Velocity**
  - Speed at which the data needs to be transformed and processed is essential

- **Variety**
  - Greater variety/types of data structures to mine
    - Structured
    - Semi-structured
Big Data Characteristics: Data Structures

- **Structured**
  - Data containing a defined data type, format, structure
  - Example: Transaction data in OLTP and OLAP

- **Semi-Structured**
  - Textual data with discernable pattern, enabling parsing
  - Example: XML data files that are self describing by xml schema

- **“Quasi”Structured**
  - Textual data with erratic data format, can be formatted with effort tools and time
  - Example, web clickstream data that may have some inconsistencies in data values and formats

- **Unstructured**
  - Data that has no inherent structure and is stored as different types of files
  - Text documents, PDFs, images, video
Business Drivers for Analytics

- Many business Problems provide opportunities for organizations to become more analytical & data driven

<table>
<thead>
<tr>
<th>Driver</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desire to optimize business operations</td>
<td>Sales, pricing, profitability, efficiency</td>
</tr>
<tr>
<td></td>
<td>Example: amazon.com, Walmart</td>
</tr>
<tr>
<td>Desire to identify business risk</td>
<td>Customer churn, fraud, default</td>
</tr>
<tr>
<td></td>
<td>Example: insurance, banking</td>
</tr>
<tr>
<td>Predict new business opportunities</td>
<td>Upsell, cross-sell, best new customer prospects</td>
</tr>
<tr>
<td></td>
<td>Example: amazon.com</td>
</tr>
<tr>
<td>Comply with laws or regulatory requirements</td>
<td>Anti-Money Laundering, Fair Lending, Basel II (Operational Risk Management in Banks)</td>
</tr>
<tr>
<td></td>
<td>Example: finance</td>
</tr>
</tbody>
</table>
## Traditional Data Analytics vs. Big Data Analytics

<table>
<thead>
<tr>
<th>Traditional Data Analytics</th>
<th>Big Data Analytics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Clean Data</strong></td>
<td>Clean Data/Messy Data/Noisy Data</td>
</tr>
<tr>
<td><strong>TBs of Data</strong></td>
<td>PBs of Data/Lots of Data/Big Data</td>
</tr>
<tr>
<td><strong>Often Know in advance the questions to ask</strong></td>
<td><strong>Often Don’t know all the questions I want to ask</strong></td>
</tr>
<tr>
<td><strong>Design BI/DW around questions I ask</strong></td>
<td><strong>?? ??</strong></td>
</tr>
<tr>
<td><strong>Architecture doesn’t lend for high computation</strong></td>
<td><strong>Need distributed storage and computation</strong></td>
</tr>
<tr>
<td><strong>Typically, answers are factual</strong></td>
<td><strong>Typically, answers are probabilistic in nature</strong></td>
</tr>
<tr>
<td><strong>Structured</strong></td>
<td><strong>Structured and Unstructured</strong></td>
</tr>
<tr>
<td><strong>Dealing 1-2 domain data sets</strong></td>
<td><strong>Dealing with dozens of domain data sets</strong></td>
</tr>
</tbody>
</table>
## Traditional Data Analytics vs. Big Data Analytics

<table>
<thead>
<tr>
<th></th>
<th>Traditional Data Analytics</th>
<th>Big Data Analytics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hardware</strong></td>
<td>Proprietary</td>
<td>Commodity</td>
</tr>
<tr>
<td><strong>Cost</strong></td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td><strong>Expansion</strong></td>
<td>Scale Up</td>
<td>Scale Out</td>
</tr>
<tr>
<td><strong>Loading</strong></td>
<td>Batch, Slow</td>
<td>Batch and Real-Time, Fast</td>
</tr>
<tr>
<td><strong>Reporting</strong></td>
<td>Summarized</td>
<td>Deep</td>
</tr>
<tr>
<td><strong>Analytics</strong></td>
<td>Operational</td>
<td>Operational, Historical, and Predictive</td>
</tr>
<tr>
<td><strong>Data</strong></td>
<td>Structured</td>
<td>Structured and Unstructured</td>
</tr>
<tr>
<td><strong>Architecture</strong></td>
<td>Physical</td>
<td>Physical or Virtual</td>
</tr>
<tr>
<td><strong>Agility</strong></td>
<td>Reactive</td>
<td>Proactive, Sense and Respond</td>
</tr>
<tr>
<td><strong>Risk</strong></td>
<td>High</td>
<td>Low</td>
</tr>
</tbody>
</table>
“Data is the oil of the 21st century”
- Gartner

“Data is the crude oil of the 21st century. You need data scientists to refine it!”
- Karthik
Data Science
What is Data Science?

Using (multiple) data elements, in clever ways, to solve iterative data problems that when combined achieve business goals, that might otherwise be intractable.
Data Science – Another Look!

Adapted from Drew Conway - http://www.drewconway.com/zia/?p=2378
Data Scientist

Languages
- Java
- Python
- Shell
- Perl

4GL
- Pig
- Hive
- R
- SQL

Machine Learning
- Mahout
- Weka

Visualization
- Gaffle
- HighCharts
- d3.js
- MatLib

Distributed Computing
- Hadoop
- HDFS
- Map/Reduce

Traditional DW
- Oracle DW

Data Scientist
What Makes a Data Scientist?

Data Scientist = Curiosity
+ Intuition
+ Data gathering
+ Standardization
+ Statistics
+ Modeling
+ Visualization
How do I become a Data Scientist?

• Some things you can do:
  – Learn about distributed computing
  – Learn about matrix factorizations
  – Learn about statistical analysis
  – Learn about optimization
  – Learn about machine learning
  – Learn about information retrieval
  – Learn about signal detection and estimation
  – Master algorithms and data structures


Enroll at JHU EP Program.
Take courses on Data Science and Big data
Online or Face to Face!!!
Going from Data ➔ Wisdom

Web Site Interaction = data
Normalized Data = Information
Knowledge

Knowledge parses, normalizes, and standardizes the data to transform it into information.

Information is then reported to yield knowledge.

Knowledge leads to insights.

Insights lead to wisdom.
Historical Data Processing Technologies
1977: CRAY-1A was used by NCAR (National Center for Atmospheric Research) to meet the needs of the atmospheric science community. Not in use any more.
Grid/Distributed Computing
RDBMS Computing

• Big Idea
  – Use a single server with attached storage for storing and processing data since they have to honor ACID properties

• Typically “scaled-up” (not scaling-out) by getting bigger/more powerful hardware

• Scale-out achieved by Sharding, Denormalizing, Distr. Caching, which have their own cons
  • Sharding requires you create and maintain schema on every server
  • Denormalizing loses some of the benefits of relational model
  • Distributed Cache suffers from “cold cache thrash”
Historical/Traditional technologies don’t work because …
All data cannot fit in a single machine and all processing cannot be done on a single machine.
Philosophy behind Distributed Computing

• “In pioneer days they used oxen for heavy pulling, and when one ox couldn’t budge a log, they didn’t try to grow a larger ox. We shouldn’t be trying for bigger computers, but for more systems of computers”

  - Grace Hopper, Computer Scientist and General in Navy
For Big Data Processing Scale is Important

Whatever system we choose, it has to scale for big data and big data processing and it has to be economical!
Big Data Computing in the Modern World
Modern Computing Benefits/Trends

• Faster processing (CPU) and more memory
  – Thanks to Moore’s law

• Storage has become cheaper
  – Organizations are buying more storage devices to deal with huge amounts of data

• Distributed systems design has matured
  – Hadoop movement, NoSQL movement

• Prevalence of Open-source software
  – A movement that started 20 years ago has yielded some of the best software, even better than proprietary software

• More and more Commodity hardware
  – Systems of commodity servers rather than supercomputers

• Public Cloud computing
  – Companies like Amazon, Google are providing cloud options
Modern Computing Challenges

- **Disks I/O is slow**
  - Servers typically use cost effective mechanical disks which are slow

- **Disks fail**
  - Wear and tear, manufacturing issues, stuff happens…

- **Not enough network bandwidth within data centers to move all the bits around**
  - Once the data is read, transmitting data within datacenter or across is slow
Solutions to Disk/IO Challenges

• Organizations use striping and mirroring together called RAID configuration
  – RAID stands for Redundant Array of Inexpensive disks
  – Use Striped (RAID0) and Mirrored (RAID1) configuration
Hadoop
Hadoop History Timeline

- 2002: Doug Cutting & Mike Cafarella started working on Nutch
  - Google publishes GFS & MapReduce papers

- 2004: Doug Cutting adds DFS & MapReduce support to Nutch

- 2005: Yahoo! hires Cutting, Hadoop spins out of Nutch

- 2006: NY Times converts 4TB of image archives over 100 EC2s

- 2007: Yahoo! fastest sort of a TB, 3.5 mins over 910 nodes

- 2008: Yahoo! fastest sort of a TB, 62 secs over 1,460 nodes
  - Sorted a PB in 16.25 hours over 3,658 nodes

- 2009: Cloudera Founded
  - Doug Cutting joins Cloudera
  - Facebook launches Hive: SQL Support for Hadoop
  - Hadoop Summit 2009, 750 attendees

Source: Cloudera, Inc.
What is Apache Hadoop?

- An open source project to manage “Big Data”
- Not just a single project, but a set of projects that work together
- Deals with the three V’s
- Transforms commodity hardware to
  - Coherent storage service that lets you store petabytes of data
  - Coherent processing service to process data efficiently
Key Attributes of Hadoop

• **Redundant and reliable**
  – Hadoop replicates data automatically, so when machine goes down there is no data loss

• **Makes it easy to write distributed applications**
  – Possible to write a program to run on one machine and then scale it to thousands of machines without changing it

• **Runs on commodity hardware**
  – Don’t have to buy special hardware, expensive RAIDs, or redundant hardware; reliability is built into software
**Hadoop – The Big Picture**

Unified storage provided by distributed file system called HDFS

Unified computation provided MapReduce distributed computing framework

Commodity Hardware

Hardware contains bunch of disks and cores
Hadoop Technology Stack

- Common Libraries/Utilities
- HDFS Distributed Storage
- MapReduce Distributed Processing
- YARN Distributed Processing
- HBase NOSQL DB
- YARN Frameworks
- Pig Script
- Hive Query

Ancillary Projects
- Ambari
- Avro
- Flume
- Oozie
- Zookeeper etc.

Core Hadoop Modules

Ancillary Projects
HDFS Architecture

Clients perform Metadata operations like create/delete file/dir and read metadata

Clients Read and Write Data from DataNode

NameNode performs block Operations on DataNode

DataNodes replicate data to each other
YARN Architecture

Applications Run Natively IN Hadoop

YARN (Cluster Resource Management)

HDFS (Distributed Reliable Storage)
HDFS + YARN
Key Principles Behind Hadoop Architecture
Key Principles behind Hadoop

• Break disk read barrier
• Scale-Out rather than Scale-UP
• Bring code to data rather than data to code
• Deal with failures
• Abstract complexity of distributed and concurrent applications
Break Disk Read Barrier

- **Storage capacity has grown exponentially but read speed has not kept up**
  - 1990:
    - Disk Store 1,400 MB
    - Transfer speed of 4.5MB/s
    - Read the entire drive in ~ 5 minutes
  - 2010
    - Disk Store 1 TB
    - Transfer speed of 100MB/s
    - Read the entire drive in ~ 2.5 hours

- **What does this mean?**
  - We can process data very quickly, but we cannot read fast enough, so the solution is to do parallel reads

- **Hadoop - 100 drives working at the same time can read 1TB of data in 2 minutes**

Scale-Out Instead of Scale Up

• **Harder and more expensive to scale-up**
  – Add additional resources to an existing node (CPU, RAM)
    Moore’s Law couldn’t keep up with data growth
  – New units must be purchased if required resources can not be added
  – Also known as scale vertically

• **Scale-Out**
  – Add more nodes/machines to an existing distributed application
    Software Layer is designed for node additions or removal
  – Hadoop takes this approach - A set of nodes are bounded together as a single distributed system
  – Very easy to scale down as well
Use Commodity Hardware

• “cheap” Commodity Server Hardware
  – Definition of “cheap” changes on a yearly basis
  – Today, it would cost about $5000
    • 32GB RAM, 12 1 TB hard drive, quad core CPU
• No need for super computers with high-end storage, use commodity unreliable hardware
  – Not desktops!

NOT

Super-computers with high end storage

BUT

Rack of Commodity Servers
Googles’s Original Chalkboard Server Rack
Data to Code = Not fit for Big Data

- Traditionally Data Processing Architectures divided systems into process and data nodes
  - Risks network bottleneck
Code to Data

- **Hadoop collocates processors and storage**
  - Code is moved to data (size is tiny, usually in KBs)
  - Processors execute code and access underlying local storage
Deal With Failures

• Given a large number machines, failures are common
  – Large warehouses see machine failures weekly or even daily
  – Example
    • If you have hardware whose MTTF (Mean Time to Failure is once in 3 years), if you have a 1000 machines, you will see a machine fail daily

• Hadoop is designed to cope with node failures
  – Data is replicated
  – Tasks are retried
Abstract Complexity

• Abstracts complexities in developing distributed and concurrent applications
  – Defines small number of components
  – Provides simple and well defined interfaces of interactions between these components

• Frees developer from worrying about system-level challenges
  – race conditions, data starvation, processing pipelines, data partitioning, code distribution, etc...

• Allows developers to focus on application development and business logic
Hadoop Ecosystem
Hadoop Technology Stack

- Apache HBase
- Tez
- Hive
- Kafka
- Storm
- Sentry
- Oozie
- Mahout
- Hadoop
- Spark
- Avro
- Cassandra
- Ambari
- Apache Thrift
- Apache Drill
- Apache Giraph
- Apache Drill Development Tools
- Chukwa
Categorizing Hadoop Tech Stack

- **Data Integration**
  - SQOOP, Flume, Chukwa, Kafka

- **Data Serialization**
  - Avro, Thrift

- **Data Storage (NOSQL)**
  - HBase, Cassandra

- **Data Access/Analytics**
  - Pig, Hive

- **Data Access/Analytics +**
  - Giraph, Storm, Drill, Tez,, Spark.

- **Management**
  - Ambari

- **Orchestration**
  - Zookeeper, Oozie

- **Data Intelligence**
  - Mahout

- **Security**
  - Knox, Sentry

- **Hadoop Dev Tools**
  - HDT
Hadoop Distributions

- Offered first commercial distribution
  - Cloudera: Hadoop Redhat: Linux
- 100% open source Hadoop with a twist
  - Proprietary admin/management console
- Cloudera Hadoop Distribution is called CDH
  - CDH = Cloudera Distribution for Hadoop

- Offered second commercial distribution
- 100% open source Hadoop with a twist
  - Proprietary C++ based filesystem
  - Proprietary admin/management console
- MapR Hadoop Distribution is called Mseries
  - M3, M5, M7

- Third commercial distribution
  - Founded for ex-Yahoo Hadoop experts
  - Spin-off Yahoo
- 100% open source Hadoop without any twist
  - 100% open source when it comes to Hadoop software
  - 100% open source admin/management tool called Ambari
- Hortonworks Hadoop Distribution is called HDP
  - HDP = Hortonworks Data Platform
Cloud Hadoop
Summary

• Data being generated at a tremendous rate

• Emerging field of Big data analytics and data science

• Businesses using both traditional data analytics and data science

• Traditional data processing not suitable for “Big Data” Processing

• Hadoop has become founding technology for Big data processing, Analytics, and Data Science!
Hadoop – A Perfect Platform for Big Data and Data Science
Steps to Write a MapReduce Program
MapReduce Programming Model

Shuffle and Sort: aggregate values by keys

- **Mapper**
  - a 1 b 2
  - c 3 c 6
  - a 5 c 2
  - b 7 c 8

- **Reducer**
  - X 5
  - Y 7
  - Z 9
The Problem

• Given a directory called /in in hdfs that contains a bunch of great books as text files, List all unique words used in all books and their respective counts
Solution Design
Solution Design – Job Input

• **Input location**
  – /in folder contains a list of books in text format

• **Input format**
  – Lines of text
  – Since text file, TextInputFormat class
  – Key is LongWritable
  – Value is Text
Solution Design – Mapper

• **Map Input**
  – Key is byte offset within the file
    Type is LongWritable
  – Value is line of text
    • Type is Text

• **Map Process**
  – Ignore the key
  – Parse the value (line of text)
    • For each word, print the word and a count of 1 (one)

• **Map Output**
  – Key is word
    • Type is Text
  – Value is count of 1 (one)
    • Type is IntWritable
Solution Design – Reducer

• **Reduce Input**
  – Key is word
    • Type is Text
  – Value is list of 1s (ones)
    • Type is Iterable of IntWritable

• **Reduce Process**
  – Add up the 1s (ones) to a variable called count

• **Reduce Output**
  – Key is word
    • Type is Text
  – Value is count
    • Type is IntWritable
Solution Design – Job Output

• Output location
  – /out will contain the output from reducer

• Output Format
  – Text file
  – Lines of text makes a record
  – Key is word
  – Value is count
  – Key value separated by a tab
Steps to Write a MapReduce Program

1. Implement the Mapper
2. Implement the Reducer
3. Configure the Job
4. Compile the classes
5. Package the classes
6. Run the Job
1. Implement the Mapper

- Create a class that extends Mapper class with 4 parameters
  1. Map input key
  2. Map input value
  3. Map output key (Should be same as Reducer input key)
  4. Map output value (Should be same as Reducer input value)
    - Map Output key has to be WritableComparable
    - Rest of the parameters should be Writable at a minimum
1. Implement the Mapper

• Override and Implement the map() method
  – Retrieve the passed input key and value
  – Write the logic necessary to do the processing
  – Use the passed Context to write the corresponding Mapper output key and value
2. Write the Mapper Class

```java
public class WordCountMapper extends Mapper<LongWritable, Text, Text, IntWritable> {

    IntWritable one = new IntWritable(1);
    Text word = new Text();

    @Override
    protected void map(LongWritable key, Text value, Context context)
    throws IOException, InterruptedException {
        String line = value.toString();
        StringTokenizer tokenizer = new StringTokenizer(line);
        while (tokenizer.hasMoreTokens()) {
            word.set(tokenizer.nextToken());
            context.write(word, one);
        }
    }
}
```
2. Implement the Reducer

- Write a class that extends Reducer class with 4 parameters
  1. Reduce input key (Should be same as Map input key)
  2. Reduce input value (Should be same as Map input value)
  3. Reduce output key
  4. Reduce output value

  - Input key classes should be WritableComparable
2. Implement the Reducer

• Override and Implement the reduce() method
  – Retrieve the passed input key and list of values
  – Write the logic necessary to do the processing
  – Use the passed Context to write the corresponding Reducer output key and value
public class WordCountReducer extends Reducer<Text, IntWritable, Text, IntWritable> {

    int i = 0;
    IntWritable count = new IntWritable();

    @Override
    protected void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {

        i = 0;
        for (IntWritable val : values) {
            i = i + 1;
        }
        count.set(i);
        context.write(key, count);
    }
}
3. Configure the Job

- Configure Job in driver class and submit

- Instantiate a Job object
  - Several factory style get methods to get a Job instance
    - Job.getInstance()
      - Used the default configuration object
    - Job.getInstance (conf)
    - Job.getInstance(conf, “jobname”)
      - Jobname is useful to track in the admin console
    - Possible to set the job name explicitly
      - job.setName(jobName)
3. Configure the Job - Input

- **Specify the Input path**
  - Could be file, directory or file pattern
    - Directory or file patterns are converted to a list of files as input
  - In this case getting the path from command line args
    - `TextInputFormat.addInputPath(job, new Path(args[0]));`
  - Can call `addInputPath()` several times for file, dir, or pattern

- **Specify the Input data format**
  - Input is specified in terms of `InputFormat`
    - Responsible for creating splits and a record reader
  - In this case `TextInputFormat`
    - Controls input types of key-value pairs, in this case `LongWritable` and `Text`
    - File is broken into lines, mapper will receive 1 line at a time
  - `job.setInputFormatClass(TextInputFormat.class);`
3. Configure the Job - Process

• Set the Mapper and Reducer classes
  – job.setMapperClass(class);
  – job.setReducerClass(class);

• Specify which jar for the Job
  – job.setJarByClass(class);
3. Configure the Job - Output

- **Specify the Output path**
  - Should be a directory
  - Output directory should not already exist
  - `FileOutputFormat.setOutputPath(path)`

- **Specify the Output data format**
  - Output is specified in terms of `OutputFormat`
  - For text files, it is `TextOutputFormat`
  - `job.setOutputFormatClass(TextOutputFormat.class)`

- **Specify the Output key-value classes**
  - `job.setOutputKeyClass(keyClass);`
  - `job.setOutputValueClass(valueClass);`
public class WordCount {

    public static void main(String args[]) {
        Job wordCountJob = null;
        wordCountJob = Job.getInstance
                       (new Configuration(), "WordCount");

        // Specify the Input path
        FileInputFormat.addInputPath(wordCountJob, new Path(args[0]));

        // Set the Input Data Format
        wordCountJob.setInputFormatClass(TextInputFormat.class);

        // Set the Mapper and Reducer Class
        wordCountJob.setMapperClass(WordCountMapper.class);
        wordCountJob.setReducerClass(WordCountReducer.class);

        // Set the Jar file
        wordCountJob.setJarByClass(WordCount.class);

        // Set the Output path
        FileOutputFormat.setOutputPath(wordCountJob,
                                       new Path(args[1]));
    }
}
3. Configure the Job - Output

// Set the Output Data Format
wordCountJob.setOutputFormatClass(TextOutputFormat.class);

// Set the Output Key and Value Class
wordCountJob.setOutputKeyClass(Text.class);
wordCountJob.setOutputValueClass(IntWritable.class);

// Submit the job
wordCountJob.waitForCompletion(true);
4. Compile the Classes

• Compile Mapper, Reducer and Main job classes

• Include Hadoop classes in CLASSPATH
  – All hadoop jar files
  – Dependent jars in the lib folder

• Include App dependent classes in CLASSPATH
  – If mappers and reducers require other dependent libraries, you need to include them in the CLASSPATH too
5. Package the Classes

• **Hadoop requires all jobs packaged as single jar**
  – Hadoop framework distributes jar file to nodes

• **Specify in code which jar file to distribute**
  – Specify jar of your job by calling `job.setJarByClass`  
    • `job.setJarByClass(getClass());`
    – Assuming the current class is part of the job of course
    – Hadoop will locate the jar file that contains the provided class

• **Dependent jars should be packaged within big jar**
  – Dependent jars are expected to be placed in `lib/` folder inside jar file
6. Run the Job

- **Two ways to run the program**
  1. **Traditional java command**
     - You have to set HADOOP CLASSPATH

     ```
     $ java –classpath mapreduce-basics.jar:... 
     bdpuh.mapreducebasics.WordCount /in /out
     ```

  2. **Use the more convenient yarn command**
     - Adds Hadoop’s libraries to CLASSPATH

     ```
     $ yarn jar mapreduce-basics.jar 
     bdpuh.mapreducebasics.WordCount /in /out
     ```
The Output Files

- **Output directory will have resultant files**
  - `_SUCCESS`
    - Indicating job was successful, otherwise file will not be present
  - Reducer output files with format “part-r-nnnnn”
    - nnnnn is an integer representing reducer number
    - Number is zero based
Steps to Write a MapReduce Program