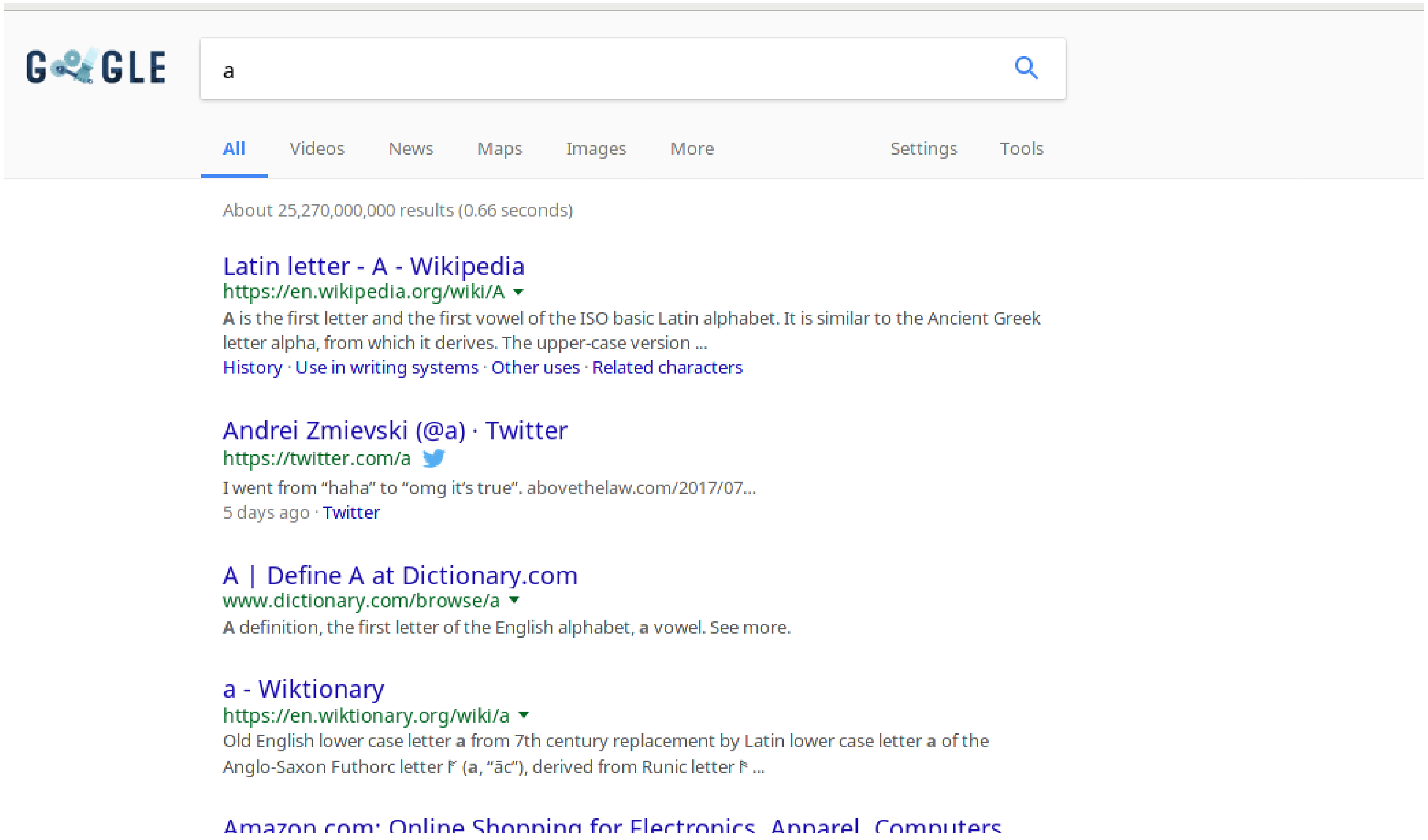
INTODUCTION TO HADOOP:
HDFS AND MAPREDUCE/YARN

SERHAT ÇEVİKEL

WHAT IS IN A GOOGLE SEARCH?



About 25,270,000,000 results (0.66 seconds)

Latin letter - A - Wikipedia
https://en.wikipedia.org/wiki/A
A is the first letter and the first vowel of the ISO basic Latin alphabet. It is similar to the Ancient Greek letter alpha, from which it derives. The upper-case version ...
History · Use in writing systems · Other uses · Related characters

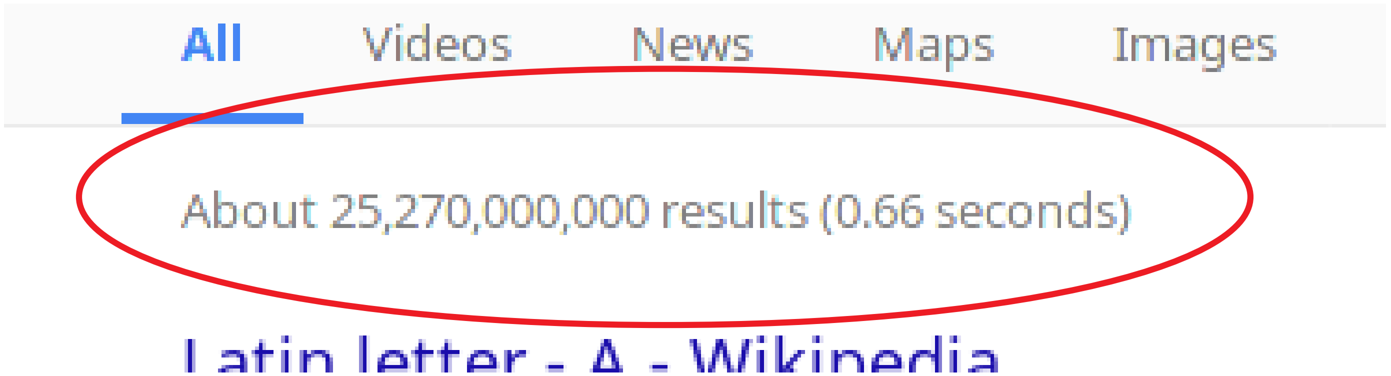
Andrei Zmievski (@a) · Twitter
https://twitter.com/a
I went from "haha" to "omg it's true". abovethelaw.com/2017/07...
5 days ago · Twitter

A | Define A at Dictionary.com
www.dictionary.com/browse/a
A definition, the first letter of the English alphabet, **a** vowel. See more.

a - Wiktionary
https://en.wiktionary.org/wiki/a
Old English lower case letter **a** from 7th century replacement by Latin lower case letter **a** of the Anglo-Saxon Futhorc letter ᚪ (**a**, "āc"), derived from Runic letter ᚨ ...

Amazon.com: Online Shopping for Electronics, Apparel, Computers

WHAT IS IN A GOOGLE SEARCH?

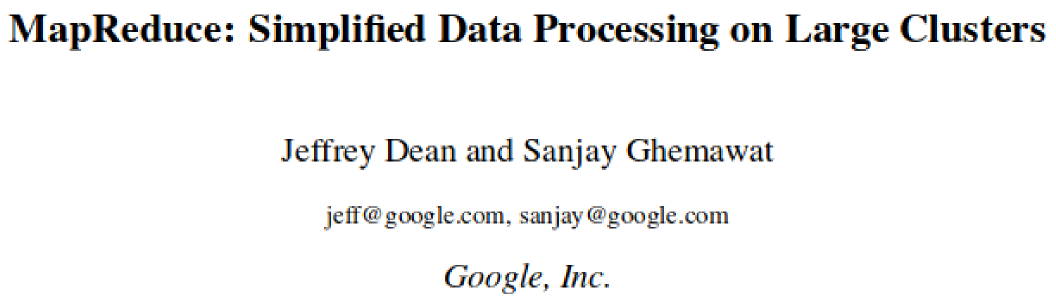


About 25,270,000,000 results (0.66 seconds)

Latin letter - A - Wikipedia

The computing power required to return that many results in the given time < 1s surpasses the capacity of any single processor

GOOGLE AND MAPREDUCE


MapReduce: Simplified Data Processing on Large Clusters

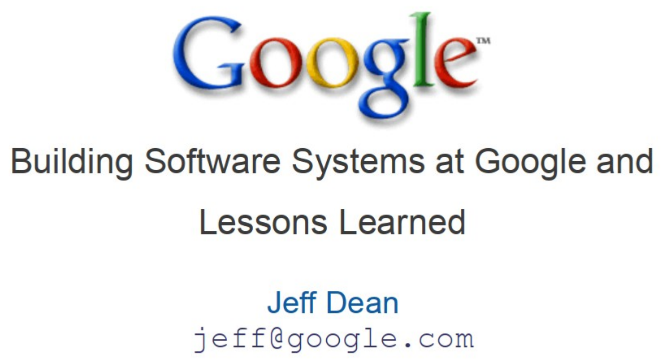
Jeffrey Dean and Sanjay Ghemawat

jeff@google.com, sanjay@google.com

Google, Inc.

“One of our most signicant uses of MapReduce to date has been a complete rewrite of the production indexing system that produces the data structures used for the Google web search service. The indexing system takes as input a large set of documents that have been retrieved by our crawling system ... The raw contents for these documents are more than 20 terabytes of data.” (2004)

GOOGLE AND MAPREDUCE



Google

Building Software Systems at Google and Lessons Learned

Jeff Dean
jeff@google.com

Hide messy details in MapReduce runtime library:

•automatic parallelization

•load balancing

•network and disk transfer optimizations

•handling of machine failures

•Robustness

(slide 68)

HADOOP MAPREDUCE PERFORMANCE

...Where MapReduce excels is in its ability to support elastic scalability, i.e. the allocation of more compute nodes from the cloud to speed up computation.

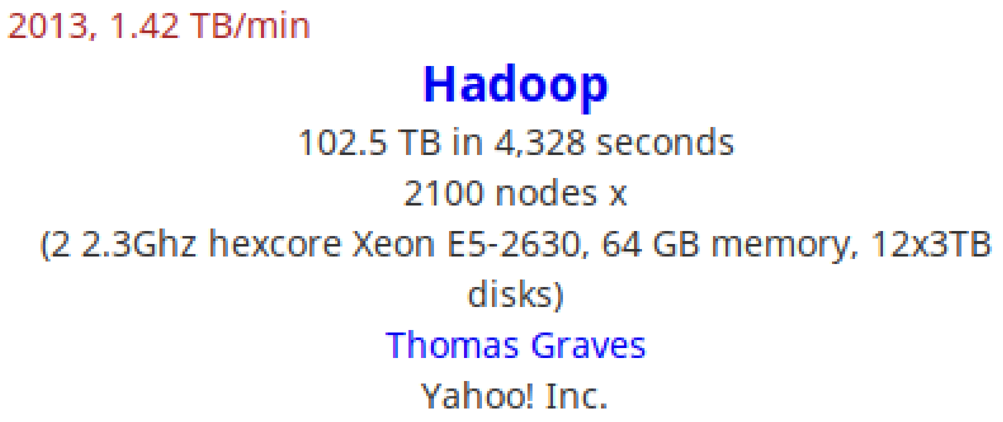
Approximately 1800 machines participated in this cluster, each with 2GHx Intel Xeon processors with Hyper-Threading enable, 4GB of memory and 320 GB of disk.

They performed two functions, a distributed grep command searching for a rare string across 1 terabyte of data, and a sort command, which sorts 1 terabyte of data.

The grep command completed in 150 seconds, while the sort completed in ~1000 seconds. RDBMS technologies would require structured data and massive parallel storage in order to match these results.

(https://mindmajix.com/mapreduce/history-and-advantages-of-hadoop-mapreduce-programming)

HADOOP MAPREDUCE PERFORMANCE



2013, 1.42 TB/min

Hadoop
102.5 TB in 4,328 seconds
2100 nodes x
(2 2.3Ghz hexcore Xeon E5-2630, 64 GB memory, 12x3TB disks)
Thomas Graves
Yahoo! Inc.

http://sortbenchmark.org/

MAPREDUCE ADVANTAGES

Simple Coding Model: With MapReduce, the programmer does not have to implement parallelism, distributed data passing, or any of the complexities that they would otherwise be faced with

Scalable: It has been reported that Hadoop can scale across thousands of nodes (Anand, 2008). The ability to scale horizontally to such a large degree can be attributed to the combination of distributed file systems and the philosophy of running the worker processes near the data, rather than attempting to move the data to the processes.

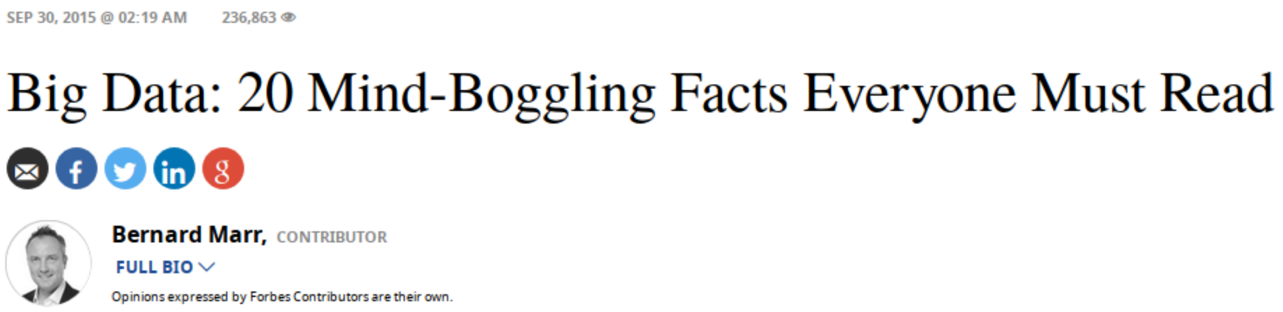
Supports Unstructured Data: Unstructured data is data that does not follow a specified format for big data. If 20 percent of the data available to enterprises is structured data, the other 80 percent is unstructured.

Because MapReduce processes simple key value pairs, it can support any type of data structure that fits into this model. This includes images, meta-data, large files, etc. The ability for the programmer to deal with irregular data is much easier in MapReduce than DBMS.

Fault Tolerance: Because of its highly distributed nature, MapReduce is very fault tolerant. Typically the distributed file systems that MapReduce support, along with the master process, enable MapReduce jobs to survive hardware failures (Dean & Ghemawat, 2004).

(https://mindmajix.com/mapreduce/history-and-advantages-of-hadoop-mapreduce-programming)

SOME FACTS ON GOOGLE SEARCH AND HADOOP


SEP 30, 2015 @ 02:19 AM 236,863

Big Data: 20 Mind-Boggling Facts Everyone Must Read

Bernard Marr, CONTRIBUTOR

FULL BIO

Opinions expressed by Forbes Contributors are their own.

Every second we create new data. For example, we perform 40,000 search queries every second (on Google alone), which makes it 3.5 searches per day and 1.2 trillion searches per year.

Distributed computing (performing computing tasks using a network of computers in the cloud) is very real. Google uses it every day to involve about 1,000 computers in answering a single search query, which takes no more than 0.2 seconds to complete.

The Hadoop (open source software for distributed computing) market is forecast to grow at a compound annual growth rate 58% surpassing $1 billion by 2020.

MAPREDUCE: FROM GOOGLE TO HADOOP

A popular open-source implementation that has support for distributed shuffles is part of Apache Hadoop

The name MapReduce originally referred to the proprietary Google technology, but has since been genericized

By 2014, Google was no longer using MapReduce as their primary Big Data processing model

(https://en.wikipedia.org/wiki/MapReduce)

MAPREDUCE: FROM GOOGLE TO HADOOP

Apache, the open source organization, began using MapReduce in the "Nutch" project, which is an open source web search engine that still is active today.

Hadoop began as a subproject in the Apache Lucern project, which provides text search capabilities across large databases.

In 2006, Doug Cutting, an employee of Yahoo!, designed Hadoop, naming it after his son's toy elephant.

While it was originally a subproject, Cutting released Hadoop as an open source Apache project in 2007. (Hadoop, 2011).

In 2008, Hadoop became a top level project at Apache. On July 2008, an experimental 4000 node cluster was created using Hadoop, and in 2009 during a performance test, Hadoop was able to sort a terabyte of data in 17 hours.

(https://mindmajix.com/mapreduce/history-and-advantages-of-hadoop-mapreduce-programming)

SO WHAT IS HADOOP?

What Is Apache Hadoop?

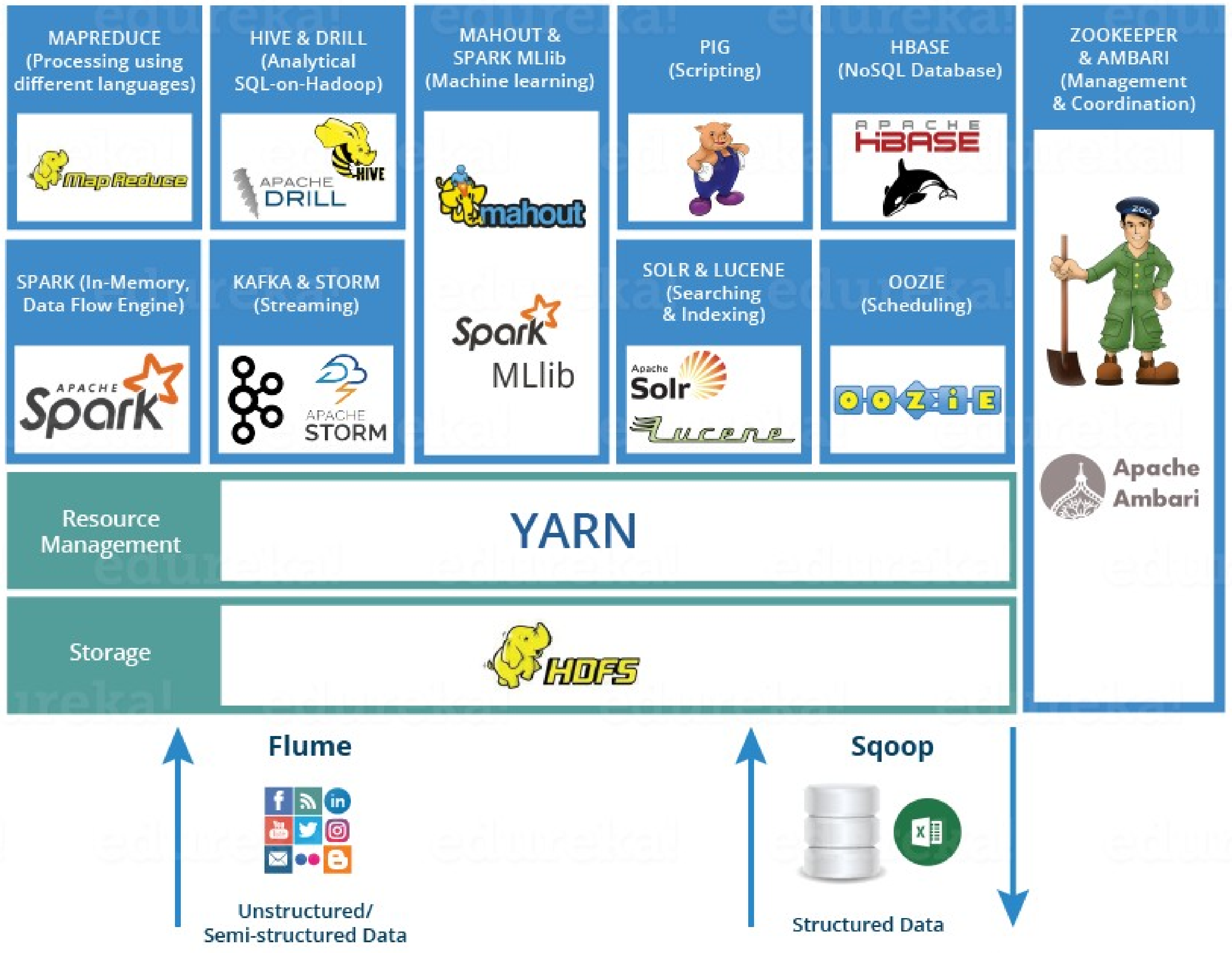
The Apache™ Hadoop® project develops open-source software for reliable, scalable, distributed computing.

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

The project includes these modules:

- Hadoop Common: The common utilities that support the other Hadoop modules.
- Hadoop Distributed File System (HDFS™): A distributed file system that provides high-throughput access to application data.
- Hadoop YARN: A framework for job scheduling and cluster resource management.
- Hadoop MapReduce: A YARN-based system for parallel processing of large data sets.

HADOOP “EXTENDED UNIVERSE”: AN ECOSYSTEM …


MAPREDUCE (Processing using different languages)
MapReduce
HIVE & DRILL (Analytical SQL-on-Hadoop)
APACHE DRILL
HIVE
MAHOUT & SPARK MLlib (Machine learning)
mahout
Spark MLlib
PIG (Scripting)
HBASE (NoSQL Database)
APACHE HBASE
ZOOKEEPER & AMBARI (Management & Coordination)
Apache Ambari
SPARK (In-Memory, Data Flow Engine)
APACHE Spark
KAFKA & STORM (Streaming)
APACHE STORM
SOLR & LUCENE (Searching & Indexing)
Apache Solr
Lucene
OOZIE (Scheduling)
OOZIE
Resource Management
YARN
Storage
HDFS
Flume
Unstructured/ Semi-structured Data
Sqoop
Structured Data

WHAT IS DISTRIBUTED COMPUTING?

CS61A Structure and Interpretation of Computer Programs

Instructor: John DeNero
MWF 2:00-3:00 PM 1 Pimentel

A distributed system is a network of autonomous computers that communicate with each other in order to achieve a goal.

The computers in a distributed system are independent and do not physically share memory or processors.

They communicate with each other using messages, pieces of information transferred from one computer to another over a network

FACTORS TOWARDS DISTRIBUTED COMPUTING

Accelerating data sizes:

The data volumes are exploding, more data has been created in the past two years than in the entire previous history of the human race.

Data is growing faster than ever before and by the year 2020, about 1.7 megabytes of new information will be created every second for every human being on the planet.

By then, our accumulated digital universe of data will grow from 4.4 zettabyets today to around 44 zettabytes, or 44 trillion gigabytes.

(https://www.forbes.com/sites/bernardmarr/2015/09/30/big-data-20-mind-boggling-facts-everyone-must-read/#1e7a4c1717b1)

FACTORS TOWARDS DISTRIBUTED COMPUTING

Stagnating clock speeds:

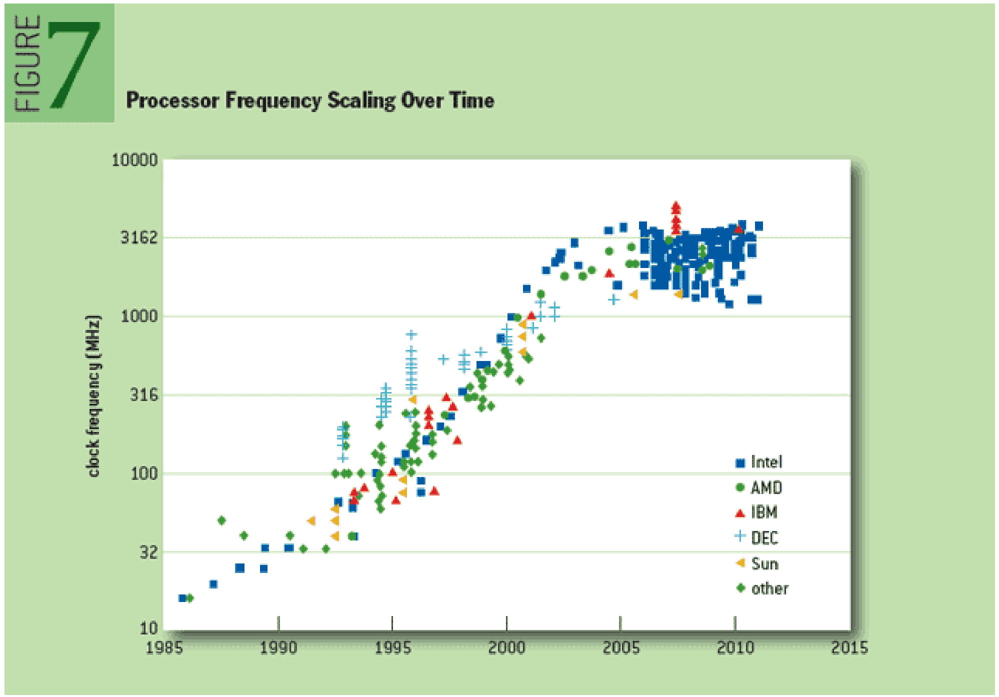
… while the benefits of making things smaller have been decreasing, the costs have been rising… the components are approaching a fundamental limit of smallness: the atom… the fewer atoms you have, the harder it becomes to store and manipulate electronic 1s and 0s. Smaller transistors now need trickier designs and extra materials… (http://www.economist.com/technology-quarterly/2016-03-12/after-moores-law)

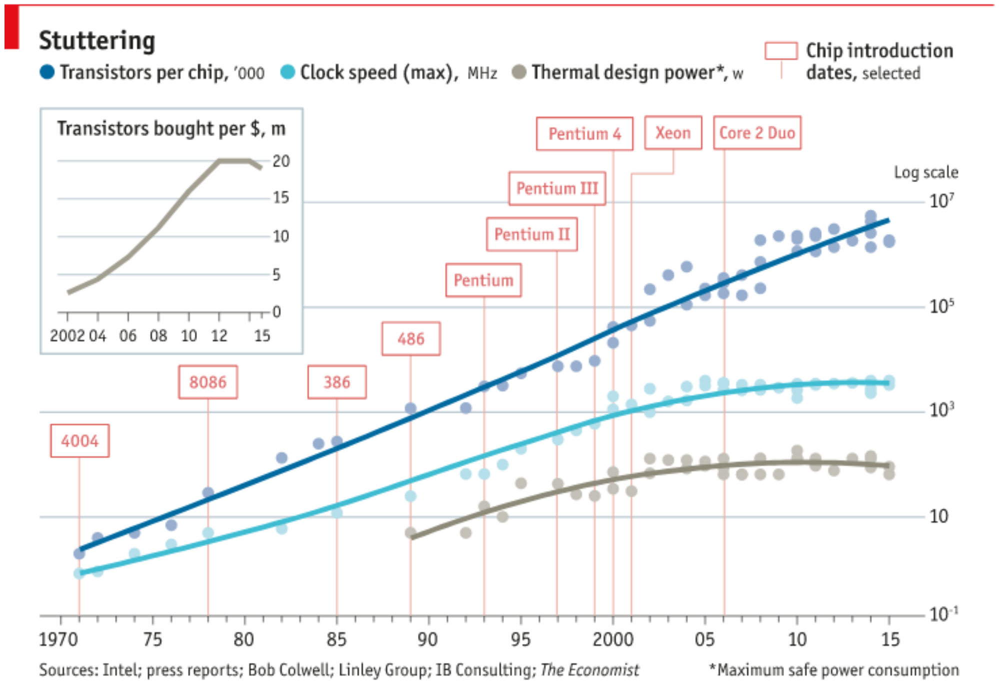
The scaling of voltage and current with length is reaching its limits, since transistor gates have become too thin, affecting their structural integrity, and currents are starting to leak (https://www.comsol.com/blogs/havent-cpu-clock-speeds-increased-last-years/)

FACTORS TOWARDS DISTRIBUTED COMPUTING

Stagnating clock speeds:

FIGURE 7 Processor Frequency Scaling Over Time





FACTORS TOWARDS DISTRIBUTED COMPUTING

Faster networking:

10 Mbps (1978), 100Mbps(early 1990's), 1Gbps (1996), 10Gbps (2000), 100Gbps (2010)

400Gbps (2017e as of 2013), 1 Tbps (2020e as of 2013), 1Pbps (2050e as of 2013)

(https://www.computerworld.com.au/slideshow/462393/pictures-20-milestones-ethernet-first-40-years/)

FACTORS TOWARDS DISTRIBUTED COMPUTING

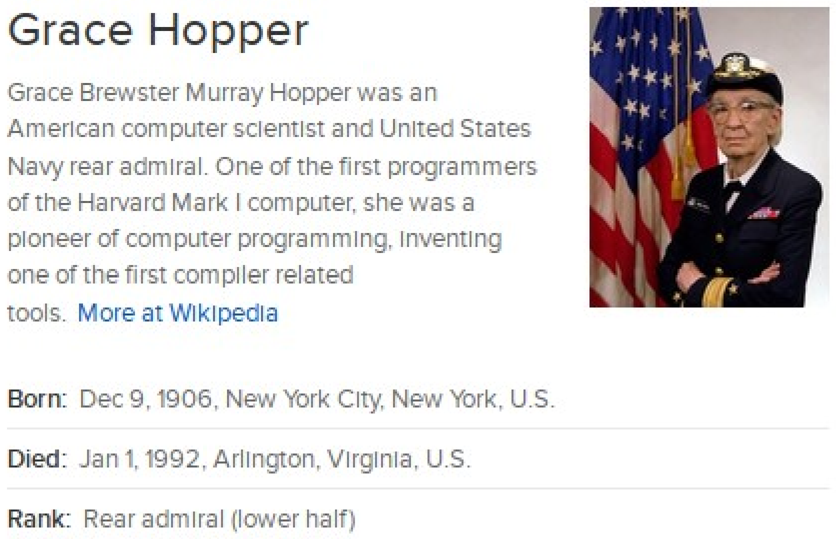
Easier deployment of clusters:

Virtualization technologies: VMs (Virtualbox, VMware, KVM, QEMU), containers (Docker)

Provisioning/configuration/deployment tools: Vagrant, Chef, Puppet, Ansible

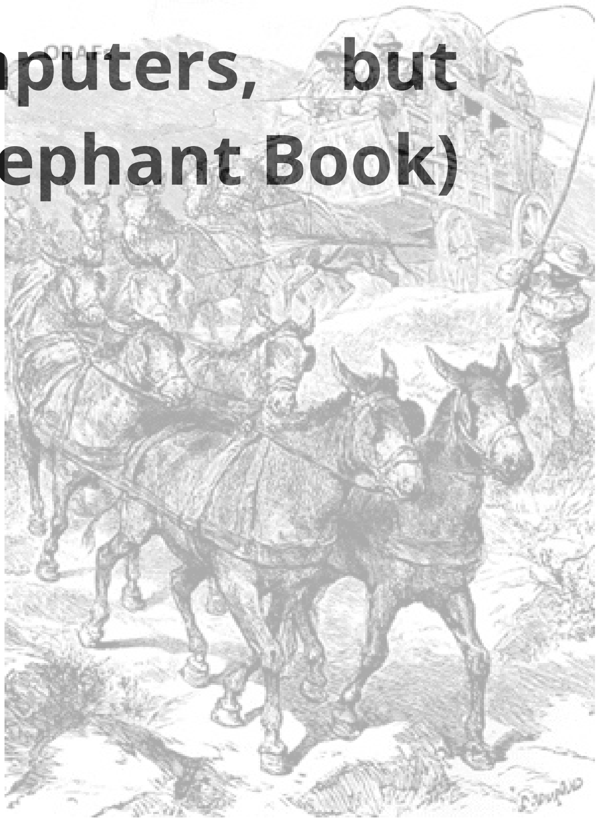
DISTRIBUTED COMPUTING

"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" (Elephant Book)


Grace Hopper

Grace Brewster Murray Hopper was an American computer scientist and United States Navy rear admiral. One of the first programmers of the Harvard Mark I computer, she was a pioneer of computer programming, inventing one of the first compiler related tools. More at Wikipedia

Born:	Dec 9, 1906, New York City, New York, U.S.
Died:	Jan 1, 1992, Arlington, Virginia, U.S.
Rank:	Rear admiral (lower half)



HADOOP DISTRIBUTED FILE SYSTEM (HDFS)

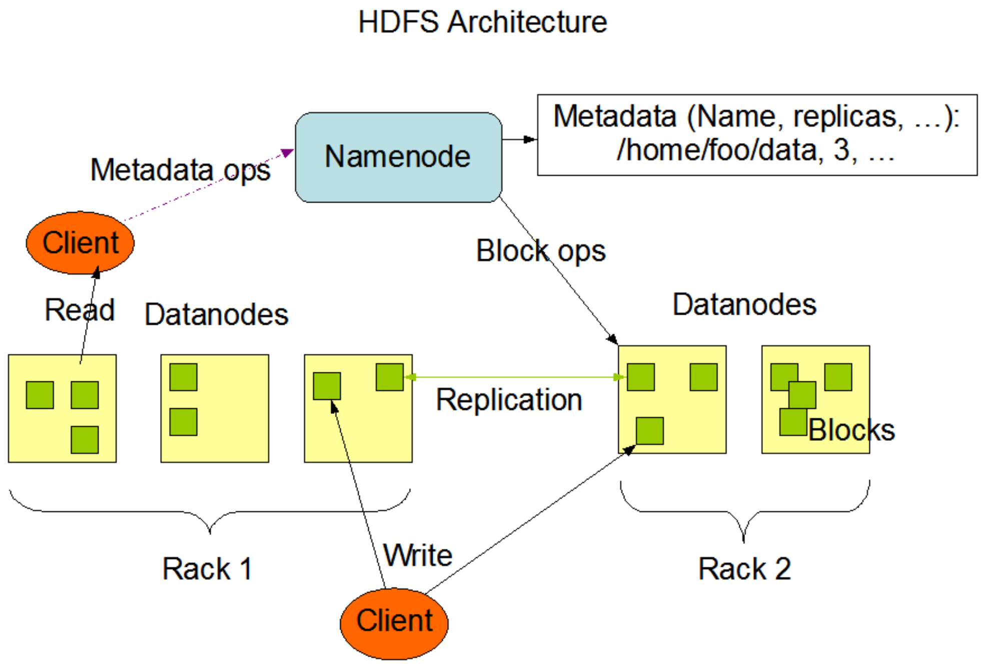
HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware

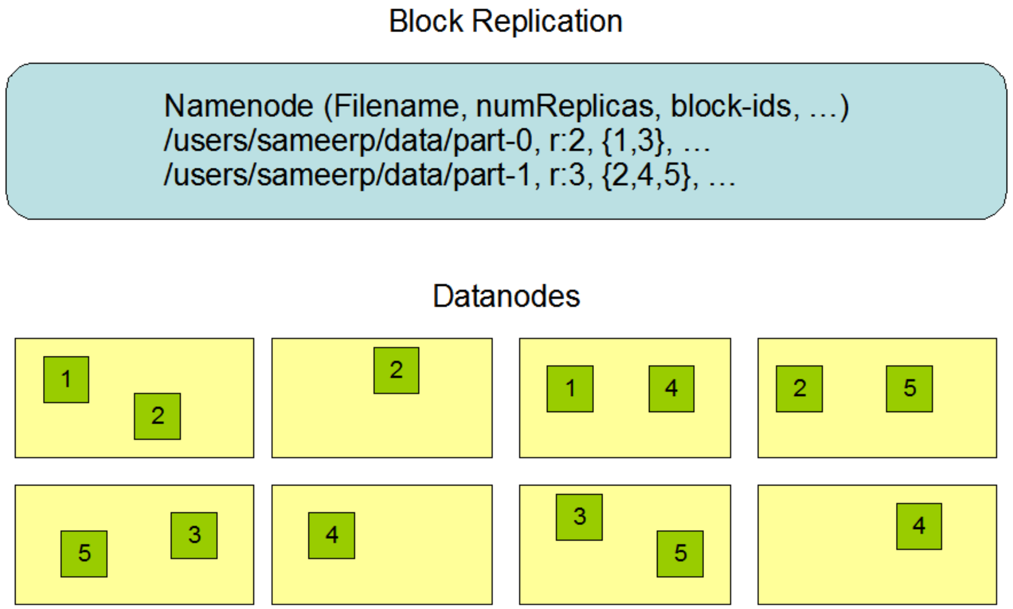
HDFS has a master/slave architecture. An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on.

Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes. The NameNode determines the mapping of blocks to DataNodes. The DataNodes are responsible for serving read and write requests from the file system's clients.

(http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html)

HADOOP DISTRIBUTED FILE SYSTEM (HDFS)



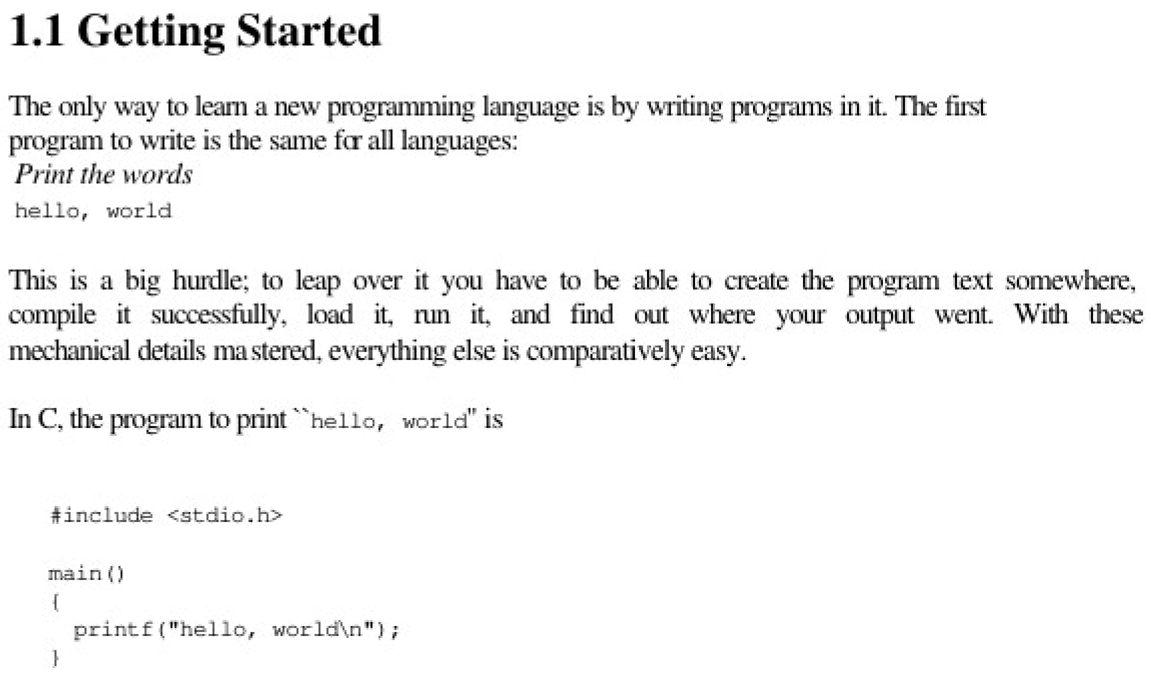

Block Replication
Namenode (Filename, numReplicas, block-ids, …)
/users/sameerp/data/part-0, r:2, {1,3}, …
/users/sameerp/data/part-1, r:3, {2,4,5}, …
Datanodes
1
2
2
1
4
2
5
5
3
4
3
5
4

HDFS is designed to reliably store very large files across machines in a large cluster. It stores each file as a sequence of blocks. The blocks of a file are replicated for fault tolerance. The block size and replication factor are configurable per file.

(http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html)

ANATOMY OF A MAPREDUCE JOB: WORD COUNT

The tradition in starting to learn a new programming language is to write a program that prints the words "Hello, World" on the screen. From the K&R book on C:



1.1 Getting Started

The only way to learn a new programming language is by writing programs in it. The first program to write is the same for all languages:
Print the words
`hello, world`

This is a big hurdle; to leap over it you have to be able to create the program text somewhere, compile it successfully, load it, run it, and find out where your output went. With these mechanical details ma stered, everything else is comparatively easy.

In C, the program to print ``hello, world'' is

```
#include <stdio.h>

main()
{
  printf("hello, world\n");
}
```

ANATOMY OF A MAPREDUCE JOB: WORD COUNT

The “Hello World” job for Hadoop MapReduce is the word count example

The easiest way to deploy a MR job is “hadoop streaming”: Hadoop streaming is a utility that comes with the Hadoop distribution. The utility allows you to create and run Map/Reduce jobs with any executable or script as the mapper and/or the reducer. (Elephant Book)

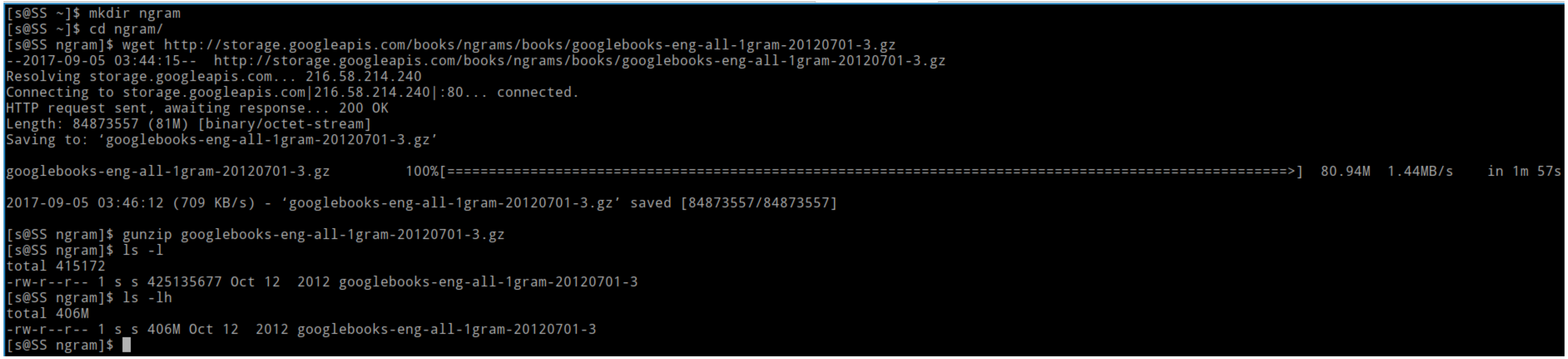
So we will construct our word count job using hadoop streaming

ANATOMY OF A MAPREDUCE JOB: WORD COUNT

Word Count: Take a large text file/files and get the count of words

The text source: “3” file from Ngram corpus: Number of occurrences of each word including “3” for each year within books in Google Books

http://storage.googleapis.com/books/ngrams/books/googlebooks-eng-all-1gram-20120701-3.gz



```
[s@SS ~]$ mkdir ngram
[s@SS ~]$ cd ngram/
[s@SS ngram]$ wget http://storage.googleapis.com/books/ngrams/books/googlebooks-eng-all-1gram-20120701-3.gz
--2017-09-05 03:44:15--  http://storage.googleapis.com/books/ngrams/books/googlebooks-eng-all-1gram-20120701-3.gz
Resolving storage.googleapis.com... 216.58.214.240
Connecting to storage.googleapis.com|216.58.214.240|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 84873557 (81M) [binary/octet-stream]
Saving to: ‘googlebooks-eng-all-1gram-20120701-3.gz’

googlebooks-eng-all-1gram-20120701-3.gz         100%[===================================================================================================>]  80.94M  1.44MB/s    in 1m 57s

2017-09-05 03:46:12 (709 KB/s) - ‘googlebooks-eng-all-1gram-20120701-3.gz’ saved [84873557/84873557]

[s@SS ngram]$ gunzip googlebooks-eng-all-1gram-20120701-3.gz
[s@SS ngram]$ ls -l
total 415172
-rw-r--r-- 1 s s 425135677 Oct 12  2012 googlebooks-eng-all-1gram-20120701-3
[s@SS ngram]$ ls -lh
total 406M
-rw-r--r-- 1 s s 406M Oct 12  2012 googlebooks-eng-all-1gram-20120701-3
[s@SS ngram]$ 
```

Compressed size: 80.9M, uncompressed size: 406M

ANATOMY OF A MAPREDUCE JOB: WORD COUNT

Traditional UNIX way to count words:

cat: UNIX tool to concatenate text files and feed to standart output

UNIX pipes (“|”): Feed the standard output of the former command into the latter command as standard input

wc: Word, line, and byte or character count

ANATOMY OF A MAPREDUCE JOB: WORD COUNT

```
[s@SS ngram]$ cat googlebooks-eng-all-1gram-20120701-3 | wc -w
92586944
```

Reads as: Concatenate the contents of files with the “cat command”, feed to output to word count by pipe “|” and count the number of words with the “w” option of wc command

ANATOMY OF A MAPREDUCE JOB: WORD COUNT

Hadoop MapReduce way to count words:

First of all, input files are put into HDFS from local:

```
hadoop fs -put /local/path/to/googlebooks-eng-all-1gram-20120701-3 myInputDirs
```

The input files are distributed by the NameNode across DataNodes with stated replication factors

And streaming job is run with following minimum parameters:

```
mapred streaming \
  -input myInputDirs \
  -output myOutputDir \
  -mapper /bin/cat \
  -reducer /usr/bin/wc
```

The YARN resource manager distributes the job to nodes, distribution defined by the mapper, and each nodes performs the task defined by the reducer. YARN then collects the results from all nodes

(http://hadoop.apache.org/docs/current/hadoop-streaming/HadoopStreaming.html)

ANATOMY OF A MAPREDUCE JOB: WORD COUNT

Both the mapper and the reducer are executables that read the input from stdin (line by line) and emit the output to stdout. The utility will create a Map/Reduce job, submit the job to an appropriate cluster, and monitor the progress of the job until it completes.

When an executable is specified for mappers, each mapper task will launch the executable as a separate process when the mapper is initialized. As the mapper task runs, it converts its inputs into lines and feed the lines to the stdin of the process. In the meantime, the mapper collects the line oriented outputs from the stdout of the process and converts each line into a key/value pair, which is collected as the output of the mapper

Each reducer task will launch the executable as a separate process then the reducer is initialized. As the reducer task runs, it converts its input key/values pairs into lines and feeds the lines to the stdin of the process. In the meantime, the reducer collects the line oriented outputs from the stdout of the process, converts each line into a key/value pair, which is collected as the output of the reducer.

RUNNING THE WORD COUNT JOB ON A BOX

Two levels of virtualization are utilized: VM & container

On Windows 10 host, an Ubuntu Linux VM is deployed by Vagrant, provisioned by an inline shell script (Vagrantfile)

The box is brought up by just “vagrant up” command

Provisioning shell script downloads and deploys Clusterdock, a docker image that includes Cloudera Distributed Hadoop (CDH) (https://hub.docker.com/r/cloudera/clusterdock/)

CDH is Cloudera’s 100% open source platform distribution, including Apache Hadoop and built specifically to meet enterprise demands. https://www.cloudera.com/products/open-source/apache-hadoop/key-cdh-components.html

So with a ONE CLICK command (“vagrant up”), a full development/learning environment including all major components of the Hadoop ecosystem including easy-to-use web-gui interfaces is deployed

RUNNING THE WORD COUNT JOB ON A BOX

The Hadoop Cluster to run the job will be in pseudo-distributed mode: Multiple nodes on the same machine. We will have two nodes on the local machines

We won't use the terminal to type in the command. We will utilize "HUE": Hadoop User Experience; the open source web GUI that lets you easily interact with the Hadoop ecosystem

Inside HUE web GUI, we use the file browser to interact with the HDFS (upload data) and Oozie Workflow Scheduler to construct the job with its parameters and track the job