CX4242:

Scaling Up Hadoop

Mahdi Roozbahani
Lecturer, Computational Science and Engineering, Georgia Tech

How to handle data that is **really** large?

Really big, as in...

- Petabytes (PB, about 1000 times of terabytes)
- Or beyond: exabyte, zettabyte, etc.

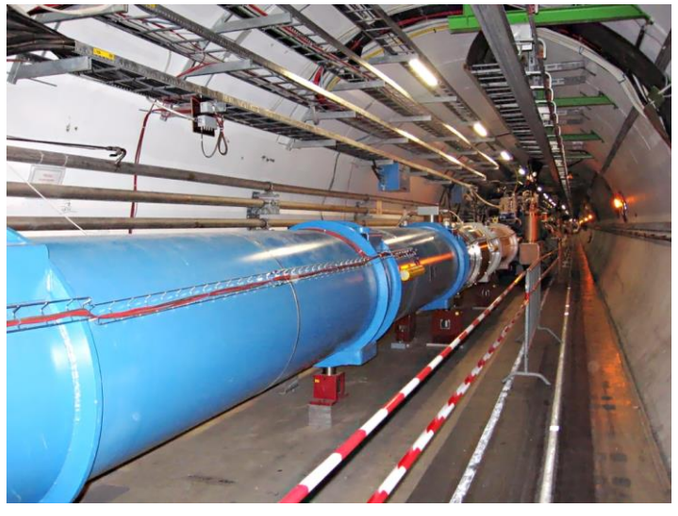
Do we *really* need to deal with such scale?

- Yes!

“Big Data” is Common...

Google processed **24 PB / day** (2009)

Facebook’s add **0.5 PB / day** to its data warehouses



CERN generated **200 PB** of data from “Higgs boson” experiments

Avatar’s 3D effects took **1 PB** to store

So, think **BIG**!

http://www.theregister.co.uk/2012/11/09/facebook_open_sources_corona/
http://thenextweb.com/2010/01/01/avatar-takes-1-petabyte-storage-space-equivalent-32-year-long-mp3/
http://dl.acm.org/citation.cfm?doid=1327452.1327492

How to analyze such large datasets?

First thing, how to **store** them?

Single machine? 60TB SSD ($$$) now available

Cluster of machines?

- How many machines?
- Need data backup, redundancy, recovery, etc.
- Need to worry about machine and drive failure.

Really? Really???

https://arstechnica.com/gadgets/2016/08/seagate-unveils-60tb-ssd-the-worlds-largest-hard-drive/

http://lifehacker.com/how-long-will-my-hard-drives-really-last-1700405627

How to analyze such large datasets?

3% of 100,000 hard drives fail within **first 3 months**

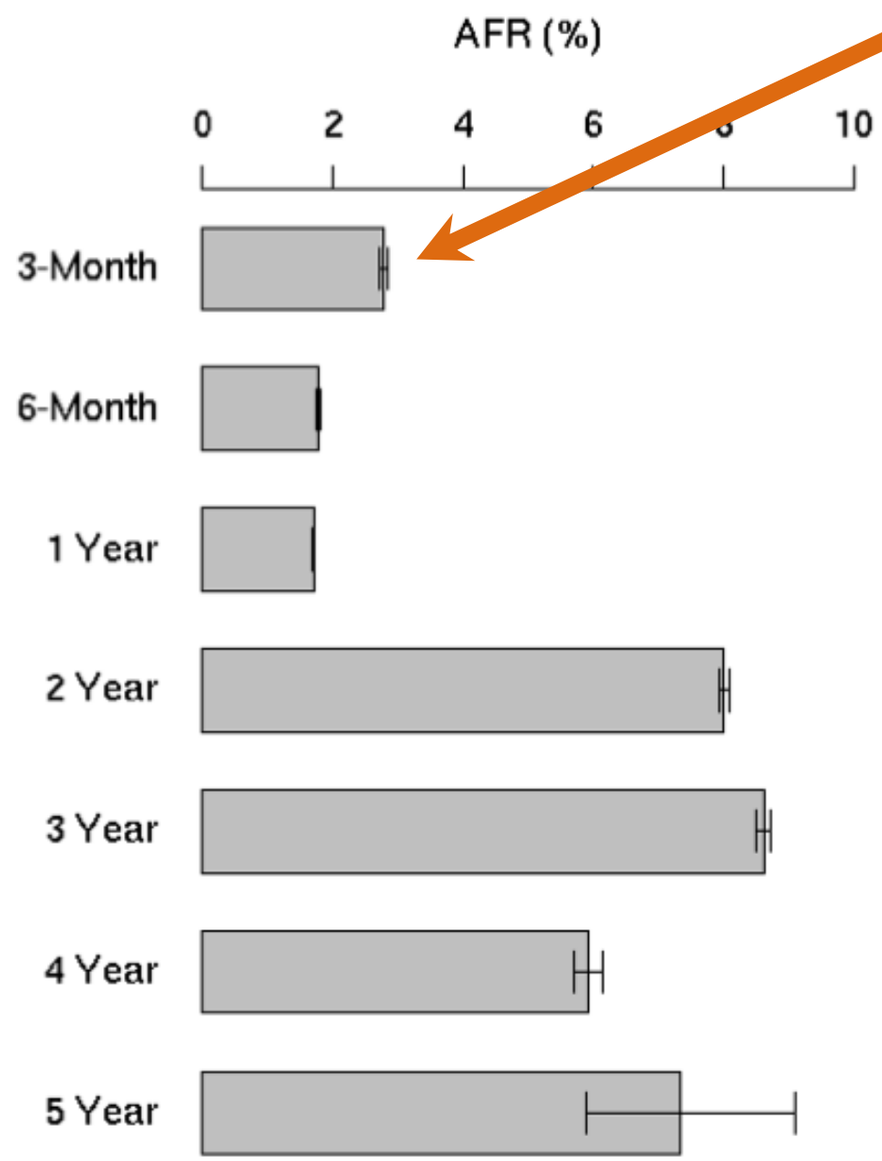


Figure 2: Annualized failure rates broken down by age groups

Failure Trends in a Large Disk Drive Population

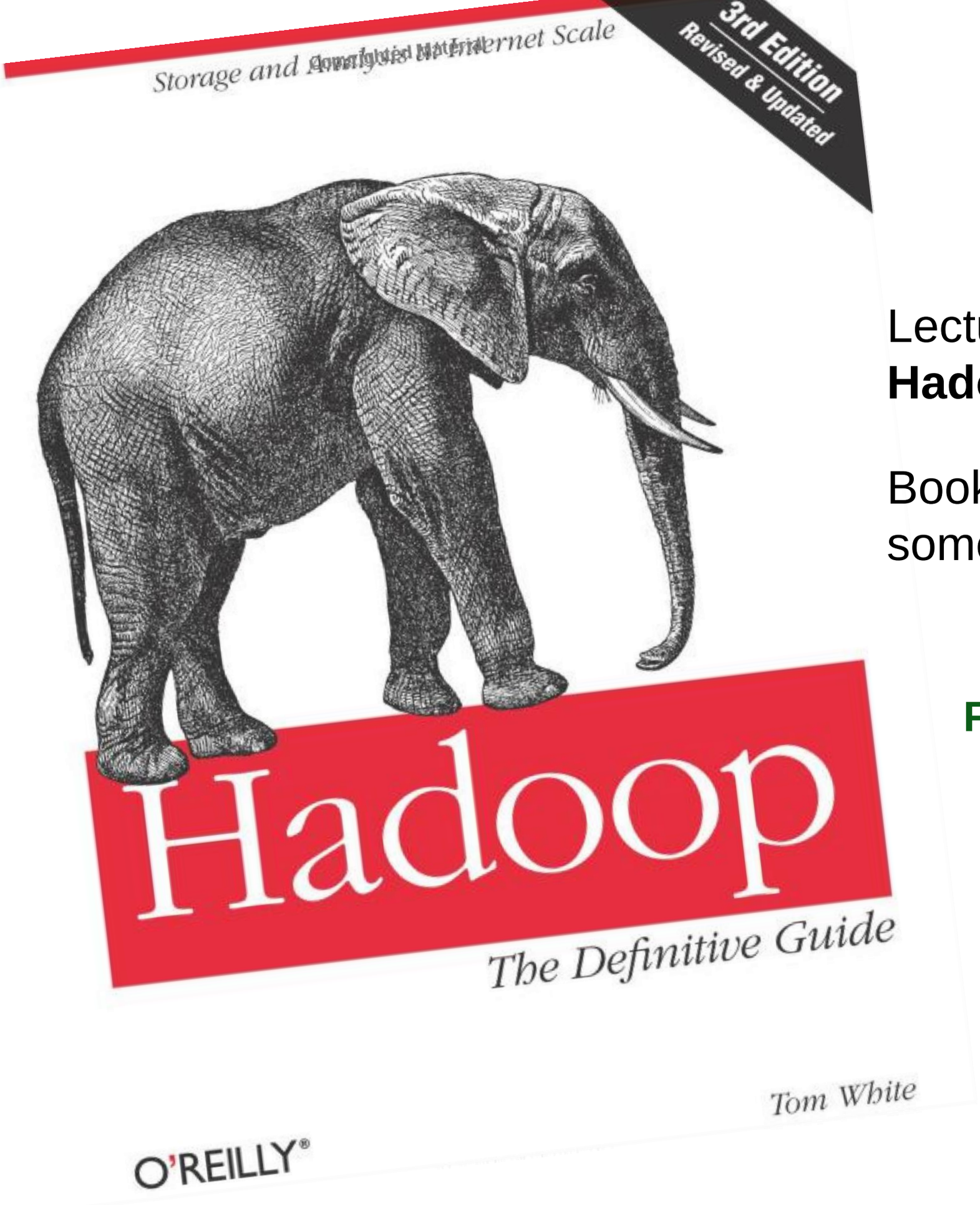
http://static.googleusercontent.com/external_content/untrusted_dlcp/research.google.com/en/us/archive/disk_failures.pdf

http://arstechnica.com/gadgets/2015/08/samsung-unveils-2-5-inch-16tb-ssd-the-worlds-largest-hard-drive/

How to analyze such large datasets?

How to analyze them?

- What software libraries to use?
- What programming languages to learn?
- Or more generally, what framework to use?



Lecture based on
Hadoop: The Definitive Guide

Book covers Hadoop, some Pig, some HBase, and other things.

FREE on Safari Books Online for Georgia Tech students!!

Open-source software for reliable, scalable, distributed computing

Written in Java

Scale to thousands of machines

- Linear scalability (with good algorithm design):
  if you have 2 machines, your job runs twice as fast (ideally)

Uses simple programming model (MapReduce)

Fault tolerant (HDFS)

- Can recover from machine/disk failure
  (no need to restart computation)

Why learn Hadoop?

Fortune 500 companies use it

Many research groups/projects use it

Strong community support, and favored/backed my major companies, e.g., IBM, Google, Yahoo, eBay, Microsoft, etc.

It's **free**, open-source

Low cost to set up (works on commodity machines)

An "essential skill", like SQL

http://strataconf.com/strata2012/public/schedule/detail/22497

Elephant in the room

Hadoop created by Doug Cutting and Michael Cafarella while at Yahoo

Hadoop named after Doug’s son’s toy elephant

How does Hadoop scale up computation?

Uses master-worker architecture, and a simple computation model called **MapReduce**
(popularized by Google's paper)

Simple way to think about it

1. Divide data and computation into smaller pieces; each machine works on one piece
2. Combine results to produce final results

MapReduce: Simplified Data Processing on Large Clusters
http://static.usenix.org/event/osdi04/tech/full_papers/dean/dean.pdf

How does Hadoop scale up computation?

More technically...

1. **Map phase**
   Master node **divides** data and computation into smaller pieces; each worker node ("mapper") works on one piece independently in parallel
2. **Shuffle phase** (automatically done for you)
   Master **sorts and moves** results to "reducers"
3. **Reduce phase**
   Worker nodes ("reducers") **combines** results independently in parallel

Example:

Find words' frequencies among text documents

Input

- "Apple Orange Mango Orange Grapes Plum"
- "Apple Plum Mango Apple Apple Plum"

Output

- Apple, 4
  Grapes, 1
  Mango, 2
  Orange, 2
  Plum, 3

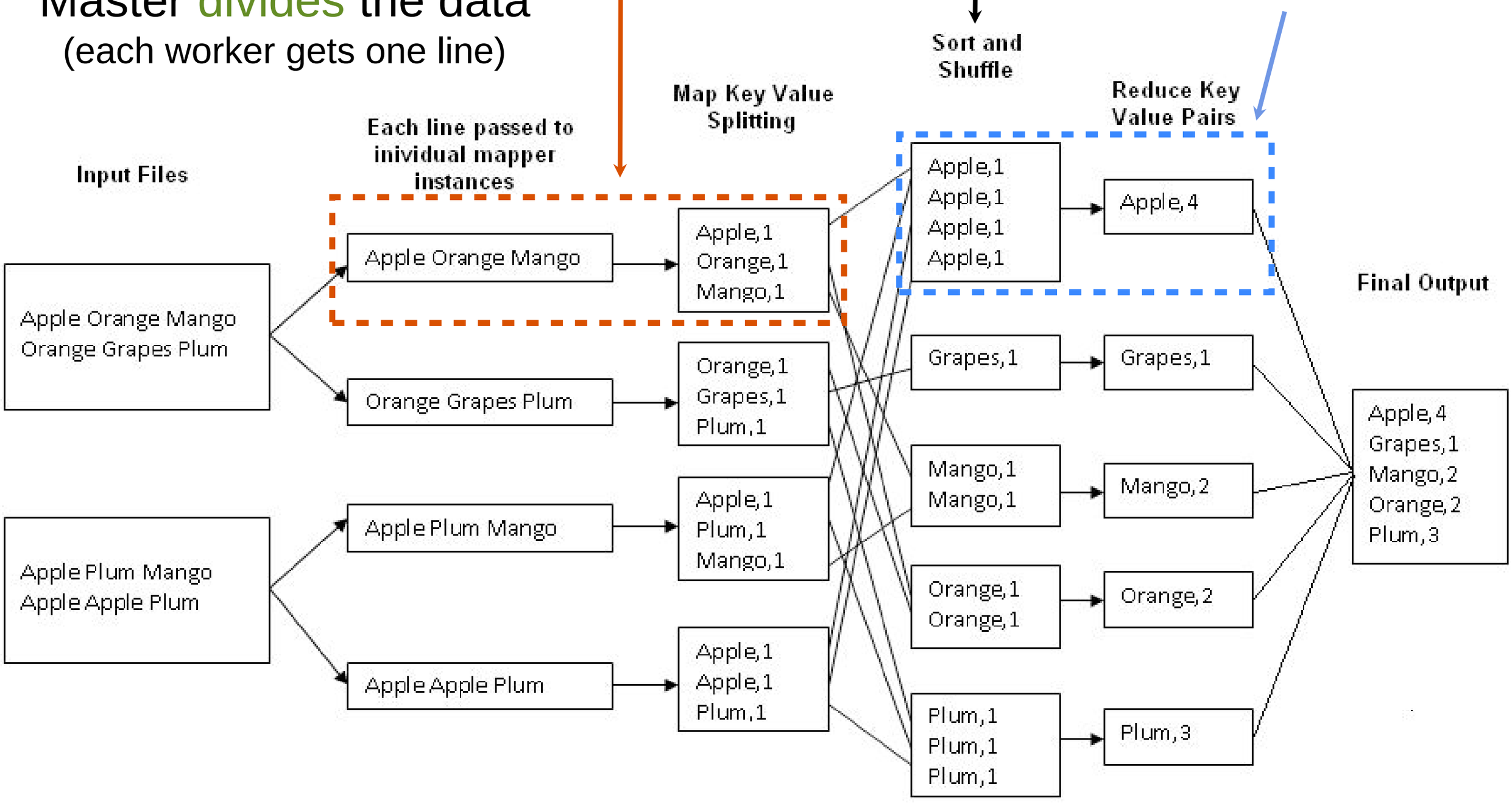
http://kickstarthadoop.blogspot.com/2011/04/word-count-hadoop-map-reduce-example.html

Each worker (mapper) outputs a **key-value pair**

Pairs sorted by key
(automatically done)

Each worker (reducer) combines pairs into one

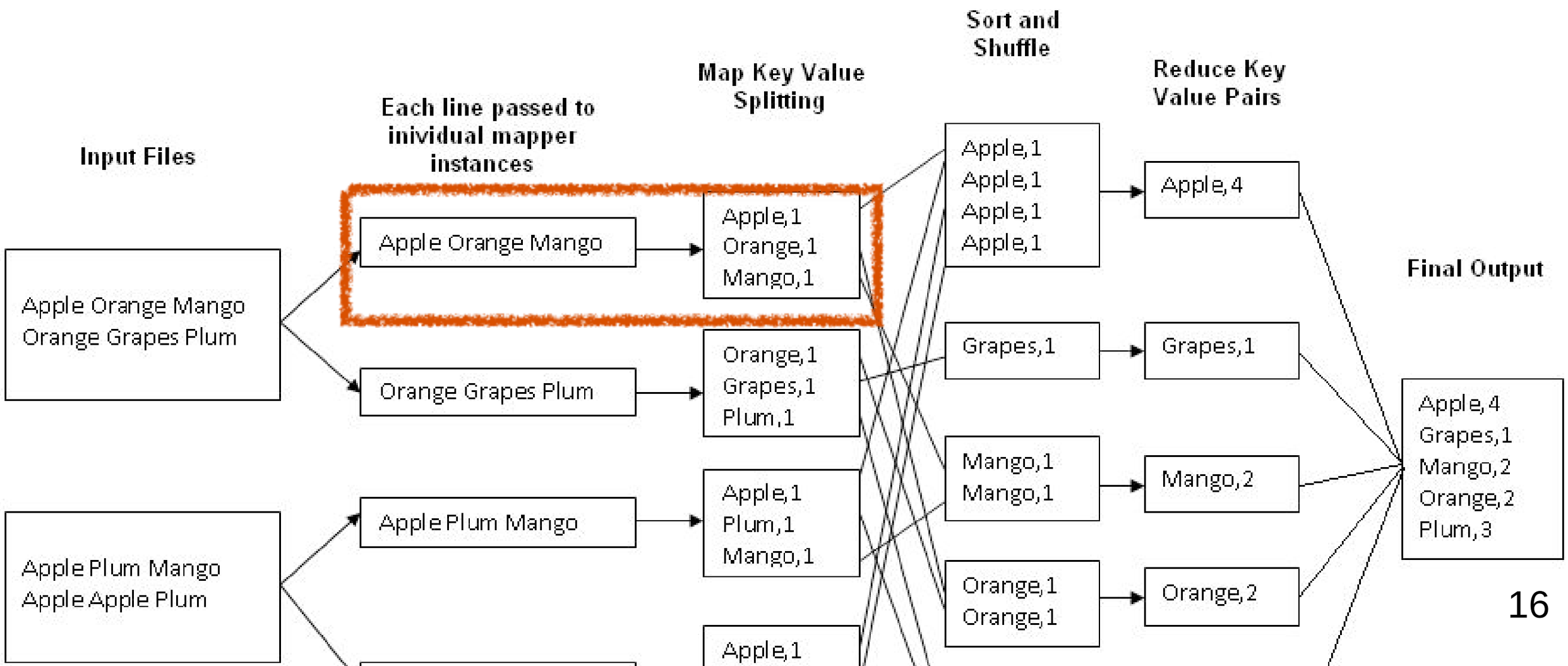
Master divides the data
(each worker gets one line)



A machine can be **both** a mapper and a reducer

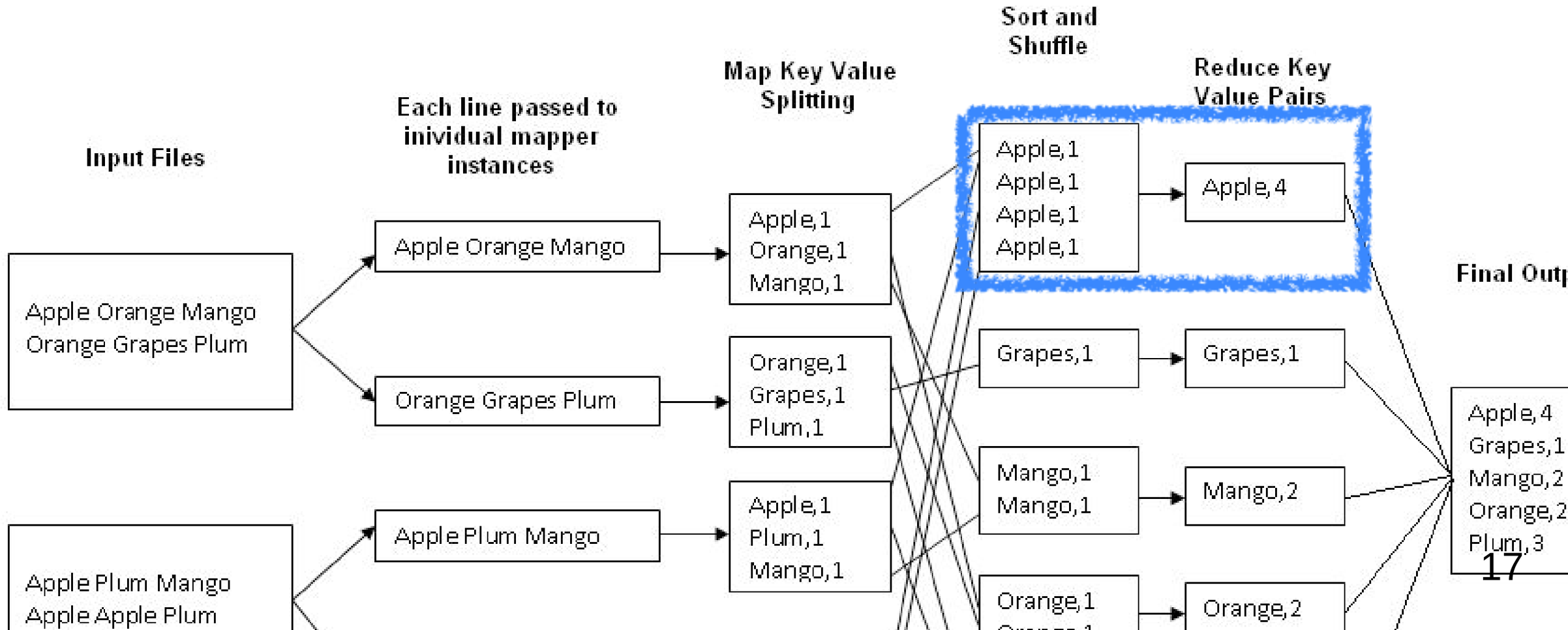
How to implement this?

```
map(String key, String value):
    // key: document id
    // value: document contents
    for each word w in value:
      emit(w, "1");
```



How to implement this?

```
reduce(String key, Iterator values):
    // key: a word
    // values: a list of counts
    int result = 0;
    for each v in values:
      result += ParseInt(v);
    Emit(AsString(result));
```



What if a machine dies?

Replace it!

"map" and "reduce" jobs **redistributed** (for you) to other machines

Hadoop's HDFS (Hadoop File System) enables this

HDFS: Hadoop File System

A distribute file system

Built on top of OS's existing file system to provide redundancy and distribution

HDFS hides complexity of distributed storage and redundancy from the programmer

In short, you don't need to worry much about this!

“History” of HDFS and Hadoop

Hadoop & **HDFS** based on...

- 2003 *Google File System* (GFS) paper
  http://cracking8hacking.com/cracking-hacking/Ebooks/Misc/pdf/The%20Google%20filesystem.pdf
- 2004 Google *MapReduce* paper
  http://static.googleusercontent.com/media/research.google.com/en/us/archive/mapreduce-osdi04.pdf

What can you use Hadoop for?

As a “swiss knife”.

Works for many types of analyses/tasks (but not all of them).

What if you want to write less code?

- There are tools to make it easier to write MapReduce program (**Pig**), or to query results (**Hive**)

How to try Hadoop?

Hadoop can run on a **single machine** (e.g., your laptop)

- Takes < 30min from setup to running

Or a **“home-grown” cluster**

- Research groups often connect retired computers as a small cluster

Amazon EC2 (Amazon Elastic Compute Cloud), **Microsoft Azure**

- You only pay for what you use, e.g, compute time, storage