Lesson 1 Notes

Introduction

Hi! Welcome to Fundamentals of Hadoop and MapReduce. My name's Sarah Sproehnle, and I'm the Vice President of Educational Services at Cloudera, a company which helps develop, support, and manage Hadoop.

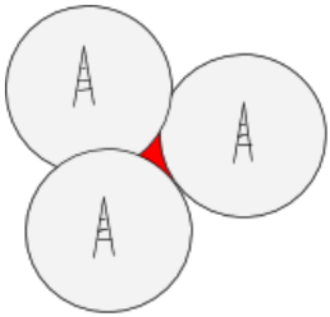
And I'm Ian Wrigley, Cloudera's Senior Curriculum Manager. Between us, Sarah and I have been responsible for bringing Hadoop training to over 20,000 people, and we're excited to reach a much bigger audience here at Udacity. During this course we're going to discuss what big data is, what Hadoop is, why it's useful, and how to write MapReduce code.

By the end of the course, you'll be able to describe the kinds of problems Hadoop addresses, and you'll have written MapReduce programs to efficiently analyze very large Web server log files. In fact, you'll have had hands-on experience running a Hadoop job by the end of lesson two.

So, let's start. In this lesson, we're going to define 'big data', the sort of problems it introduces, and how to address those problems.

Sources of Data

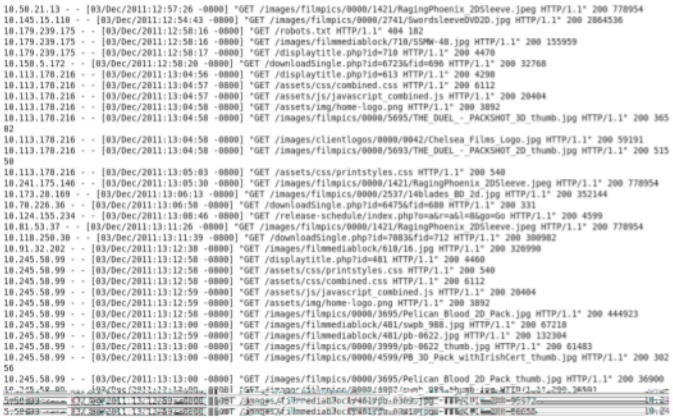
Organizations have been generating data since way back, but as time goes on, more and more data is being generated. IBM estimates that as much as 90% of the data in the world today has been created in the last two years alone.



Just as a simple example, think about your cell phone. Whenever it's turned on, it's connecting to cell towers to get reception. As you move around, it will connect to different towers, and at different signal strengths depending on how far away from them you are. All of that connection data is collected by the phone company, and it's logged.

They can use it to find dead spots in their coverage, to work out which towers are the busiest and need increased capacity... they can even trace you if you make an emergency call but don't give your exact location. That's an enormous amount of data right there.

Another example is when you visit a Website like Amazon or Netflix. Everything you do there is logged: what pages you viewed, what products you looked at, how long you spent on each page... even things like what Web browser you were using and what sort of computer you were connecting from. Again, huge amounts of data.



And that's just in the corporate world. In medicine, for example, each X-Ray creates huge amounts of potentially incredibly valuable information, and comparing large numbers of them can help us to detect similarities in tumors.

This increase in the amount of data we're generating opens up huge possibilities. But it comes with problems too. We have to store all that data, and we have to be able to process it in a sensible amount of time.

Quiz: What is a Big Data problem?

This course is about Hadoop, and how it helps to deal with Big Data. But not everything is actually a big data problem. There are lots of cases where you can use traditional systems to store, manage, and process your data. So the first thing you need to do is decide if what you have really does fall under the heading of 'big data' in the first place. And to make that call, we have to create some kind of definition for what big data is.

Let's start with a quick question. Which of these would you consider to be 'big data'? You are not going to be graded on this answer, but give it your best guess.

- [ ] order details for a purchase at a store
- [ ] all orders across hundreds of branches nationwide
- [ ] information about a person's stock portfolio
- [ ] all stock transactions made on the New York Stock Exchange during the year

Answer:

For most people, the answers are going to be 2 and 4. A list of purchases at a single store is

almost certainly small enough to be easily handled by a traditional relational database system -- or even just a spreadsheet. Orders from hundreds of stores nationwide, though, could start to overwhelm traditional systems. Likewise, information about a single person's stock portfolio is a small and easily managed chunk of data. But data on trades across the entire NYSE for a year will run into tens or hundreds of terabytes -- and that's where traditional systems really do start to struggle.

Definition of Big Data

There's no one definition for 'big data' -- it's a very subjective term. Most people would consider a data set of terabytes or more to be 'big data', but there are certainly people using Hadoop with great success on smaller chunks of data than that. One reasonable definition is that it's data which can't comfortably be processed on a single machine.

Quiz: Challenges

But Big Data is more than just size of the data. What additional problems can you see in this field?

[ ] most data is worthless and it's hard to find the useful parts
[ ] it's hard to gather data (視頻中沒有此項)
[ ] data is created very fast
[ ] data from different sources is in different formats

Answer:

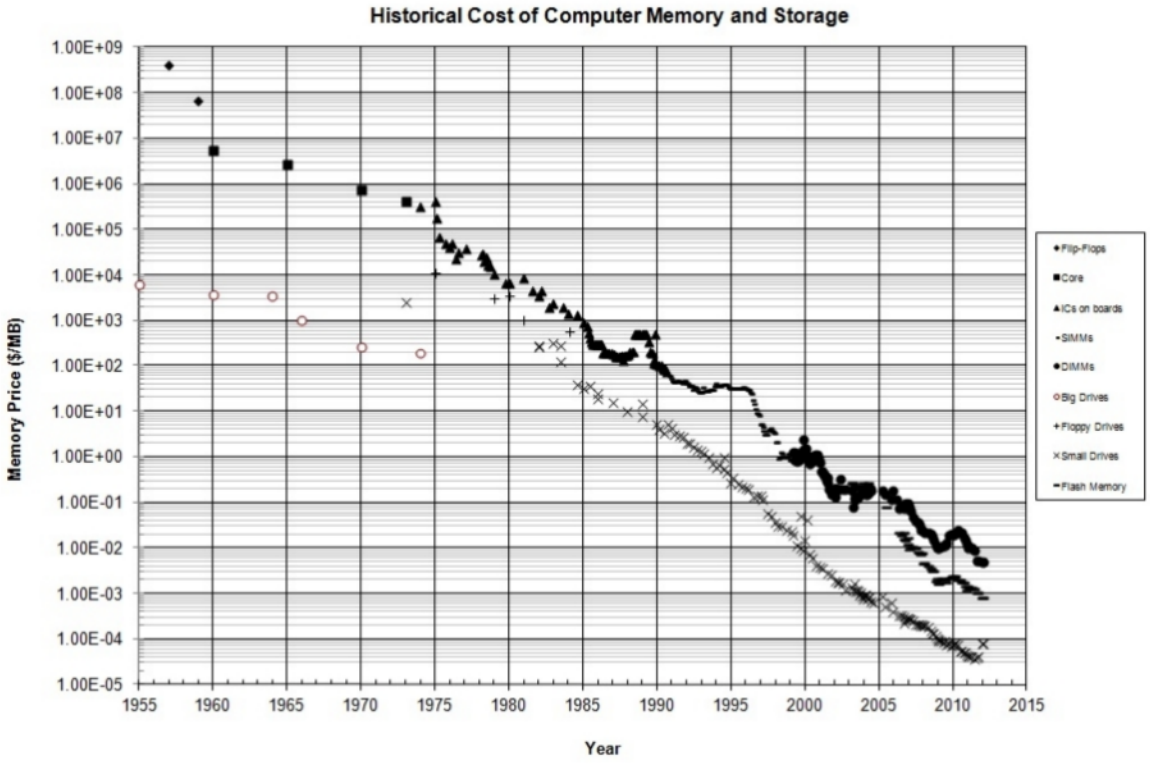
A potential challenge with big data is that it is created very fast and does come from different sources which could come in a variety of formats. In my experience, most data is not worthless but actually does have a lot of value.

The 3 V's of Big Data:

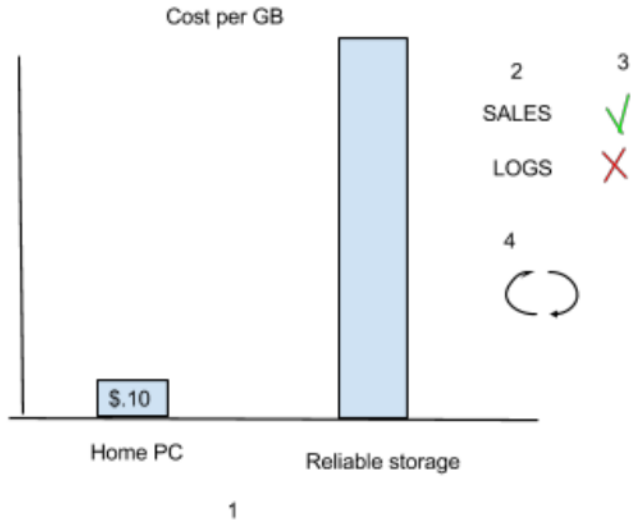
When you read or talk about Big Data, you'll often hear people refer to the 'three Vs'. **V**olume refers to the size of data that you're dealing with, **V**ariety refers to the fact that the data is often coming from lots of different sources and in many different formats, and **V**elocity refers to the speed at which the data is being generated, and the speed at which it needs to be made available for processing. So let's look in more detail at each of them.

Volume

The price to store data has dropped incredibly over the last 60 years. In 1980, the cost per gigabyte was several hundred thousand dollars. In 2013, it's well under 10 cents.



Although it's worth saying that if you actually want to store the data reliably, you're going to end up paying rather more than that -- probably several dollars per gigabyte, maybe even more.



That's particularly the case with more traditional data storage devices such as storage area networks, or SANs, which can be extremely expensive. The high cost of reliable storage puts a cap on the amount of data companies can practically store. At some point, they'd say, "OK, it's too expensive to store all that data that I'm not doing anything with. Let's just store the critical stuff: my actual sales, for example, rather than all that stuff about how long people spent on each page of my Web site." But it turns out, as we'll see, that the data they're currently throwing away can be incredibly useful. What we need is a cheaper way to store it reliably.

And of course storing the data is only one part of the equation; you also need to be able to read

and process it efficiently. Storing a terabyte of data on a SAN isn't so hard, but streaming the data from the SAN across the network to some central processor can take a long time, and processing it can be extremely slow.

QUIZ: Volume

Which of the following data do you think is worth storing and analyzing?

[ ] transactions (financial, government related)
[ ] logs (records of activity, location)
[ ] business data (product catalogs, prices, customers)
[ ] user data (images, documents, video)
[ ] sensor data (temperature, pollution)
[ ] medical data (x-rays, brain activity records)
[ ] social (email, twitter etc)

Answer

And the answer is that all of these can provide useful information. But in order to store it, you'll need a way to scale your storage capacity up to massive volume. Hadoop, which stores data in a distributed way across multiple machines, does that. You'll see just how in the next lesson.

Variety

The second 'V' is data variety. For a long time, people have used databases to store and process their data -- either smaller databases like MySQL, or big data warehouses based on software from companies like Oracle and IBM. But for a data warehouse to effectively process information, all that information has to fit nicely into a predefined set of tables. The problem is that these days, lots of the data you want to store is what we tend to call 'unstructured data', or 'semi-structured data'. Sarah can give us some examples.

By 'unstructured', we mean the data arrives in lots of different formats. For example, a bank might have a list of your credit card and account transactions, but they may also have scans of your checks, records of your interactions with customer service representatives on the Web and over the phone, perhaps even recordings of those phone calls. All of that data is in a variety of different formats, and it can be hard to store and reconcile it all using traditional systems.

And this also ties back to volume. You want to store that data in its original format so you're not throwing any information away. That way you can then process the data later in different ways you might not even have thought of originally.

For instance, if we just transcribe call center conversations into text, we have what people said to our customer service representatives. But if we have the actual recordings, then later on we might develop software which can interpret the tone of voice the customer uses… and that might lead to a very different interpretation of the data. And the nice thing about Hadoop is that it doesn't care what format the data comes in. Unlike a traditional database, you can just store the data in its raw format, and manipulate and reformat it later.

Quiz: Data Variety

Sometimes the most unlikely data can be extremely useful and lead to savings due to better planning. For example, a conventional system for coordinating logistics system might send the closest truck to the warehouse to pick up the package. However, it might be that the closest truck is not the best solution -- perhaps there are traffic jams, or the most direct route is on small roads that would take longer to drive. Maybe the truck doesn't have enough free space for the new load. So what kind of data would be helpful in making a better plan that could save money and time for the company?

[ ] Current GPS location from all trucks
[ ] Current itineraries for all trucks
[ ] Current traffic speed in related areas as reported by services such as Waze
[ ] Current load of trucks by volume and weight
[ ] Fuel efficiency of the different vehicles

Answer:

And again - all of these answers are correct. You can save a lot of money, and time, by making better decisions, driven by more varied data. The world we live in is extremely complex, and there are a lot of variables to consider that you can tweak to get large benefits.

Velocity

Velocity, the third V, is about the speed at which the data arrives, ready to be processed. We need to be able to accept and store that data -- even when it's coming in at a rate of terabytes or more a day, which is often the case. If we can't store it as it arrives, we'll end up discarding some of it, and that's what we absolutely want to avoid.

What problems can we solve?

Think about an e-commerce Web site. If we know what products you've looked at in the past, we could recommend similar products the next time you visit our site. If you spent five minutes looking at a particular item, we could maybe send you an email informing you when that item is on sale. If we know that you typically browse our site using a first-generation iPad, we could suggest the latest model.

This is a huge difference to what we would do before, when we only stored records of actual purchases. If we can store and process all of our Web server log files, along with the purchase data that's in our traditional data warehouse, we can give the customer a much better shopping experience -- which should directly translate into bigger profits.

Yet another example is a movie site like Netflix. Based on what they know about your viewing habits, they can recommend movies to you -- as you can see here, because of what Ian's rated highly before, the movie on the left is recommended for him -- and they can even predict what rating he'll give the movie.

History of solving data problems

So there are plenty of things we can do with 'big data'. But first we have to solve a couple of problems. We need to be able to store the data in a cost-effective way, and we need to be able to process it efficiently. And it turns out that these are not easy problems to solve when we're talking about massive amounts of data. Fortunately, though, some extremely smart people at Google were working on them in the late 1990s and released the results of their work as research papers in 2003 and 2004. Let's see what Doug Cutting, one of the founders of Hadoop, has to say.

DOUG CUTTING about History of Hadoop:

So, let me tell you how Hadoop came to be. About ten years ago in around 2003, I was working on an Open Source web search engine called Nutch, and we knew it needed to be something very scalable, because the Web was you know, billions of pages. terabytes, petabytes, of data, that we needed to be able to process, and we set about doing the best job we could and it was tough. We got things up and running on four or five machines, not very well, and around that time Google published some papers about how they were doing things internally. Published a paper about their distributed file system, TFS. and about their processing, framework, MapReduce. So my partner and I, at the time, in this project, Mike Cafarella. said about trying to reimplement these in Open Source. So that more people could use them than just folks at Google. Took us a couple of years, and we had Nutch up and running on, instead of four or five machines, on, 20 to 40 machines. It wasn't perfect, it wasn't totally reliable, but it worked. And we realize that to get it to the point where it was scaled to thousands of machines, and be as bullet proof as it needed to be, would take more than just the two of us, working part time.

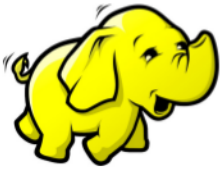
Around that time, Yahoo approached me and said they were interested in investing in this. So I went to work for Yahoo in January of 2006. First thing I did there, was, we took the parts of Nutch that were a distributed computing platform, and put them into a separate project. A new project christened Hadoop. Over the next couple years, with, Yahoo's help, and the help of others, we took Hadoop, and really got it to the point where it did scale to petabytes, and running on thousands of processors. And doing so quite reliably.

It spread to lots of companies, and mostly in the Internet sector, and became quite a success. after that, we, we started to see a bunch of other projects grow up around it. And Hadoop's grown to be the kernel of a, which, pretty much an operating system for big data. We've got tools that, allow you to, more easily do, MapReduce programming, so, you can develop using SQL or a data flow language called Pig. And
we've also got the beginnings of higher-level tools. We've got interactive SQL with Impala. We've got Search. and so we're really seeing this develop to being a general purpose platform for data processing. that scale's much better and that it is much more flexible than anything that's, that's, else is out there.

That's the story of the genesis of Hadoop: it's based on work done by the folks at Google, and it's grown from small beginnings to the point now where hundreds of people contribute to the project, and where it's being used by thousands and thousands of companies worldwide. The

Hadoop logo is actually a little yellow elephant, but do you know where the name came from? There's a funny story attached to that. Here's Doug again.

DOUG about Name of Hadoop



So the name Hadoop comes from my son's toy elephant. When he was about two, a friend gave him a little stuffed elephant which he played with incessantly. And we overheard him calling it something, this strange word that he invented, and said Hadoop. So I immediately wrote it down because I was in the software business. And we're always looking for good names. And this one came with a mascot, even. And a few years later when I needed a project name, pulled it out. Now, I wrote it down as H A D O O P. And figured that everyone would say Hadoop. Now it turns out everyone says Hadoop instead, but I persist in saying Hadoop. Now my son, of course, is 13, and expects royalties for the name. He he wants more credit. He also accuses me of stealing the toy. At some point, he was using it in some kind of rocket ship experiment, and I had to rescue it. And now it, it lives in my sock drawer for, for safety.

Hadoop Cluster

此話總結得好, 我之前看的兩個課(Big data, Cousera Hadoop)都沒說清楚

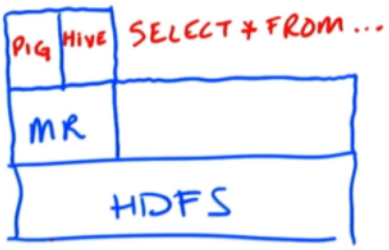
The core Hadoop project consists of a way to store data, known as the Hadoop Distributed File System, or HDFS, and a way to process the data, called MapReduce. The key concept is that we split the the data up and store it across a collection of machines, known as a cluster. Then, when we want to process the data, we process it where it's actually stored. Rather than retrieving the data from a central server, instead it's already on the cluster, and we can process it in place. You can add more machines to the cluster (make the cluster bigger) as the amount of data you're storing grows -- and, indeed, many people start with just a few machines and add more as they're needed. The machines in the cluster don't need to be particularly high-end; although most clusters are built using rack-mount servers, they are typically mid-range servers rather than top-of-the-range equipment.

Hadoop Ecosystem

以下講得很好, 我之前看的那兩門課都沒講清楚

Core Hadoop consists of HDFS and MapReduce.

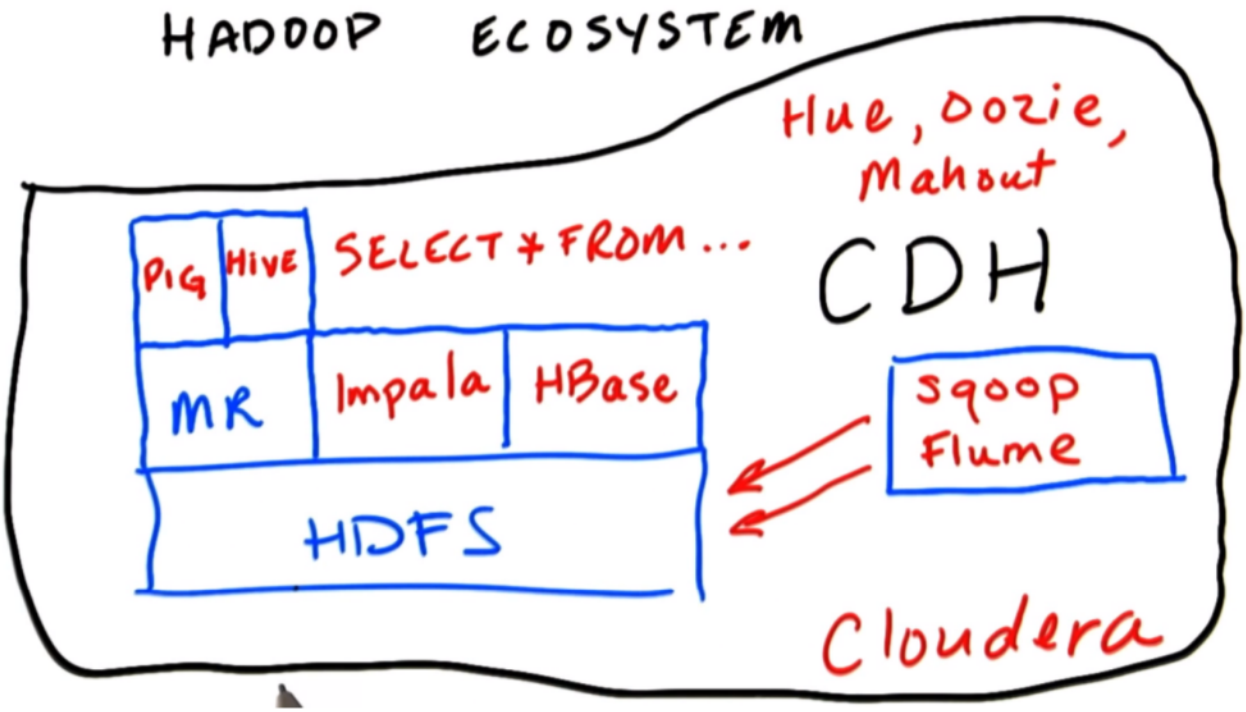
But since the project was first started, an awful lot of other software has grown up around it. And that's what we call the Hadoop Ecosystem. Some of the software is intended to make it easy to load data into the Hadoop cluster, while lots of it is designed to make Hadoop easier to use. For example, as you'll see in the next lesson, writing MapReduce code isn't completely simple. You need to know a programming language like Java, or Python, or Ruby, or Perl. But there are lots of folks out there who aren't programmers but who can write SQL queries to access data in a traditional relational database like SQL Server. And of course a lot of business intelligence tools also want to hook into Hadoop.



For that reason, other open source projects have been created to make it easier for people to query their data without knowing how to code. Two key ones are Hive and Pig. Instead of having to write Mappers and Reducers, in Hive you just write statements, which look very much like standard SQL. The Hive interpreter turns that SQL into MapReduce code, which it then runs on the cluster. And an alternative is Pig, which allows you to write code to analyse your data in a fairly simple scripting language rather than MapReduce -- again, the code is turned into actual Java MapReduce and run on the cluster.

Hive and Pig are great, but they're still running MapReduce jobs, which mean they will take a reasonable amount of time, especially when running on really large amounts of data. So another open source project called Impala was developed which again allows you to query your data using SQL but which directly accesses that data, rather than accessing it via MapReduce. Impala is optimized for low-latency queries -- in other words, Impala queries run very quickly, typically many times faster than Hive queries -- while Hive is optimized for long-running batch processing jobs.

Another project used by many people is Sqoop. That takes data from a traditional relational database server such as Microsoft SQL Server and puts it in HDFS as delimited files so it can be processed along with the other data on the cluster. Then there's Flume, which ingests data as it's generated by external systems. HBase is a real-time database built on top of HDFS. Hue is a graphical front-end to the cluster. Oozie is a workflow management tool. Mahout is a machine learning library…



In fact, there are so many different ecosystem projects that making them all talk to each other, and work well with each other, can be tricky. To make installing and maintaining a cluster easier, Cloudera, the company we work for, has put together a distribution of Hadoop called CDH. This takes all the key ecosystem projects, along with Hadoop itself, and packages them together so that installation is a really simple process. And the components are all tested together, so you can be sure that there are no incompatibilities between them. Of course it's completely free and open source, just like Hadoop itself. You could install everything from scratch yourself, but it's far easier to use CDH, and that's certainly what we'd recommend. In the next lesson, in fact, you'll be downloading and running a virtual machine which has CDH installed.

Conclusion

So in this lesson you learned what 'big data' is, and how Hadoop can help with big data problems. In the next lesson, we'll take a deeper look at the two key parts of Hadoop: that's HDFS, the Hadoop Distributed File System, and MapReduce, the way you can process that data.