CS 535 Large Scale Data Analysis

Amit Jain

Big Data, Big Disks, Cheap Computers

- *"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."* Rear Admiral Grace Hopper.

Big Data, Big Disks, Cheap Computers

- *"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."* Rear Admiral Grace Hopper.
- *"More data usually beats better algorithms."* Anand Rajaraman.

Big Data, Big Disks, Cheap Computers

- *"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."* Rear Admiral Grace Hopper.
- *"More data usually beats better algorithms."* Anand Rajaraman.
- *"The good news is that Big Data is here. The bad news is that we are struggling to store and analyze it."* Tom White.

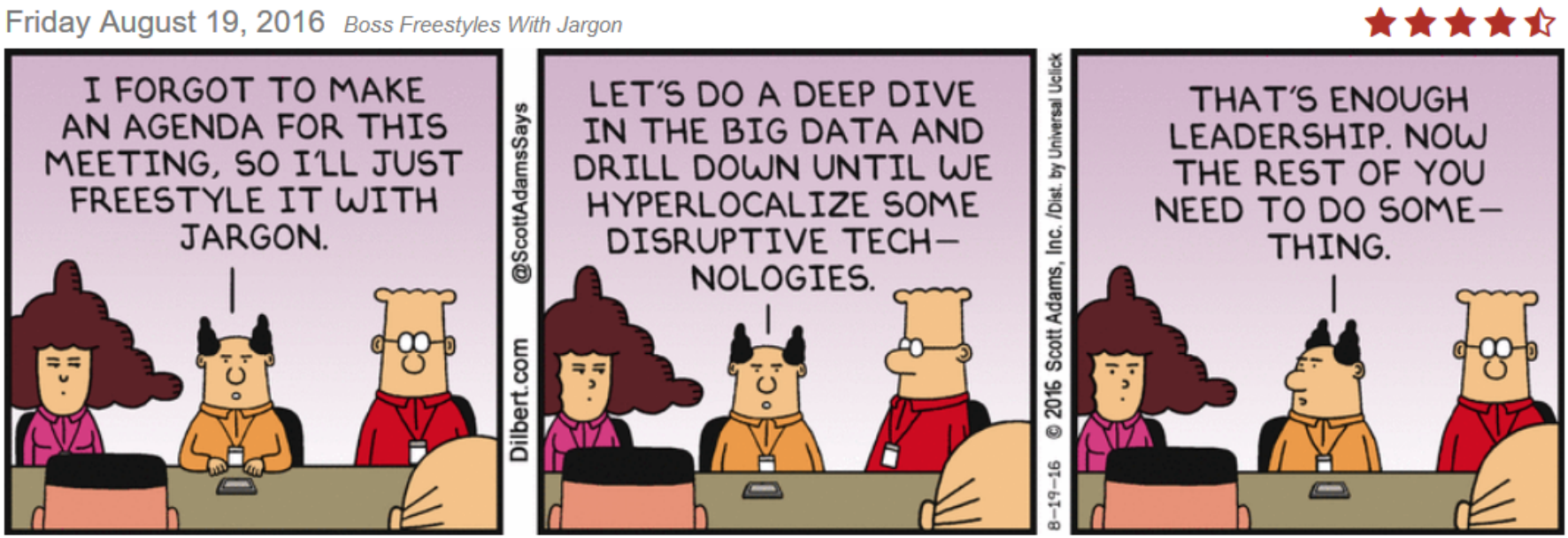
Units and Units

Check out http://en.wikipedia.org/wiki /Petabyte

Big Data


Big Data knows everything
CONSULTANTS SAY THREE QUINTILLION BYTES OF DATA ARE CREATED EVERY DAY.
DilbertCartoonist@gmail.com
IT COMES FROM EVERYWHERE. IT KNOWS ALL.
ACCORDING TO THE BOOK OF WIKIPEDIA, ITS NAME IS "BIG DATA."
BIG DATA
©2012 Scott Adams, Inc.
BIG DATA LIVES IN THE CLOUD. IT KNOWS WHAT WE DO.
IN THE PAST, OUR COMPANY DID MANY EVIL THINGS.
BUT IF WE ACCEPT BIG DATA IN OUR SERVERS, WE WILL BE SAVED FROM BANKRUPTCY.
LET US PAY.
$
www.dilbert.com
7-29-12
IS IT TOO LATE TO SIDE WITH EVIL?
SHHHH! IT HEARS YOU.

Big Data


Friday August 19, 2016 Boss Freestyles With Jargon
I FORGOT TO MAKE AN AGENDA FOR THIS MEETING, SO I'LL JUST FREESTYLE IT WITH JARGON.
LET'S DO A DEEP DIVE IN THE BIG DATA AND DRILL DOWN UNTIL WE HYPERLOCALIZE SOME DISRUPTIVE TECH-NOLOGIES.
THAT'S ENOUGH LEADERSHIP. NOW THE REST OF YOU NEED TO DO SOME-THING.
Dilbert.com @ScottAdamsSays
8-19-16 © 2016 Scott Adams, Inc. /Dist. by Universal Uclick

Big Data


DO WE HAVE ANY ACTIONABLE ANALYTICS FROM OUR BIG DATA IN THE CLOUD?
YES, THE DATA SHOWS THAT MY PRODUCTIVITY PLUNGES WHENEVER YOU LEARN NEW JARGON.
MAYBE IN-MEMORY COMPUTING WILL ACCEL-ERATE YOUR APPLICA-TIONS.
PLUNGE, PLUNGE, PLUNGE.
Dilbert.com DilbertCartoonist@gmail.com
1-9-13 ©2013 Scott Adams, Inc./Dist. by Universal Uclick

Word-count: Hello World of Big Data

Problem: Given a collection of text files, find the frequency of each word.

Word-count: Hello World of Big Data

Problem: Given a collection of text files, find the frequency of each word.

For example:

File1.txt	File2.txt	File3.txt
large	data	huge
big	data	big
large	data	small
big	data	deluge

Word-count: Hello World of Big Data

Problem: Given a collection of text files, find the frequency of each word.

For example:

File1.txt	File2.txt	File3.txt
large	data	huge
big	data	big
large	data	small
big	data	deluge

Result:

```
large 2
big 3
data 4
small 1
deluge 1
huge 1
```

Questions:

Word-count: Hello World of Big Data

Problem: Given a collection of text files, find the frequency of each word.

For example:

File1.txt	File2.txt	File3.txt
large	data	huge
big	data	big
large	data	small
big	data	deluge

Result:

```
large 2
big 3
data 4
small 1
deluge 1
huge 1
```

Questions: Do we want the output sorted by frequency?

Word-count: Hello World of Big Data

Problem: Given a collection of text files, find the frequency of each word.
For example:

File1.txt	File2.txt	File3.txt
large	data	huge
big	data	big
large	data	small
big	data	deluge

Result:

```
large 2
big 3
data 4
small 1
deluge 1
huge 1
```

Questions: Do we want the output sorted by frequency? Sorted by word?

Word-count: Hello World of Big Data

Problem: Given a collection of text files, find the frequency of each word. For example:

File1.txt	File2.txt	File3.txt
large	data	huge
big	data	big
large	data	small
big	data	deluge

Result:

large 2
big 3
data 4
small 1
deluge 1
huge 1

Questions: Do we want the output sorted by frequency? Sorted by word? How would you solve this problem?

Sequential Solutions (1)

```
create empty dictionary
for f over all input files
    open file f
    while not end of file f
        read next word
        if search(word, dictionary)
            increment frequency count for word
        else
            add word to the dictionary

open output file
iterate over dictionary
    write next word to output file
```

Sequential Solutions 1 (Analysis)

- Let's say that the size of all the files together is O(n).

Sequential Solutions 1 (Analysis)

- Let's say that the size of all the files together is O(n).
- We then have $O(n)$ search operations in the dictionary. The time for the search depends on how we implement the dictionary.

Sequential Solutions 1 (Analysis)

- Let's say that the size of all the files together is O(n).
- We then have $O(n)$ search operations in the dictionary. The time for the search depends on how we implement the dictionary.
  - Hashtable: $O(1)$ average time
  - Height balanced tree: $O(\lg n)$ worst-case time

Sequential Solutions 1 (Analysis)

- Let's say that the size of all the files together is O(n).
- We then have $O(n)$ search operations in the dictionary. The time for the search depends on how we implement the dictionary.
  - Hashtable: $O(1)$ average time
  - Height balanced tree: $O(\lg n)$ worst-case time
- So the main loop takes $O(n)$ time on average and $O(n \lg n)$ in the worst case

Sequential Solutions 1 (Analysis)

- Let's say that the size of all the files together is O(n).
- We then have $O(n)$ search operations in the dictionary. The time for the search depends on how we implement the dictionary.
  - Hashtable: $O(1)$ average time
  - Height balanced tree: $O(\lg n)$ worst-case time
- So the main loop takes $O(n)$ time on average and $O(n \lg n)$ in the worst case
- The time to output is insignificant as the size of the dictionary will be much smaller than n. Why?

Examples

- See CS535-resources git repo for example implementations
  - WordCount.java for a Java solution using a dictionary

Examples

- See CS535-resources git repo for example implementations
  - WordCount.java for a Java solution using a dictionary
  - wordcount.sh for a streaming solution in a shell script:

```
cat input/* | tr ' ' '\n' | sort | uniq -c
```

Examples

- See CS535-resources git repo for example implementations
  - WordCount.java for a Java solution using a dictionary
  - wordcount.sh for a streaming solution in a shell script:

    ```
    cat input/* | tr ' ' '\n' | sort | uniq -c
    ```

  - The first command `cat` outputs all the files into one stream to the next program `tr` in the pipeline.

Examples

- See CS535-resources git repo for example implementations
  - WordCount.java for a Java solution using a dictionary
  - wordcount.sh for a streaming solution in a shell script:

    ```
    cat input/* | tr ' ' '\n' | sort | uniq -c
    ```

  - The first command `cat` outputs all the files into one stream to the next program `tr` in the pipeline.
  - The `tr` command breaks the words in each line into a line by itself and then streams them to the `sort` command.

Examples

- See CS535-resources git repo for example implementations
  - WordCount.java for a Java solution using a dictionary
  - wordcount.sh for a streaming solution in a shell script:

    ```
    cat input/* | tr ' ' '\n' | sort | uniq -c
    ```

  - The first command `cat` outputs all the files into one stream to the next program `tr` in the pipeline.
  - The `tr` command breaks the words in each line into a line by itself and then streams them to the `sort` command.
  - After sorting, all instances of a word are together, which `uniq -c` counts and outputs.

Large Scale Word-Count (1)

- What if the the number of files is in millions and will not fit in one server?

Large Scale Word-Count (1)

- What if the the number of files is in millions and will not fit in one server?
- What if the total size of the files is in Petabytes and will not fit in one server?

Large Scale Word-Count (1)

- What if the the number of files is in millions and will not fit in one server?
- What if the total size of the files is in Petabytes and will not fit in one server?
- How do you modify your solution from before? Assume that you have a cluster of n servers available with the files distributed across the servers.

Large Scale Word-Count (1)

- What if the the number of files is in millions and will not fit in one server?
- What if the total size of the files is in Petabytes and will not fit in one server?
- How do you modify your solution from before? Assume that you have a cluster of n servers available with the files distributed across the servers.
- But how do we create a cluster and get the files on it?

Large Scale Word-Count (2)

- What if some of the servers fail while running your program?

Large Scale Word-Count (2)

- What if some of the servers fail while running your program?
- What if some of the server disks fail or get corrupted while your program is running?

Large Scale Word-Count (2)

- What if some of the servers fail while running your program?
- What if some of the server disks fail or get corrupted while your program is running?
- What if the some system administrator reboots some of your servers for software/hardware updates without letting you know?

Frameworks/Systems for Distributed Storage and Analysis

- MapReduce: A framework.

Frameworks/Systems for Distributed Storage and Analysis

- MapReduce: A framework.
- Hadoop and MapReduce: batch processing

Frameworks/Systems for Distributed Storage and Analysis

- MapReduce: A framework.
- Hadoop and MapReduce: batch processing
- Hive and Hadoop HDFS: SQL on top of MapReduce

Frameworks/Systems for Distributed Storage and Analysis

- MapReduce: A framework.
- Hadoop and MapReduce: batch processing
- Hive and Hadoop HDFS: SQL on top of MapReduce
- Spark and Hadoop HDFS: faster batch processing and streaming

Frameworks/Systems for Distributed Storage and Analysis

- MapReduce: A framework.
- Hadoop and MapReduce: batch processing
- Hive and Hadoop HDFS: SQL on top of MapReduce
- Spark and Hadoop HDFS: faster batch processing and streaming
- Storm and Heron: large real-time data flow

Frameworks/Systems for Distributed Storage and Analysis

- MapReduce: A framework.
- Hadoop and MapReduce: batch processing
- Hive and Hadoop HDFS: SQL on top of MapReduce
- Spark and Hadoop HDFS: faster batch processing and streaming
- Storm and Heron: large real-time data flow
- and more...

Data Scientist versus Data Engineer

Data Science tasks:

- Goal is to answer a question or discovering insights.

Data Scientist versus Data Engineer

Data Science tasks:

- Goal is to answer a question or discovering insights.
- Often uses interactive shells for adhoc analysis

Data Scientist versus Data Engineer

Data Science tasks:

- Goal is to answer a question or discovering insights.
- Often uses interactive shells for adhoc analysis
- Typically uses Python, R, Matlab, and Spark

Data Scientist versus Data Engineer

Data Science tasks:

- Goal is to answer a question or discovering insights.
- Often uses interactive shells for adhoc analysis
- Typically uses Python, R, Matlab, and Spark

Data engineer tasks:

- Builds and maintains a production application (that may use hardened versions of the original data science work)

Data Scientist versus Data Engineer

Data Science tasks:

- Goal is to answer a question or discovering insights.
- Often uses interactive shells for adhoc analysis
- Typically uses Python, R, Matlab, and Spark

Data engineer tasks:

- Builds and maintains a production application (that may use hardened versions of the original data science work)
- Use principles of software engineering like encapsulation, object-oriented design and interface design

Data Scientist versus Data Engineer

Data Science tasks:

- Goal is to answer a question or discovering insights.
- Often uses interactive shells for adhoc analysis
- Typically uses Python, R, Matlab, and Spark

Data engineer tasks:

- Builds and maintains a production application (that may use hardened versions of the original data science work)
- Use principles of software engineering like encapsulation, object-oriented design and interface design
- They have to deal with parallelization, complexity of distributed systems, fault tolerance etc

Data Scientist versus Data Engineer

Data Science tasks:

- Goal is to answer a question or discovering insights.
- Often uses interactive shells for adhoc analysis
- Typically uses Python, R, Matlab, and Spark

Data engineer tasks:

- Builds and maintains a production application (that may use hardened versions of the original data science work)
- Use principles of software engineering like encapsulation, object-oriented design and interface design
- They have to deal with parallelization, complexity of distributed systems, fault tolerance etc
- Typical languages would be Java (with Hadoop and/or Spark), Scala, Python (at smaller scales)

Insights from Big Data

The point of large scale data analysis is meaningful insight!
We should consider two things about insights presented by analysis:

- Investigate carefully to see if it uses a significant amount of data.
- Think about each of the insights and label them Actionable, Useless (trivia), or potentially Misleading or dangerous.

For example:
https://blogs.scientificamerican.com/guest-blog/9-bizarre-and-surprising-insights-from-data-science/