Lambda School Data Science

Unit 3 - Data Engineering

Sprint 3 - Big Data

Module 2 - Scala for Spark

Why Scala? Why Spark?

Let's compare:

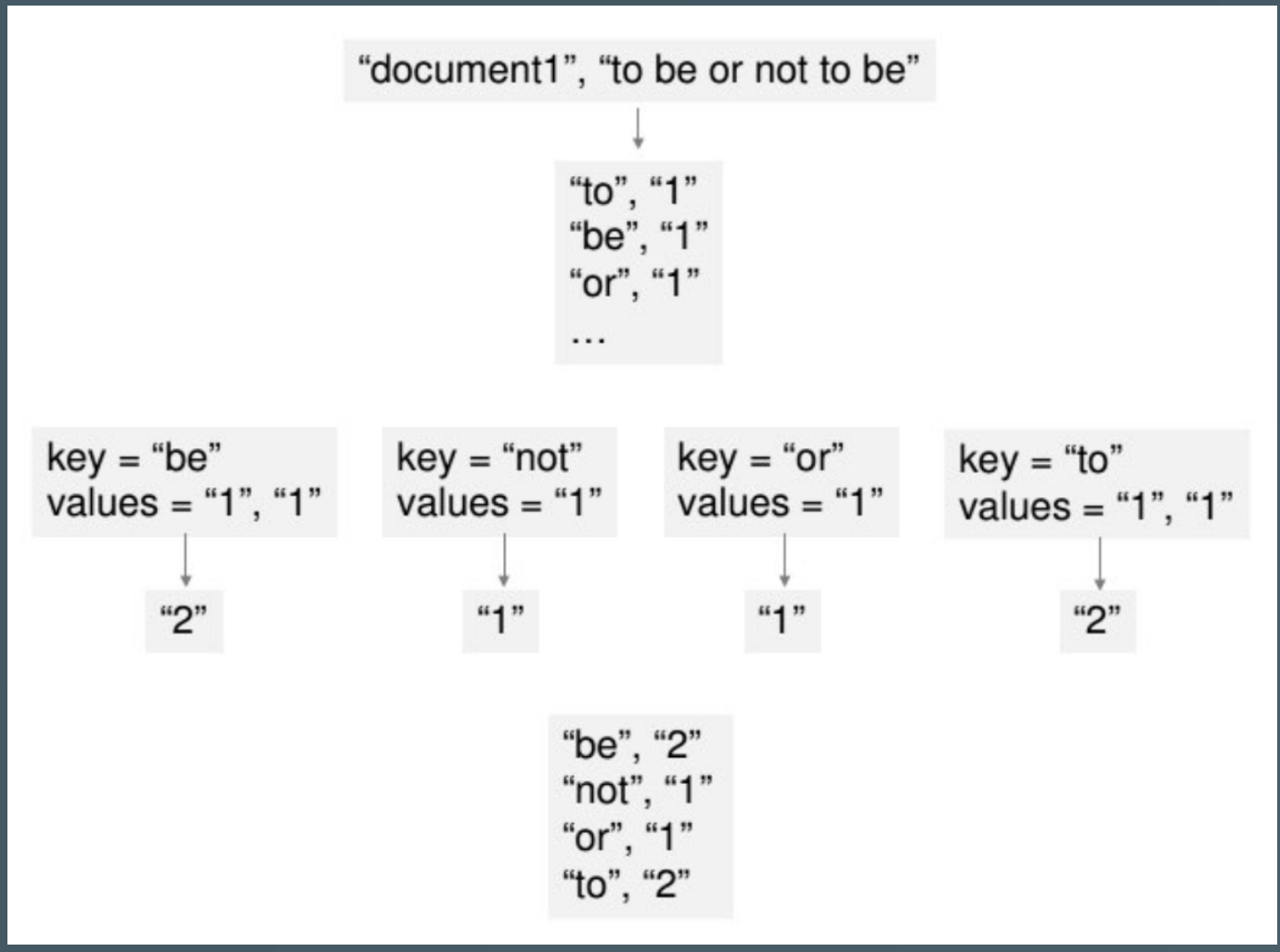
- Hadoop MapReduce in Java
- Spark in Scala

MapReduce

- Read a lot of data
- **Map:** extract something you care about from each record
- Shuffle and sort
- **Reduce:** aggregate, summarize, filter, or transform
- Write the results

  This outline stays the same. The mapper and reducer change to fit the problem.

Word Count in MapReduce



```
class WordCountMapper extends MapReduceBase
    implements Mapper<IntWritable, Text, Text, IntWritable

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

  @Override public void map(IntWritable key, Text valueCon
      OutputCollector<Text, IntWritable> output, Reporter
    String[] tokens = valueContents.toString.split("\\s+")
    for (String wordString: tokens) {
      if (wordString.length > 0) {
        word.set(wordString.toLowerCase);
        output.collect(word, one);
      }
    }
  }
}
```

Alex Payne, Dean Wampler, *Programming Scala, 2nd Edition*, Chapter 18: Scala for Big Data

```
class WordCountReduce extends MapReduceBase
    implements Reducer<Text, IntWritable, Text,
    IntWritable> {

  public void reduce(Text keyWord,
      java.util.Iterator<IntWritable> counts,
      OutputCollector<Text, IntWritable> output,
      Reporter reporter) {

    int totalCount = 0;
    while (counts.hasNext) {
      totalCount += counts.next.get;
    }
    output.collect(keyWord, new IntWritable(totalCount));
  }
}
```

Alex Payne, Dean Wampler, *Programming Scala, 2nd Edition*, Chapter 18: Scala for Big Data

Why was MapReduce good when it was new?

Didn't have to write code for

- Distributed processing
- Fault tolerance

If you're curious about the history, see:

- The Friendship That Made Google Huge
- MapReduce original research paper

How did Spark improve upon MapReduce?

- Concise language
- Flexible APIs
- Persist data in memory

Word Count in Spark (old RDD API)

```
object SparkWordCount {
  def main(args: Array[String]) = {
    val sc = new SparkContext("local", "Word Count")

    val input = sc.textFile(args(0)).map(_.toLowerCase)
    input
      .flatMap(line => line.split("""\W+"""))
      .map(word => (word, 1))
      .reduceByKey((count1, count2) => count1 + count2)
      .saveAsTextFile(args(1))

    sc.stop()
  }
}
```

Alex Payne, Dean Wampler, *Programming Scala, 2nd Edition*, Chapter 18: Scala for Big Data

More Spark examples

https://spark.apache.org/examples.html

Compare:

- Scala vs Python
- RDD API vs DataFrame API

Spark vs Dask

Dask Documentation: Comparison to Spark

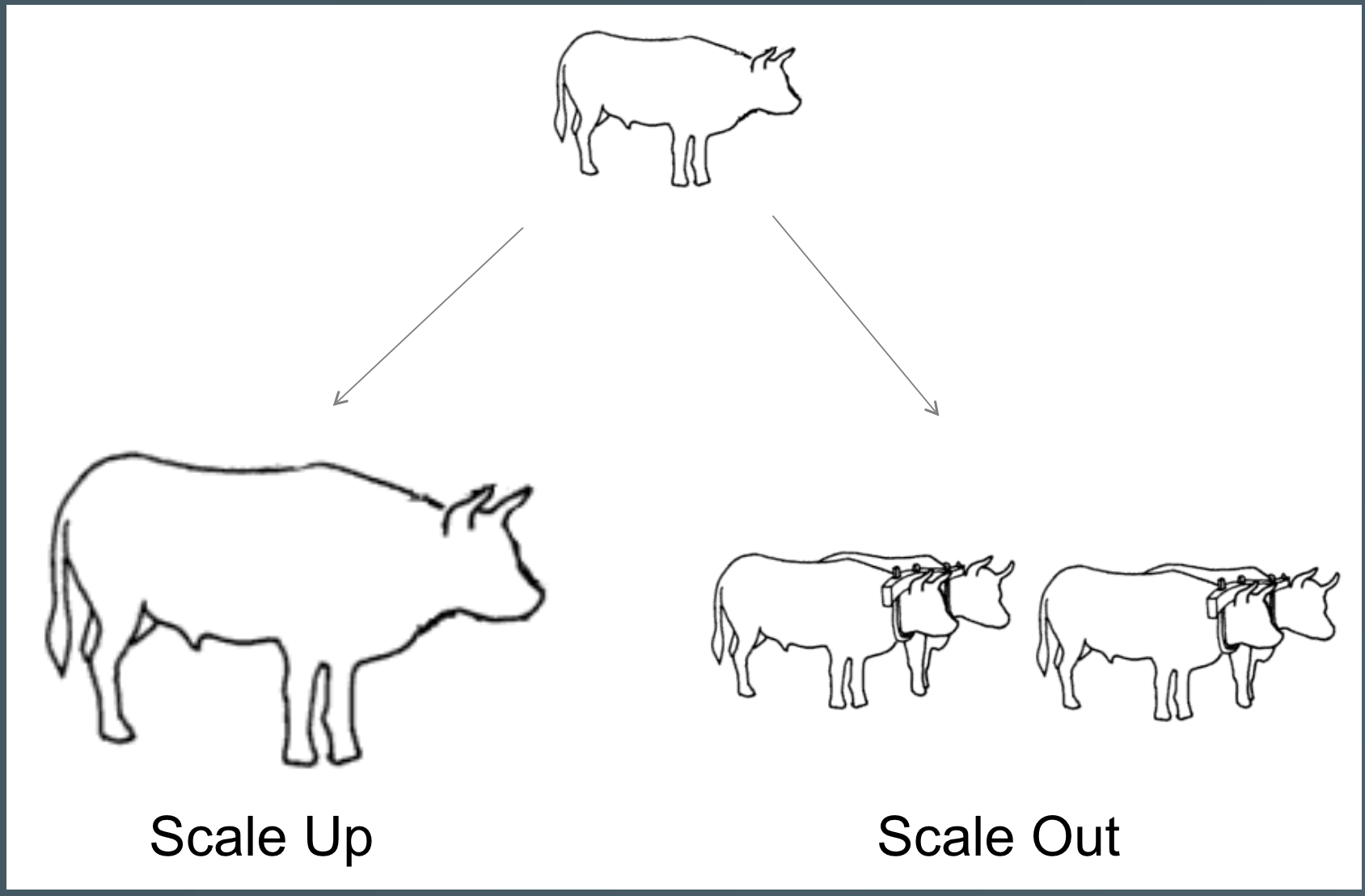
(Sometimes you'll get to choose what you want, sometimes someone else at your company will)

Argument for scaling out

“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

“When one ox couldn't budge a log ...”



Argument for scaling up

Gary Bernhardt tweets about "big data"

Assignment

[Learn Scala the Hard Way]