A Very Brief Introduction to MapReduce

Diana MacLean for CS448G, 2011

What is MapReduce?

MapReduce is a software framework for processing (large[1]) data sets in a distributed fashion over a several machines. The core idea behind MapReduce is *mapping* your data set into a collection of <key, value> pairs, and then *reducing* over all pairs with the same key. The overall concept is simple, but is actually quite expressive when you consider that:

1. Almost all data can be mapped into <key, value> pairs somehow, and
2. Your keys and values may be of any type: strings, integers, dummy types… and, of course, <key,value> pairs themselves.

The canonical MapReduce use case is counting word frequencies in a large text (this is what we'll be doing in Part 1 of Assignment 2), but some other examples of what you can do in the MapReduce framework include:

- Distributed sort
- Distributed search
- Web-link graph traversal
- Machine learning
- …

A MapReduce Workflow

When we write a MapReduce workflow, we'll have to create 2 scripts: the map script, and the reduce script. The rest will be handled by the Amazon Elastic MapReduce (EMR) framework.

When we start a map/reduce workflow, the framework will *split* the input into segments, passing each segment to a different machine. Each machine then runs the *map script* on the portion of data attributed to it.

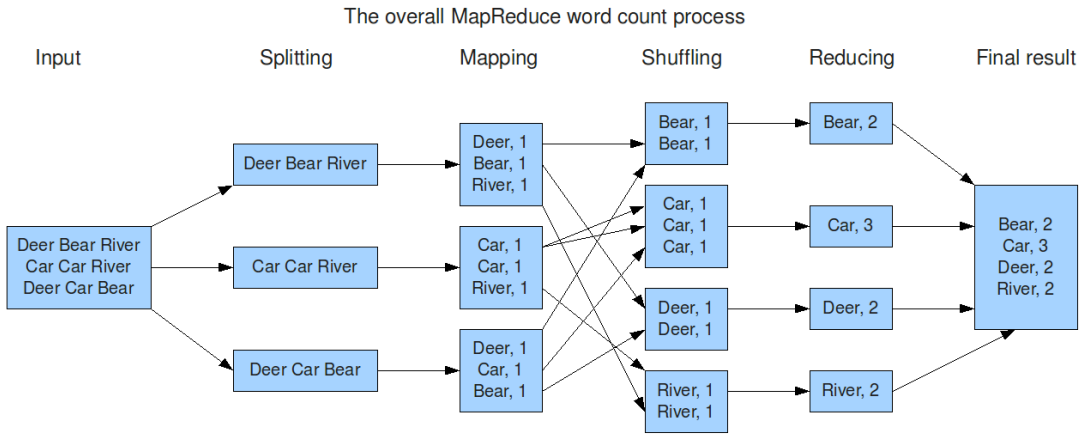
The *map script* (which you write) takes some input data, and maps it to <key, value> pairs according to your specifications. For example, if we wanted to count word frequencies in a text, we'd have <word, count> be our <key, value> pairs. Our map

[1] The data doesn't **have** to be large, but it is almost always much faster to process small data sets locally than on a MapReduce framework. You can see this for yourself when you do Part1a of Assignment 2. Try downloading the Shakespeare corpus and running it through your MapReduce scripts on your local machine.

script, then, would emit a <word, 1> pair for each word in the input stream. Note that the map script does no *aggregation* (i.e. actual counting) – this is what the reduce script it for. The purpose of the map script is to model the data into <key, value> pairs for the reducer to aggregate.
Emitted <key, value> pairs are then "shuffled" (to use the terminology in the diagram below), which basically means that pairs with the same key are grouped and passed to a single machine, which will then run the *reduce script* over them[2].

The *reduce script* (which you also write) takes a collection of <key, value> pairs and "reduces" them according to the user-specified reduce script. In our word count example, we want to count the number of word occurrences so that we can get frequencies. Thus, we'd want our reduce script to simply sum the *values* of the collection of <key, value> pairs which have the same key[3].

The diagram below illustrates the described scenario nicely[4].



Standard Input and Output

If you're curious: you'll notice that the provided scripts (mapper.py and reducer.py) are reading from and writing to standard output. Amazon's framework handles this

---

[2] Note: this is over-simplified, as the workflow may utilize multiple reduce stages, but it accurately imparts the general idea of what's going on.
[3] Note: do not assume that all <key, value> pairs passed to a reduce script will have the same key. If you take a look at the provided reduce.py script, you'll notice that we maintain a *dictionary* of keys. The "shuffling" phase will try to pass pairs with the same key to the same reducer, but due to different rates of computation (from different mappers) and the distributed nature of the workflow (we can just use multiple reduce stages) this will not often be the case.
[4] Diagram from http://blog.jteam.nl/wp-content/uploads/2009/08/MapReduceWordCountOverview1.png

by streaming the output from the map scripts into the reduce scripts; we can always use standard I/O for streaming jobs.

So What does Hadoop have to do with Anything?

Hadoop is Apache's free and open-source implementation of a MapReduce framework (as opposed, for example, to Google's proprietary implementation). Our EMR workflows will be run over the Hadoop framework.

Want to Read More?

This was a very brief overview of MapReduce, designed to get you started on Assignment 2. There's much more you can read! For example…

The Google MapReduce paper gives the nitty-gritty details[5]

www.mapreduce.org has some great resources on state-of the art MapReduce research questions, as well as a good introductory "What is MapReduce" page.

Wikipedia's[6] overview is also pretty good.

Finally, the same group who produced the word-count map/reduce diagram above have a detailed and helpful walkthrough of MapReduce workflows[7]. Although the code snippets are in Java, the overall concepts are clear.

[5] http://labs.google.com/papers/mapreduce.html
[6] http://en.wikipedia.org/wiki/MapReduce
[7] http://blog.jteam.nl/2009/08/04/introduction-to-hadoop