In this exercise, we’ll simulate Map Reduce on a 5-node cluster.

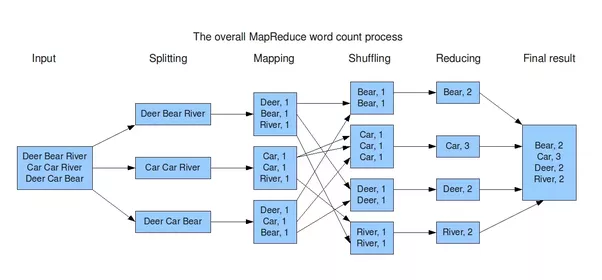
Our task is to produce an output file detailing the word count for the words in an extract of Shakespeare’s Macbeth.

Let us assume that our data is stored on HDFS however it is too large to be processed on a single machine.

In MapReduce we have three well-defined steps: Map, Shuffle, Reduce.

**Background information**

* Input – First we’ll break up the data into smaller chunks so commodity machines can handle the computational load.
* Map – In this step each node will create key value pairs with a count of one for each word e.g (the, 1)
* Shuffle – Once all the map processes are complete, all these intermediate files are shuffled across the cluster so the pairs with the same key end up on the same machine.
* Reduce – In the final step, the values for each key are combined. Here we just need to sum all the ones for a given key.
* Output – Finally counts are written to output files.



**Input:**

Is this a dagger which I see before me,

The handle toward my hand? Come, let me clutch thee;

I have thee not, and yet I see thee still.

Art thou not, fatal vision, sensible

To feeling as to sight? or art thou but

A dagger of the mind, a false creation,

Proceeding from the heat-oppressed brain?

I see thee yet, in form as palpable

As this which now I draw.

**Task:**

* Split up into teams of five. Each member will be a node in the cluster.
* As a team split the data into 5 parts.
* You will work through the roles Mapper, Shuffler, Reducer ultimately producing an output file with the word counts for the text.
* Each person starts off in the mapper role – taking a section of the text produce a key value pair for each word. E.g. if your section is ‘my cat is my cat’, produce five key value pairs: (my, 1), (cat, 1), (is, 1), (my, 1), (cat, 1).
* Shuffle the key value pairs into alphabetical order and distribute it amongst yourselves so the pairs with the same key end up on the same machine.
* Each machine will now combine the values for each key by adding them.
* Finally write out all the final word counts.