Map/Reduce using R, Hadoop and an EMR cluster

@anettebgo
@sivmhollup

Agenda

What do all those words mean?

Amazon EMR example

More about R

Map/Reduce with R - Example 1

Map/Reduce with R - Example 2

Summary

R

A programming language

Hadoop

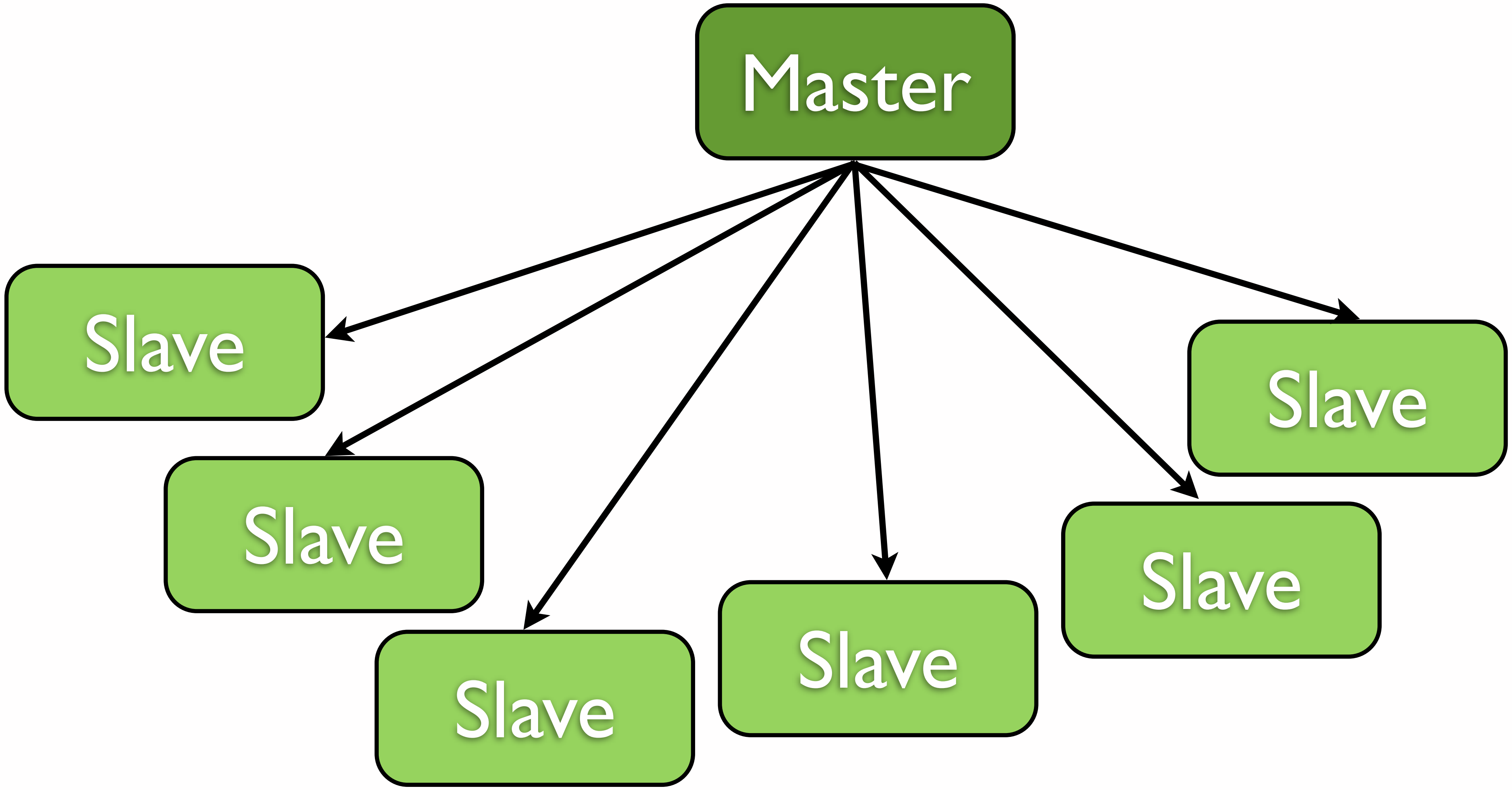
A framework to run map/reduce algorithms

EMR

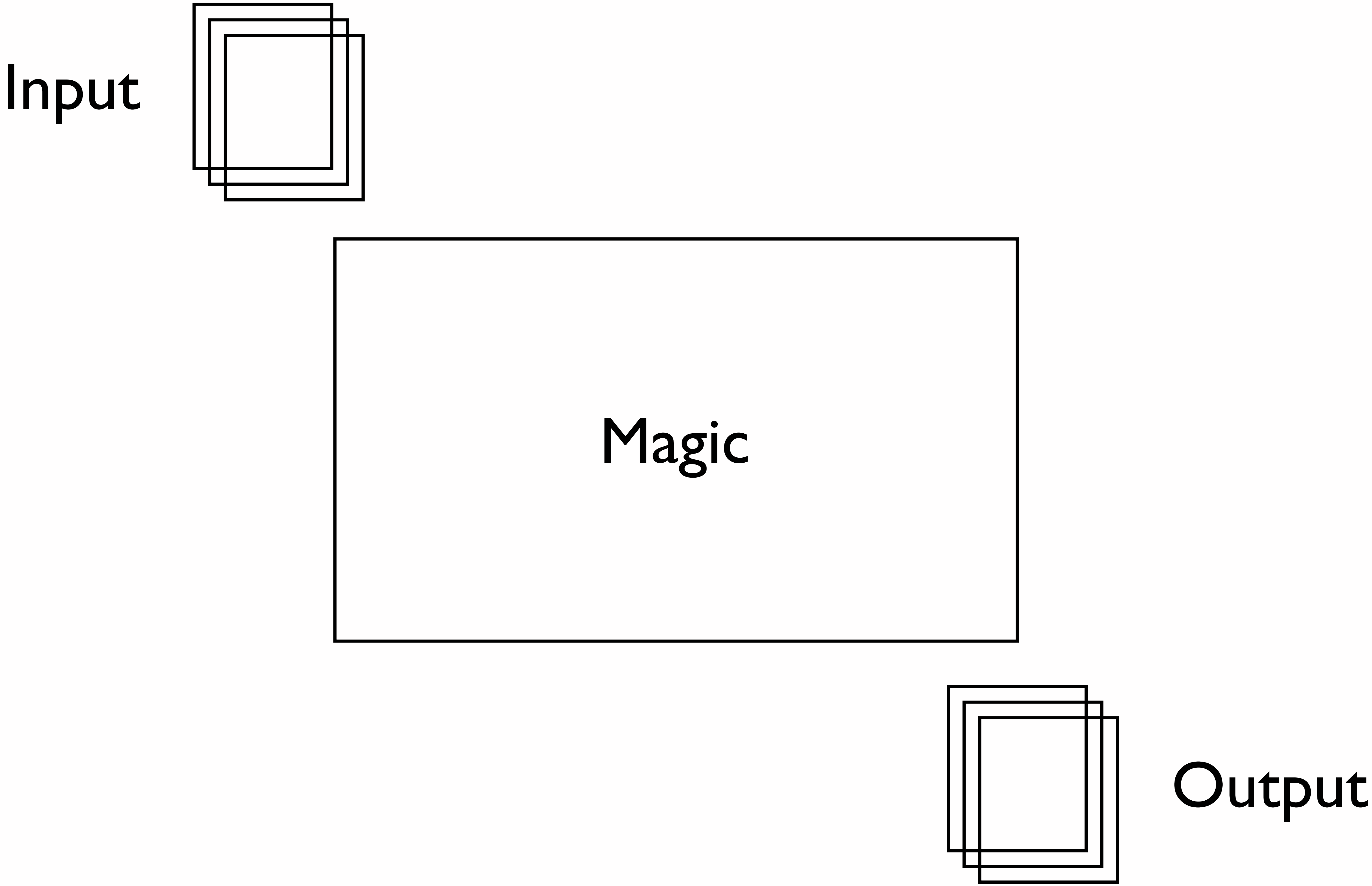
Elastic Map/Reduce

A service from Amazon to easily set up and tear down clusters with the Hadoop framework on them.

Cluster



Map/Reduce

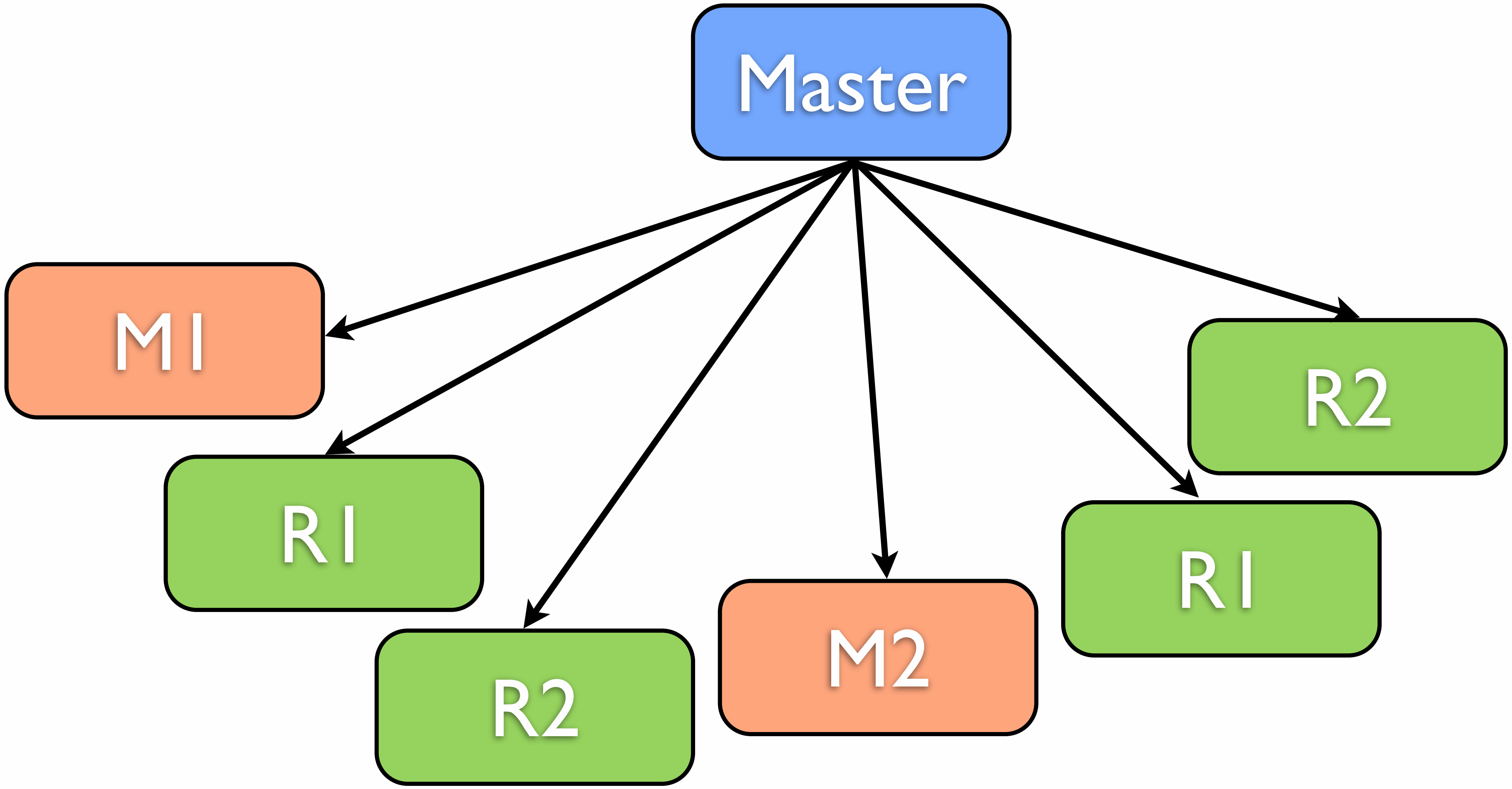


Input

Magic

Output

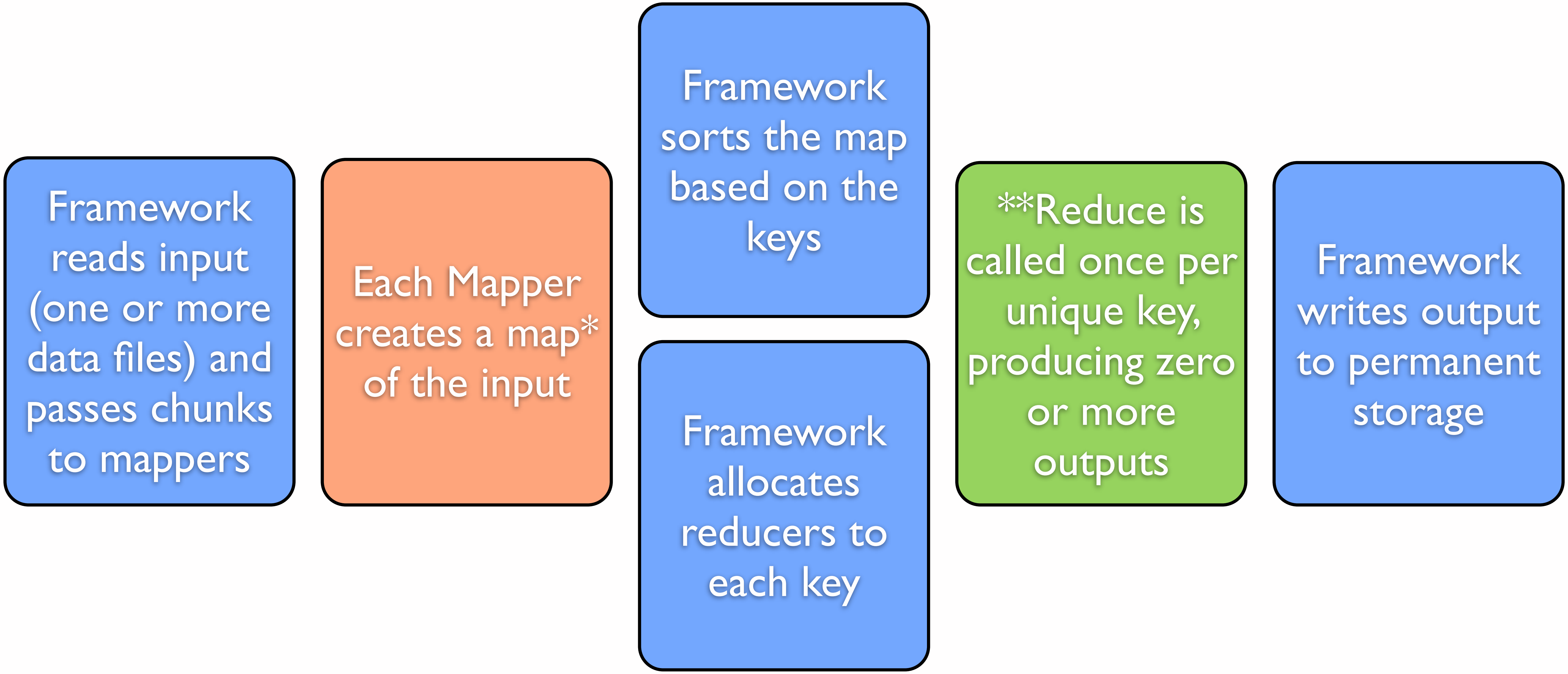
Map/Reduce Cluster



Map/Reduce

- an *input reader*
- a ***Map*** function
- a *partition* function
- a *compare* function
- a ***Reduce*** function
- an *output writer*

Map/Reduce



Framework reads input (one or more data files) and passes chunks to mappers

Each Mapper creates a map* of the input

Framework sorts the map based on the keys

Framework allocates reducers to each key

**Reduce is called once per unique key, producing zero or more outputs

Framework writes output to permanent storage

* Map = set of key/value pairs

** Reduce = collapse map into result

Word Count Example

Word Count Example

Input

```
bla bla bla
and so
forth and
more
```

Word Count Example

Input

```
bla bla bla
and so
forth and
more
```

Mapped

```
bla    1
bla    1
bla    1
and    1
so     1
forth  1
and    1
more   1
```

Word Count Example

Input

```
bla bla bla
and so
forth and
more
```

Mapped

```
bla     1
bla     1
bla     1
and     1
so      1
forth   1
and     1
more    1
```

Sorted

```
and     1
and     1
bla     1
bla     1
bla     1
so      1
forth   1
more    1
```

Word Count Example

Input

```
bla bla bla
and so
forth and
more
```

Mapped

```
bla     1
bla     1
bla     1
and     1
so      1
forth   1
and     1
more    1
```

Sorted

```
and     1
and     1
bla     1
bla     1
bla     1
so      1
forth   1
more    1
```

Reduced

```
and     2
bla     3
forth   1
more    1
so      1
```

Word Count Example

```
cat data.txt | ./mapper.R | sort | ./reducer.R
```

Input

```
bla bla bla
and so
forth and
more
```

Mapped

```
bla     1
bla     1
bla     1
and     1
so      1
forth   1
and     1
more    1
```

Sorted

```
and     1
and     1
bla     1
bla     1
bla     1
so      1
forth   1
more 1
```

Reduced

```
and     2
bla     3
forth   1
more    1
so      1
```

Testing the account

- Log into your amazon account and go to AWS management console (top right)
- Click Services (top left), then S3 and create a bucket (region doesn't matter)
- **Write down** the name of your bucket.

Testing the account

- Click on your name in the top right corner > security credentials > **access keys**. If there isn't at least one access key here, create one.
- You do not need to save the key file.

Testing the account

- Click Service (top left) then Elastic Map Reduce
- Click Create Cluster, then Configure sample application (top right)
- Choose word count, and make sure you fill in the Logging and Ouput locations with your bucket name!
- Click Create Cluster at the bottom

This may take a while... let's hear more about R.

R introduction

How do you work in R?

Command line interpreter + your fav editor

R app (Windows and others)

RStudio: an IDE for R

Variable assignment

```
aVar = 23
```

```
aVar <- 23
```

Don’t mix them in the same script!

Everything is a vector

single = 45

multitple = c(2, 3, 4, 5)

single is a vector of 1 element, multiple has 4.

Vectors are 1-indexed!

multiple[2] gives second element in multiple

Getting help: ?

In R editor:

?mean gives built-in documentation of mean() function

Google it!

Basic operations on vectors

Scaling: multiple * 4

Summing: sum(multiple), sum(single, multiple)

Multiplication: multiple * multiple (gotcha: different vector lengths work)

Data into R

Direct from command line: | (see run.sh in examples)

Read from file: read.table("filename.csv")

Pretty plots

```
plot(vector)

lines(vector)

pdf(“myPrettyPlot.pdf”)

plot(vector)

dev.off()
```

functions

```
myname <- function (parameters) {
        important_stuff = do_the_magic(paramters)
        return (stuff_to_return)
}
```

The lambda:
applying a function over a vector

sapply(), lapply(), apply()

sapply(cats, FUN=function(kitty){paste(kitty, 'cat')})

R documentation

http://www.johndcook.com/R_language_for_programmers.html

Check results from first run

Check results

- Go back to bucket
- S3 > Your Bucket > wordcount > output

Example 1 - Locally

- Check out the code from git
- Run example one locally using the run.sh script
- You may need to install some missing packages:

```
> R CMD INSTALL /R_packages/HadoopStreaming_0.2.tar.gz

> R CMD INSTALL /R_packages/getopt_1.17.tar.gz
```

Example 1 - AWS

- Edit bootstrapR.sh with the name of your bucket, then upload it to the bucket
- Create an **example1 folder** and upload the mapper.R reducer.R and data.txt here
- Create a folder called **R_packages** and upload content of R_Packages here

Running the example

- EMR > Create cluster
- Cluster Configuration > give a location in the bucket for logging
- Bootstrap Actions > Custom action. Give the bootstrapR.sh location
- Under steps, add a new Streaming step. Give it the **fully qualified location** (s3://mybucket/example1/myfile) of mapper.R, reducer.R, data.txt, and a folder name where you want the result to show up. Click Create!

All done?

Example 2

Do people try to trick their way around the toll-free import limit (currently 200 NOK) by having goods from a single larger purchase sent in multiple parcels?

Example 2 - bank data

```
id date       shop.name       currency amount.paid
1  2013-09-01 petters verktøy GBP      738
2  2013-09-01 petters bøker   GBP      119
3  2013-09-01 amazon bøker    NOK      844
```

Example 2 - post data

```
id  order.date  ...  sender.country ... declared.value
1   2013-09-01  ...  Norge          ... 347
2   2013-09-01  ...  Norge          ... 211
```

Example 2

Do people try to trick their way around the toll-free import limit (currently 200 NOK) by having goods from a single larger purchase sent in multiple parcels?

Example 2

- How many purchases do norwegians do from shops outside Norway?
- Does it match the number of parcels they get delivered?

Example 2 - go hack (locally)

- Check out the example2 folder
- Create a reducer_bank.R to run with the mapper. It should produce output of the following format:

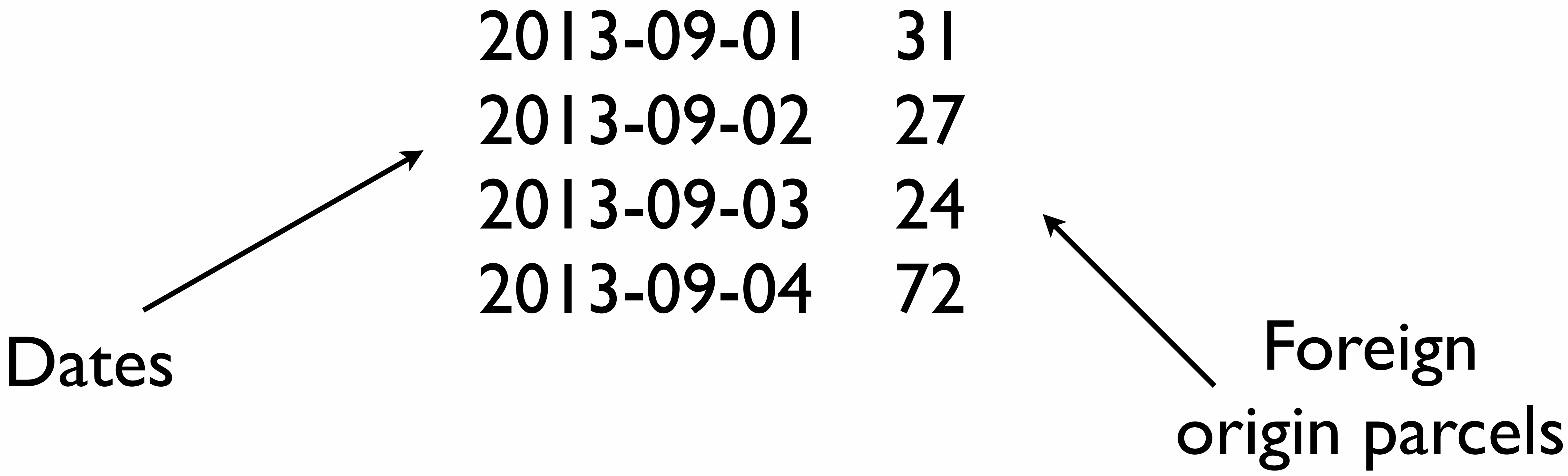
```
2013-09-01    32
2013-09-02    27
2013-09-03    26
2013-09-04    42
```

Dates

Foreign transactions

Example 2 - go hack (locally)

- Create the mapper_post.R and reducer_post.R for the post data. It should also end up with a result with this format:



```
2013-09-01    31
2013-09-02    27
2013-09-03    24
2013-09-04    72
```

Dates

Foreign origin parcels

Example 2 - go to AWS

- Run the map/reduce jobs you have created on AWS
- Download the results for plotting!

Example 2 - plotting

- Have a look at the file plots.R
- Modify to take in your results and run in the RStudio console

Extensions

- Do example two for domestic transactions and parcels. Is there a difference?
- Wordcount to ignore case & punctuation
- How many parcels are delivered in total from Australia?
- What is the amount in NOK of things paid for in Germany?
- Wordcount - do any words occur next to each other more often than others?
- Total amount purchased vs. total amount

Summary

Summary

Map/Reduce is for BIG DATA

Hadoop can be used with a range of languages

Amazon console is rubbish. Use boto!

Summary

Big Data is Dirty Data

Interpretation is important

Datasparsemkeit

Questions?