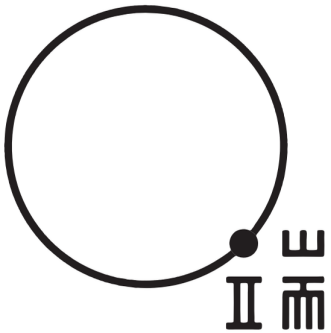
Hands-on Hadoop

(Hands-on Massive Data Processing Platform Series)

Pili Hu

Initium Media 端傳媒

June 26, 2015



Preparation

- Download Install
  - VirtualBox: https://www.virtualbox.org/
  - Vagrant: https://www.vagrantup.com/
- Download our Vagrantfile
- Follow instructions in this project

  https://github.com/initiummedia/hkosc2015-workshop

[Trouble shoot]
Hadoop DFS not running?

type `jps`
can you see `namenode` and `datanode`?

If not:

./bin/hdfs namenode -format

./sbin/hadoop-daemon.sh start namenode

./sbin/hadoop-daemon.sh start datanode

This Workshop Is About

- Hands-on three massive data processing platforms:
  - Hadoop
  - Spark
  - GraphLab
- Get the basic programming concept of the framework
- Get a feel of command-line/ shell of the framework

This Workshop Is NOT About

- How to install/ configure a cluster
- Rigorous performance evaluation
- Mathematical principle behind the frameworks
- Architecture of the platform from an implementation perspective

Expected Take-aways

- Demythify “Big Data Platforms”
- Benefit of framework:

  Dealing with small == dealing with big

Show-off to your friends:

Yeah, I got my hands-on XXX!

Choices of Platforms

- Hadoop: 1st widely adopted platform by industry; popularised MapReduce
- Spark: A lot optimisation over Hadoop to reach hundreds times acceleration; current de facto standard
- GraphLab: Cutting edge framework to implement Machine Learning algorithms; New programming concept -- Vertex Program
- ~~Storm: widely adopted Streaming platform~~

Agenda of the Workshop Series

June 26 (Fri) afternoon:

- Basic introduction of this workshop
- Hands-on Hadoop
- Hands-on Spark

June 27 (Sat) afternoon:

- Hands-on GraphLab
- Comparison between three platforms

Agenda of the Hadoop Hands-on Session

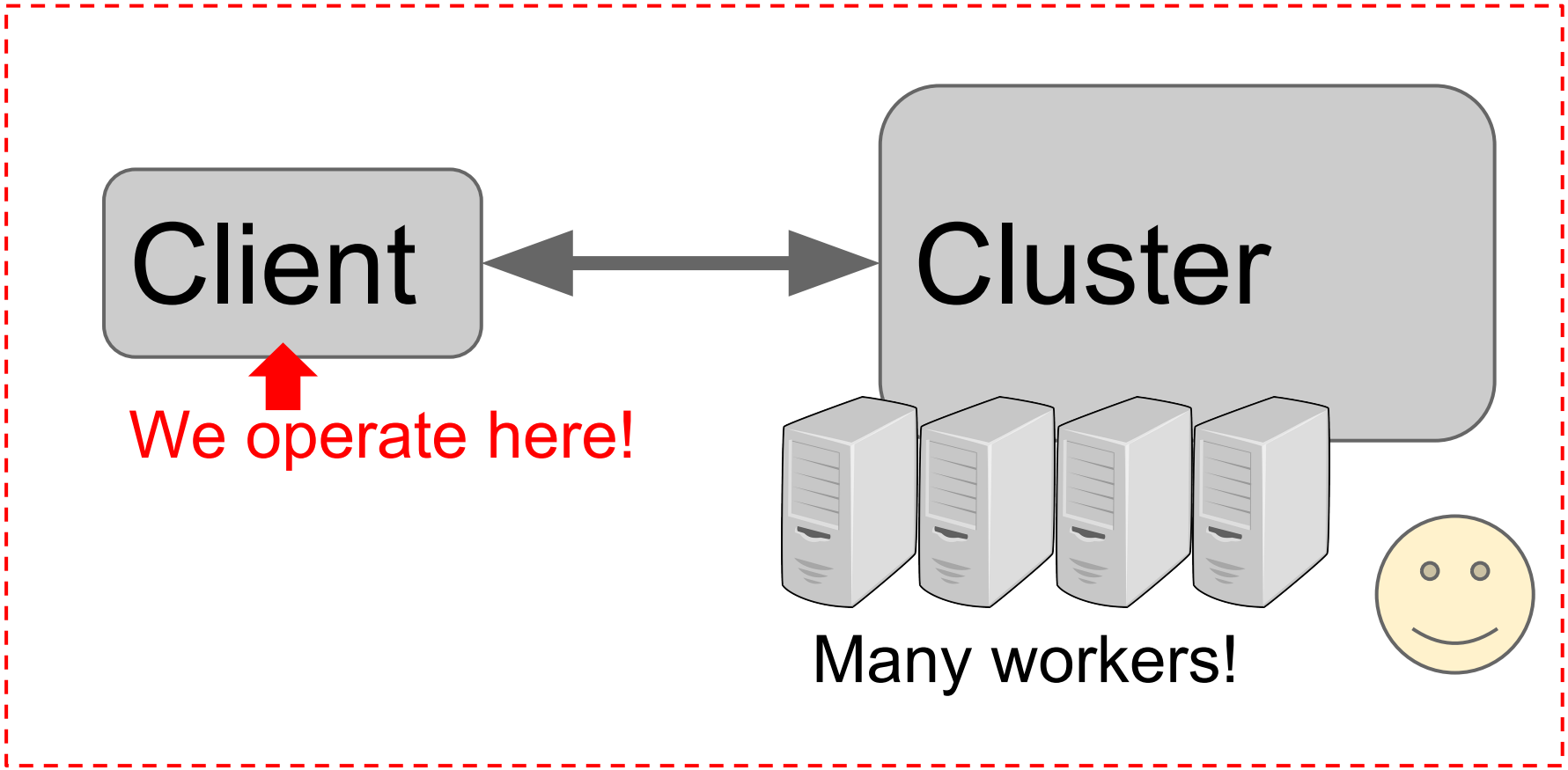
- Overview
- Project Layout
- Manipulating HDFS
- Run Examples
- 1st MapReduce Program (Python)

History

- 2003/2004 -- Google made GFS/ MapReduce
- 2006 -- Yahoo made Hadoop based on Google paper
- 2008 -- Apache took over Hadoop
- 2011 -- Hadoop v1.0
- 2013 -- Hadoop v2.0

Hadoop from a practical view

The whole thing on a single machine!

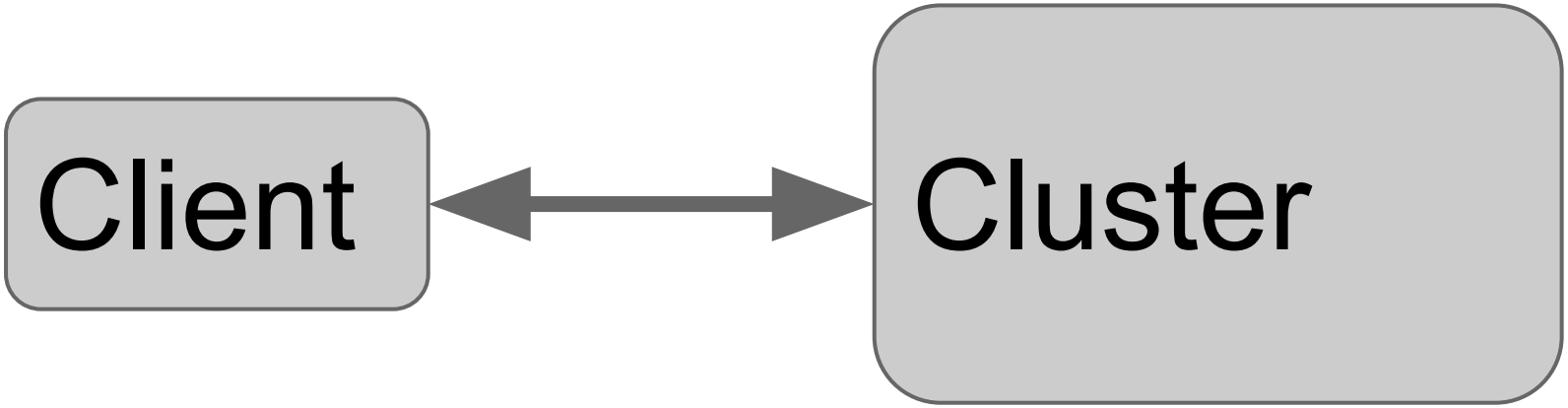


Hadoop Package Layout

- bin: executables
- sbin: administration executables
- etc: configurations
  - core-site.xml: HDFS
  - hdfs-site.xml: HDFS
  - mapred-site.xml: MapReduce
- src: sources
- logs:

HDFS

Hadoop Distributed File System

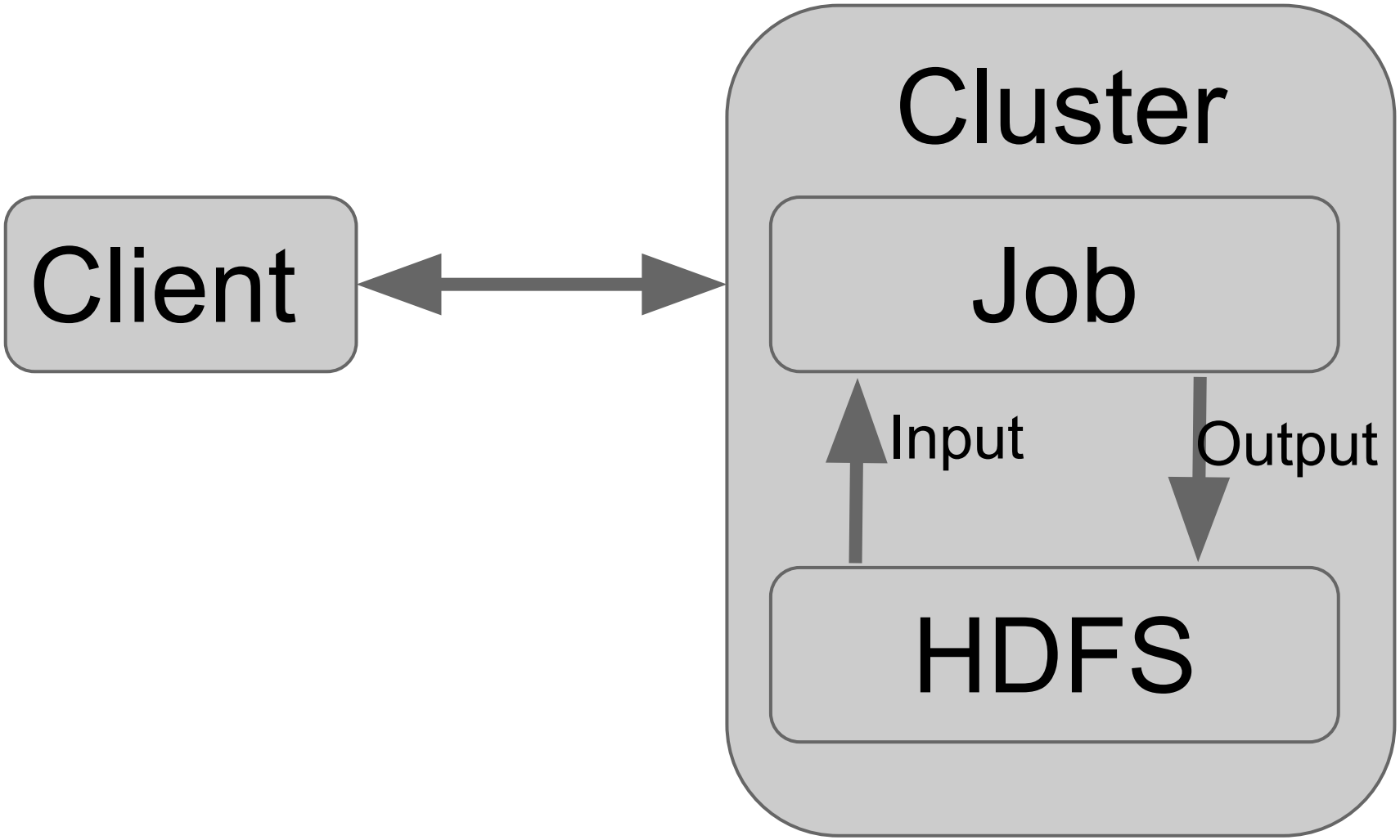


HDFS Operation

./bin/hadoop dfs XXXX

- -ls
- -rm
- -mkdir
- -put
- -cat
- -get
- -rmr

Hadoop Job & HDFS



Examples

vagrant@master:~/hadoop-2.7.0$ ./bin/hadoop dfs -put README.txt /

vagrant@master:~/hadoop-2.7.0$ ./bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.0.jar grep /README.txt /output 'oft'

vagrant@master:~/hadoop-2.7.0$ ./bin/hadoop dfs -cat /output/part-r-00000

vagrant@master:~/hadoop-2.7.0$ grep oft README.txt -o | wc -l

First MapReduce

MapReduce?

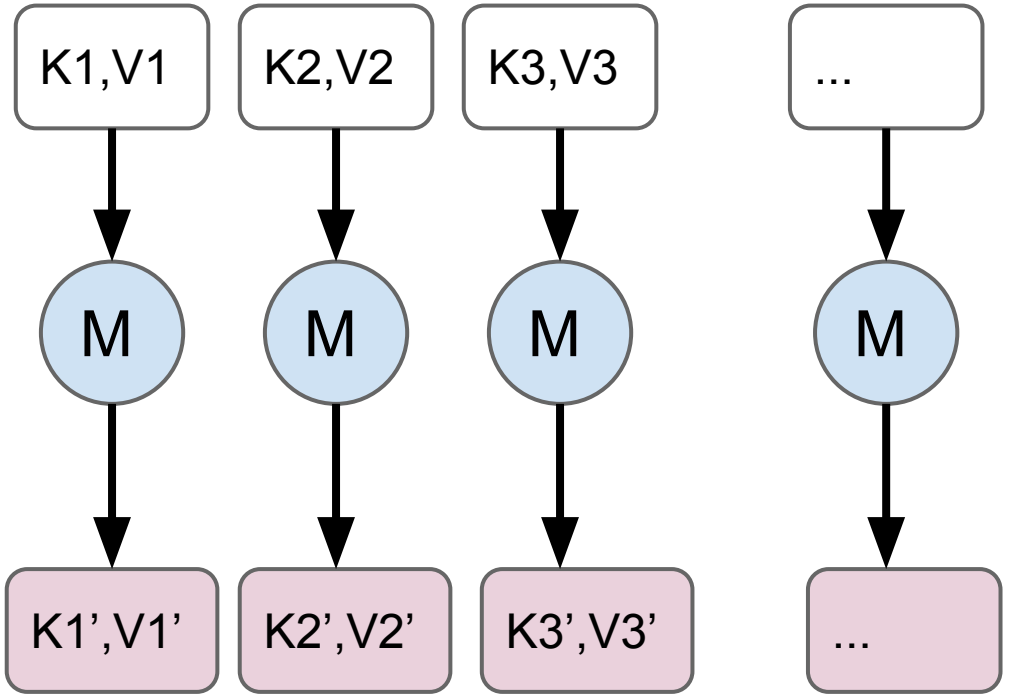
- A programming framework
- Not new stuff
- Popularised by Google's paper and Hadoop

Very like the map() and reduce() in Python2

=D

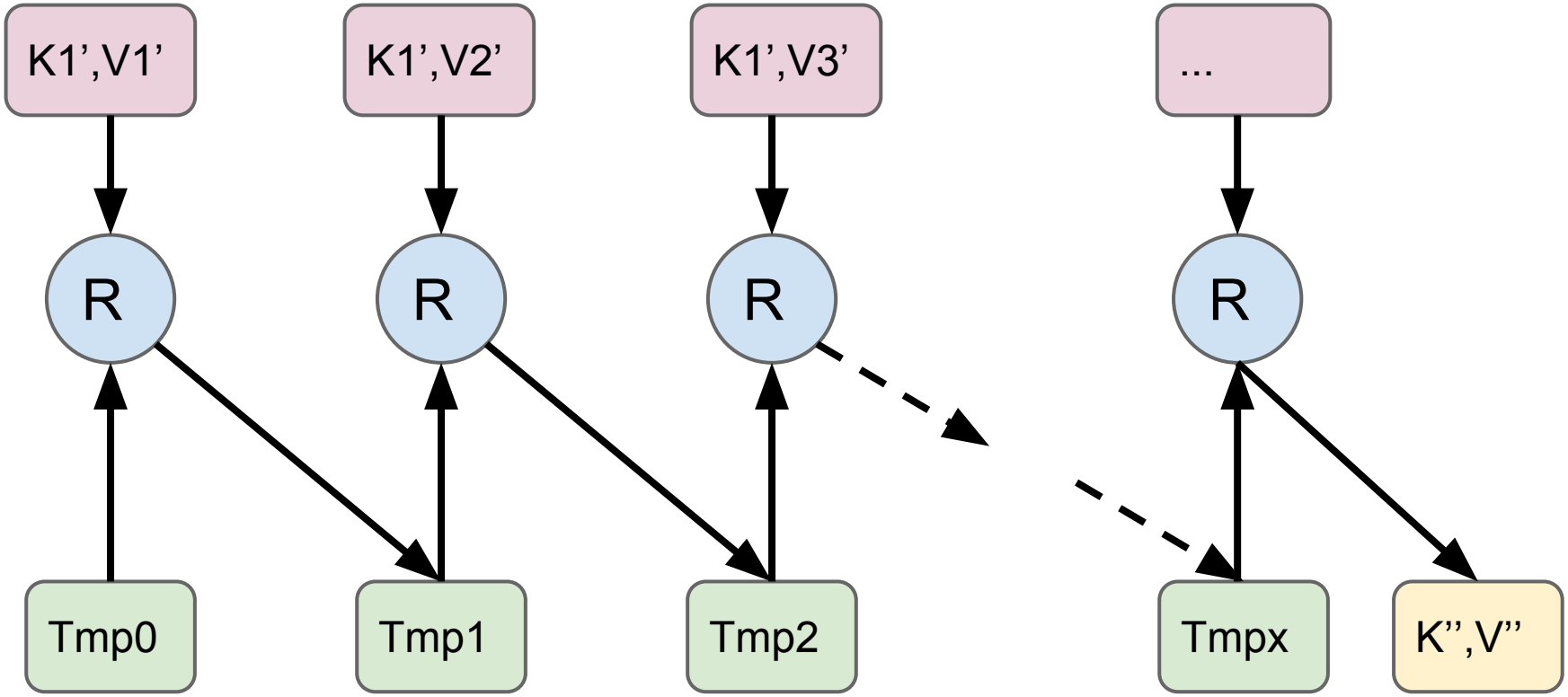
https://github.com/initiummedia/hkosc2015-workshop/tree/master/hadoop-python-examples/wordcount

Mapper



Example: scalar-product, M(x) = x * 2
[1, 2, 3] -> [M(1), M(2), M(3)] = [2, 4, 6]

Reducer



Example: Sum

R(0, [1, 2, 3]) = R(Sum(0, 1), [2, 3]) = … = Sum(Sum((Sum(0, 1), 2), 3)

Word Count Problem

Hong Kong Open Source Conference is best open source conference in the world!

→

hong 1, kong 1, open 2, source 2, conference 2, is 1, best 1, in 1, the 1, world 1

Mapper

```
#!/usr/bin/env python

import sys

for line in sys.stdin:
    for word in line.split():
        print '%s\t%s' % (word, 1)
```

Reducer

```
#!/usr/bin/env python

import sys

cur_key = None
cur_count = 0

for line in sys.stdin:
    key, value = line.split()
    if key == cur_key:
        cur_count += int(value)
    else:
        if cur_key:
            print '%s\t%s' % (cur_key, cur_count)
        cur_key = key
        cur_count = int(value)

print '%s\t%s' % (cur_key, cur_count)
```

Command-line

./bin/hdfs dfs -put README.txt /

./bin/hadoop jar ./share/hadoop/tools/lib/hadoop-streaming-2.7.0.jar -input /README.txt -output /wordcount-output -mapper mapper.py -reducer reducer.py -file mapper.py -file reducer.py

./bin/hadoop jar ./share/hadoop/tools/lib/hadoop-streaming-2.7.0.jar -input /README.txt -output /wordcount-output2 -mapper mapper.py -reducer cat -file mapper.py

Further

- Check the intermediate result of MapReduce
- How to get the output sorted by frequency

Thanks & Q/A

Contact me:

http://hupili.net

WE ARE HIRING