

Lab 13. Hadoop MapReduce

Chan-Hyun YOUN

Dept of Electrical Engineering, KAIST

LAB 13-1)

HADOOP INSTALLATION

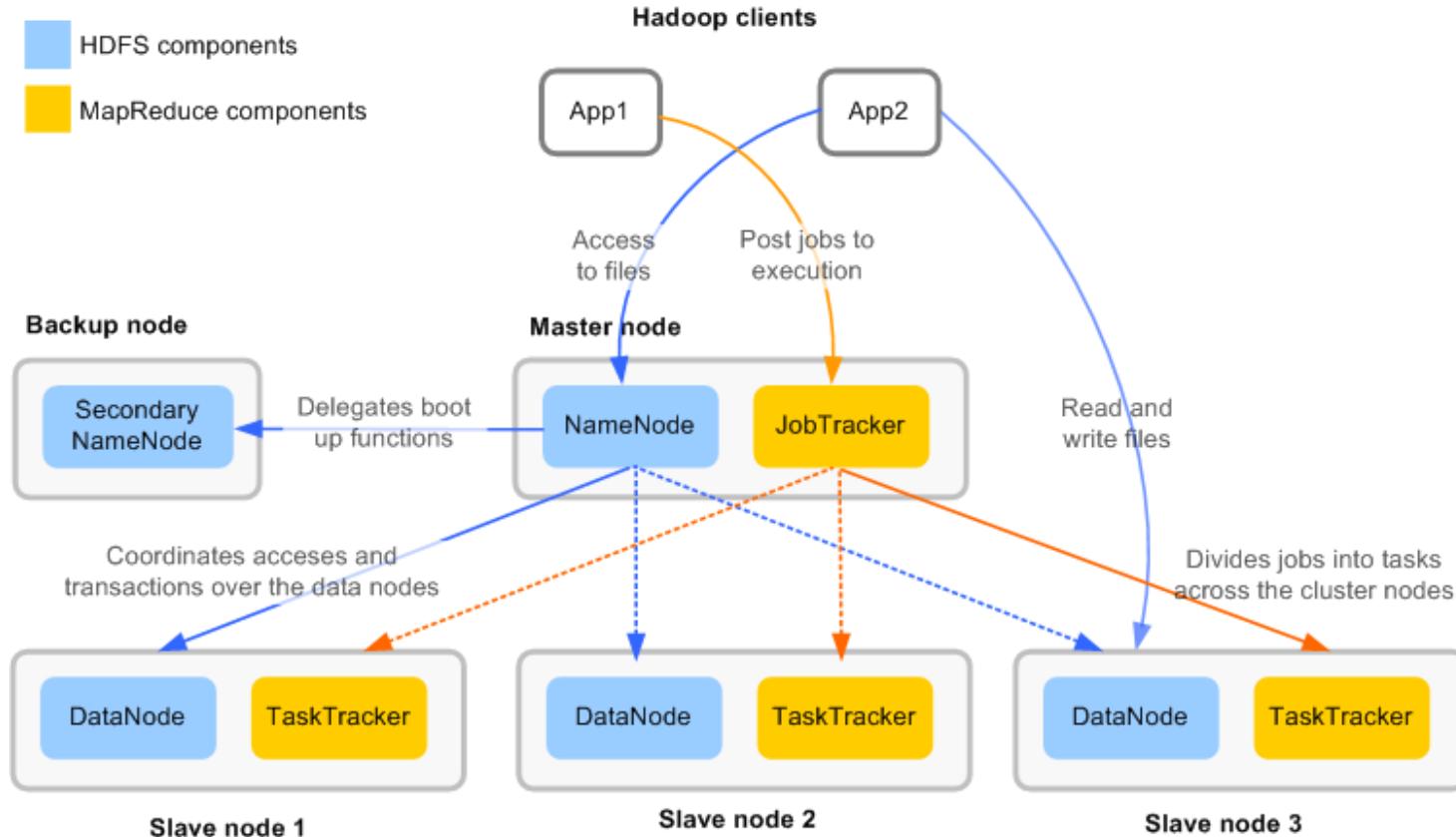
Entire System Overview



For more information visit me at
www.hadooper.blogspot.com

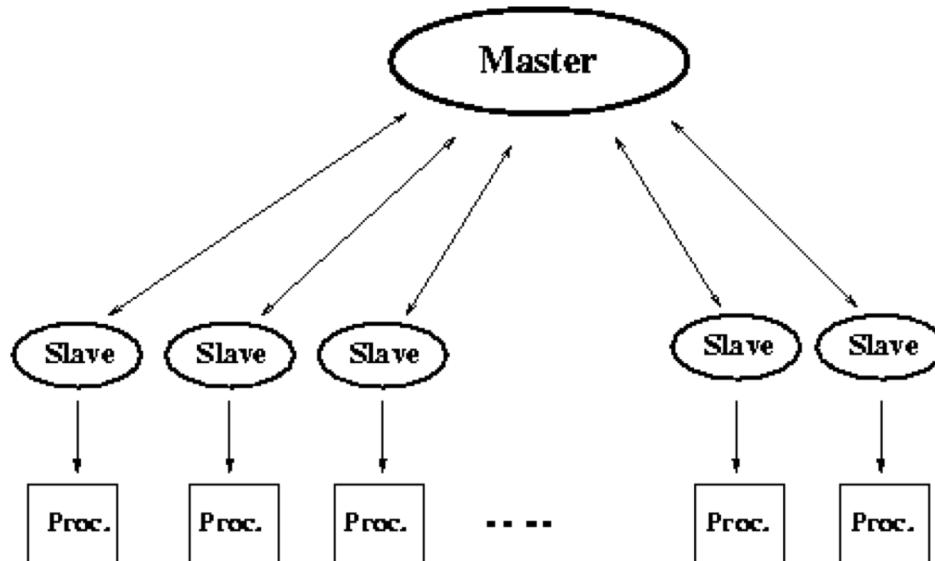
Fully-Distributed Mode

Hadoop base platform brief



Hadoop Architecture

- Master-Slave Model
 - In the master/slave (sometimes called boss/worker) model, a master entity receives one or more requests, then creates slave entities to execute them. Typically, the master controls the number of slaves and what each slave does.
 - The job of the master is to divide the work among the available slaves, keeping each of them busy without using too much communication
 - In slave, the most important aspect we have to deal with is the job handling



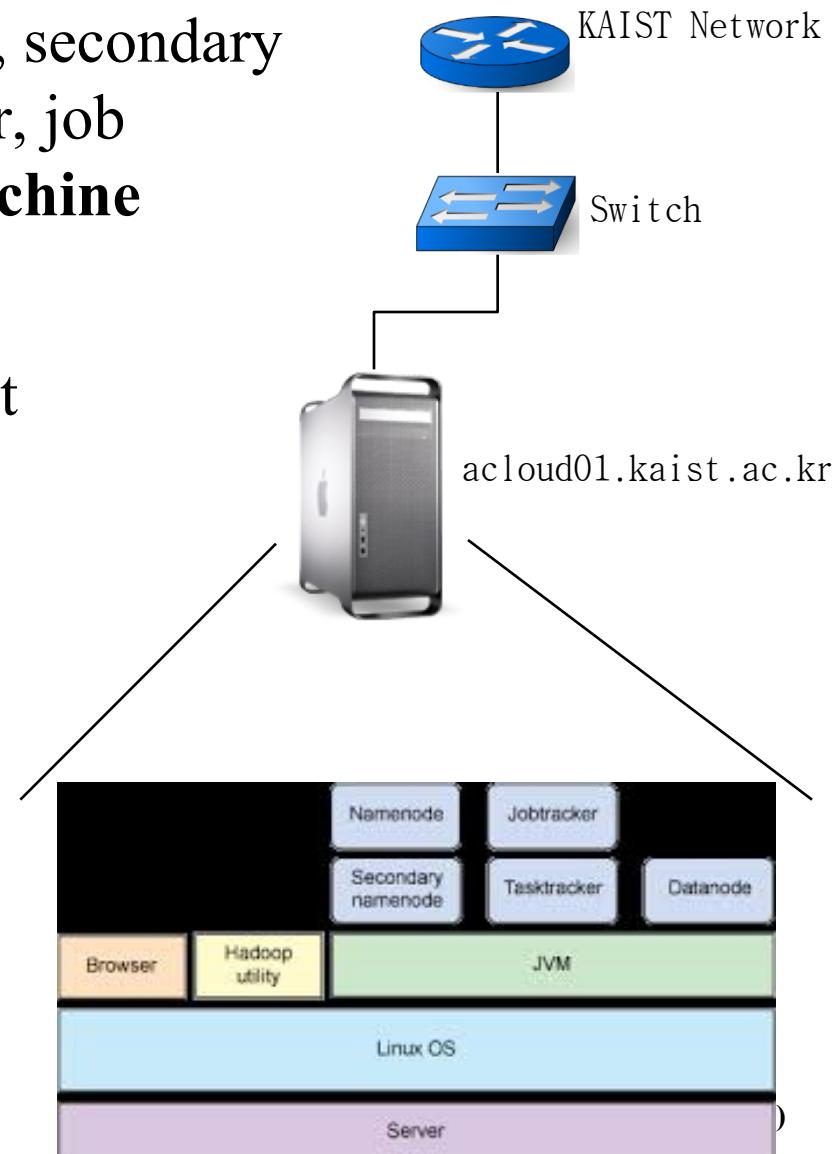
Hadoop Installation

- Supported Platforms
 - GNU/Linux as a development and production platform.
 - Win32 is supported as a *development platform*.
- Required Software
 - **JavaTM 1.6.x**, preferably from Sun, must be installed.
 - **SSH** must be installed and sshd must be running to use the Hadoop
- Supported modes
 - **Local (Standalone) Mode**
 - Used for debugging
 - **Pseudo-Distributed Mode**
 - All Hadoop process (namenode, secondary namenode, datanode, task tracker, job tracker) runs in **single linux machine**
 - **Fully-Distributed Mode**

Pseudo-Distributed Mode Installation

- **All Hadoop process (namenode, secondary namenode, datanode, task tracker, job tracker) runs in single linux machine**

1. Prepare linux machine with root authentication
2. Java Installation
3. Hadoop Download
4. SSH configuration
5. Hadoop Configuration



Pseudo-Distributed Mode Installation

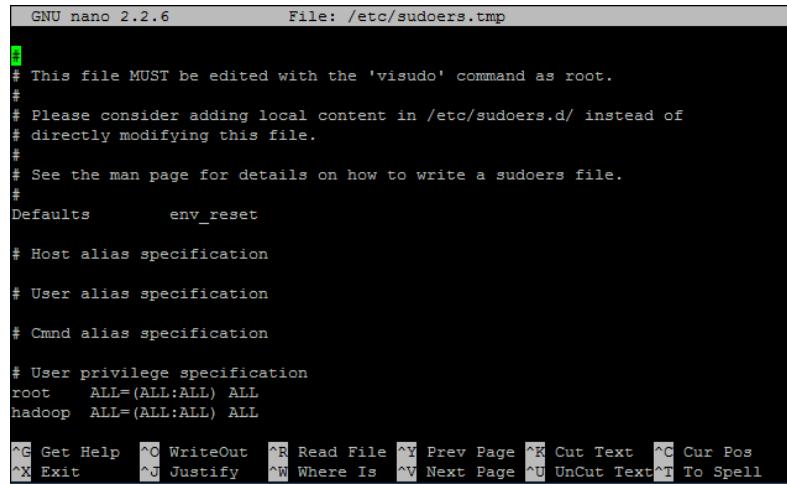
1. Prepare linux machine with root authentication

- Prepare user account ‘Hadoop’ with root authentication (some modifications are needed in /etc/sudoers file)

```
# User privilege specification
root      ALL=(ALL:ALL)  ALL
hadoop   ALL=(ALL:ALL)  ALL
```

- To modify sudoers file, you might need root authentication. Use command ‘visudo’

→ \$ sudo visudo



```
GNU nano 2.2.6          File: /etc/sudoers.tmp

# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root      ALL=(ALL:ALL)  ALL
hadoop   ALL=(ALL:ALL)  ALL
```

The terminal window shows the command '^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos ^X Exit ^J Justify ^W Where Is ^V Next Page ^U Uncut Text ^T To Spell' at the bottom.

Pseudo-Distributed Mode Installation

8

- For Pseudo-Distributed mode installation, we will use '**localhost**' as hostname (Hostname is important in hadoop system. Hadoop manages every nodes with hostname like MPI)
- Can modify the local machine's hostname in **/etc/hostname** and can modify the remote machine's hostname in **/etc/hosts**

```
hadoop@acloud01:~$ cat /etc/hostname  
acloud01
```

```
hadoop@acloud01:~$ cat /etc/hosts  
127.0.0.1      localhost  
  
143.248.152.16 accloud01.kaist.ac.kr      accloud01  
143.248.152.17 accloud02.kaist.ac.kr      accloud02  
  
143.248.6.161   anclabvm1.kaist.ac.kr    avm1  
  
# The following lines are desirable for IPv6 capable hosts  
::1      ip6-localhost ip6-loopback  
fe00::0  ip6-localnet  
ff00::0  ip6-mcastprefix  
ff02::1  ip6-allnodes  
ff02::2  ip6-allrouters
```

Pseudo-Distributed Mode

Installation

- After the modification of hostname, you should restart your network service
→ \$ sudo /etc/init.d/networking restart

```
hadoop@acloud01:~$ sudo /etc/init.d/networking restart
 * Running /etc/init.d/networking restart is deprecated because it may not enable again some interfaces
 * Reconfiguring network interfaces...
 * Disconnecting iSCSI targets
   ...done.
 * Stopping iSCSI initiator service
   ...done.
 * Disconnecting iSCSI targets
   ...done.
 * Stopping iSCSI initiator service
   ...done.

 * Starting iSCSI initiator service iscsid
   ...done.
 * Setting up iSCSI targets
   ...done.

ssh stop/waiting
ssh start/running, process 25566
 * Setting up iSCSI targets
   ...done.

ssh stop/waiting
ssh start/running, process 25649
```

[OK]

Pseudo-Distributed Mode

Installation

2. Java Installation

- Java which version is above the 1.6 is suggested

```
$ sudo apt-get update  
$ sudo apt-get install openjdk-6-jdk
```

...

(installation process)

...

```
$ java -version
```

```
java version "1.6.0_27"  
OpenJDK Runtime Environment (IcedTea6 1.12.3) (6b27-1.12.3-0ubuntu1~12.04.1)  
OpenJDK 64-Bit Server VM (build 20.0-b12, mixed mode)
```

Java Location

→ /usr/lib/jvm/java-6-openjdk-amd64

Pseudo-Distributed Mode Installation

3. Hadoop Download

- In this lab, we will use Hadoop 1.0.0
- Download Hadoop at
<http://archive.apache.org/dist/hadoop/core/>

www.apache.org/dyn/closer.cgi/hadoop/common/



The Apache Software Foundation

Home > Dyn

Apache Download Mirrors

We suggest the following mirror site for your download:

<http://apache.tt.co.kr/hadoop/common/>

Other mirror sites are suggested below. Please use the backup mirrors only to download PGP and MD5 signatures to verify your downloads or if no other mirrors are working.

HTTP

<http://apache.tt.co.kr/hadoop/common/>

<http://apache.mirror.cdnetworks.com/hadoop/common/>

<http://mirror.apache-kr.org/hadoop/common/>

Backup Sites

Please use the backup mirrors only to download PGP and MD5 signatures to verify your downloads or if no other mirrors are working.

<http://www.eu.apache.org/dist/hadoop/common/>

<http://www.us.apache.org/dist/hadoop/common/>

[The full listing of mirror sites](#) is also available.

Becoming a mirror

The procedure for setting up new mirrors is described in [How to become a mirror](#).

← → ⌂ archive.apache.org/dist/hadoop/core/

archive.apache.org

This site contains the historical archive of old software releases.

For current releases, please visit the [mirrors](#).

Name	Last modified	Size	Description
Parent Directory		-	
alpha/	2013-04-12 21:53	-	
beta/	2013-01-31 22:42	-	
hadoop-0.10.1/	2008-01-22 23:12	-	
hadoop-0.11.2/	2008-01-22 23:12	-	
hadoop-0.12.0/	2008-01-22 23:12	-	
hadoop-0.12.1/	2008-01-22 23:12	-	
hadoop-0.12.2/	2008-01-22 23:12	-	
hadoop-0.12.3/	2008-01-22 23:12	-	
hadoop-0.13.0/	2008-01-22 23:12	-	
hadoop-0.13.1/	2008-01-22 23:12	-	
hadoop-0.14.0/	2008-01-22 23:44	-	
hadoop-0.14.1/	2008-01-22 23:12	-	

Installation

- Download with linux command

```
hadoop@acloud01:~$ wget http://archive.apache.org/dist/hadoop/core/hadoop-1.0.0/
hadoop-1.0.0.tar.gz
--2013-05-13 17:30:27--  http://archive.apache.org/dist/hadoop/core/hadoop-1.0.0
/hadoop-1.0.0.tar.gz
Resolving archive.apache.org (archive.apache.org)... 140.211.11.131, 192.87.106.
229, 2001:610:1:80bc:192:87:106:229
Connecting to archive.apache.org (archive.apache.org)|140.211.11.131|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 59468784 (57M) [application/x-gzip]
Saving to: `hadoop-1.0.0.tar.gz'

99% [=====>] 59,395,576  135K/s  eta 2s
```

- Untar the download file
\$ tar -zxvf hadoop-1.0.0.tar.gz
- Move the directory into /opt
\$ mv hadoop-1.0.0 /opt

```
hadoop@acloud01:/opt/hadoop-1.0.0$ ls
bin          docs           ivy        NOTICE.txt
build.xml    hadoop-ant-1.0.0.jar  ivy.xml    README.txt
c++          hadoop-core-1.0.0.jar lib        sbin
CHANGES.txt  hadoop-examples-1.0.0.jar libexec   share
conf         hadoop-test-1.0.0.jar  LICENSE.txt src
contrib      hadoop-tools-1.0.0.jar logs      webapps
```

Pseudo-Distributed Mode Installation

4. SSH Configuration (Refer the Lab 11)

- Can access to the host named ‘localhost’ without password

```
$ ssh-keygen -t rsa -P ""
```

```
$ cd ~/.ssh
```

```
$ cat id_rsa.pub >> authorized_keys
```

```
$ cat authorized_keys
```

```
$ ssh localhost
```

Pseudo-Distributed Mode Installation

- Can access to the host named ‘localhost’ without password

```
hadoop@acloud01:~/.ssh$ ssh localhost
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.2.0-40-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

48 packages can be updated.
21 updates are security updates.

Last login: Mon May 13 16:48:33 2013 from localhost
```

Pseudo-Distributed Mode

Installation

15

5. Hadoop Configuration

- The \$HADOOP_INSTALL/hadoop/conf directory contains some configuration files for Hadoop.

```
hadoop@accloud01:/opt/hadoop-1.0.0/conf$ ls
capacity-scheduler.xml      hadoop-policy.xml      slaves
configuration.xsl            hdfs-site.xml        ssl-client.xml.example
core-site.xml                log4j.properties    ssl-server.xml.example
fair-scheduler.xml           mapred-queue-acls.xml taskcontroller.cfg
hadoop-env.sh                mapred-site.xml
hadoop-metrics2.properties   masters
```

- hadoop-env.sh:** This file contains some environment variable settings used by Hadoop. You can use these to affect some aspects of Hadoop daemon behavior, such as where log files are stored, the maximum amount of heap used etc. The only variable you should need to change in this file is JAVA_HOME, which specifies the path to the Java 1.6.x installation used by Hadoop.

Pseudo-Distributed Mode Installation

- **masters:** This file specifies the host, where the Secondary NameNode will run. By default this contains the single entry localhost
- **slaves:** This file lists the hosts, one per line, where the Hadoop slave daemons (datanodes and tasktrackers) will run. By default this contains the single entry localhost
- **core-site.xml:** This file contains site specific settings for Hadoop daemons and Map/Reduce jobs.
- **mapred-site.xml:** This file contains site specific settings for the Map/Reduce daemons and jobs.
- **hdfs-site.xml:** This file contains site specific settings for HDFS daemons.

Pseudo-Distributed Mode Installation

- Hadoop Configuration for Pseudo-Distributed Mode

- masters, slaves

```
hadoop@acloud01:/opt/hadoop-1.0.0/conf$ cat masters
localhost
hadoop@acloud01:/opt/hadoop-1.0.0/conf$ cat slaves
localhost
```

- To launch all hadoop modules in ‘localhost’ node, modify **conf/masters** and **conf/slaves** file

- **hadoop-env.sh**

```
# The java implementation to use. Required.
export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-amd64/jre
```

- Specify the location of Java installation in **conf/hadoop-env.sh**

Pseudo-Distributed Mode Installation

- Hadoop Configuration for Pseudo-Distributed Mode
 - conf/core-site.xml:

```
<configuration>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/opt/hdfs</value>
  </property>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://localhost:9000</value>
  </property>
</configuration>
```

- Above configuration define the local file system directory for HDFS storage
- Also define the namenode and its port for RPC

Pseudo-Distributed Mode Installation

- Hadoop Configuration for Pseudo-Distributed Mode

- conf/hdfs-site.xml:

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
```

- Above configuration define the number of replication for failure control

- conf/mapred-site.xml:

```
<configuration>
  <property>
    <name>mapred.job.tracker</name>
    <value>localhost:9001</value>
  </property>
</configuration>
```

- Above configuration define the JobTracker node and its port for RPC

Detail options:

http://hadoop.apache.org/docs/stable/cluster_setup.html

C.H. Youn (Nov. 21, 2017)

Pseudo-Distributed Mode Launch

- HDFS (namenode) format
\$ bin/hadoop namenode -format

```
hadoop@acloud01:/opt/hadoop-1.0.0$ bin/hadoop namenode -format
Warning: $HADOOP_HOME is deprecated.

13/05/13 23:34:50 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = acloud01/143.248.152.16
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 1.0.0
STARTUP_MSG:   build = https://svn.apache.org/repos/asf/hadoop/common/branches/b
ranch-1.0 -r 1214675; compiled by 'hortonfo' on Thu Dec 15 16:36:35 UTC 2011
*****
13/05/13 23:34:50 INFO util.GSet: VM type      = 64-bit
13/05/13 23:34:50 INFO util.GSet: 2% max memory = 17.77875 MB
13/05/13 23:34:50 INFO util.GSet: capacity      = 2^21 = 2097152 entries
13/05/13 23:34:50 INFO util.GSet: recommended=2097152, actual=2097152
13/05/13 23:34:50 INFO namenode.FSNamesystem: fsOwner=hadoop
13/05/13 23:34:50 INFO namenode.FSNamesystem: supergroup=supergroup
13/05/13 23:34:50 INFO namenode.FSNamesystem: isPermissionEnabled=true
13/05/13 23:34:50 INFO namenode.FSNamesystem: dfs.block.invalidate.limit=100
13/05/13 23:34:50 INFO namenode.FSNamesystem: isAccessTokenEnabled=false accessK
eyUpdateInterval=0 min(s), accessTokenLifetime=0 min(s)
13/05/13 23:34:50 INFO namenode.NameNode: Caching file names occurring more than
10 times
13/05/13 23:34:50 INFO common.Storage: Image file of size 112 saved in 0 seconds
.
13/05/13 23:34:51 INFO common.Storage: Storage directory /opt/hdfs/dfs/name has
been successfully formatted.
13/05/13 23:34:51 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at acloud01/143.248.152.16
*****
```

Pseudo-Distributed Mode Launch

- The `$HADOOP_INSTALL/hadoop/bin` directory contains some scripts used to launch Hadoop DFS and Hadoop Map/Reduce daemons
 - **start-dfs.sh**: Starts the Hadoop DFS daemons, the namenode and datanodes. Use this before start-mapred.sh
 - **stop-dfs.sh**: Stops the Hadoop DFS daemons.
 - **start-mapred.sh**: Starts the Hadoop Map/Reduce daemons, the jobtracker and tasktrackers.

Pseudo-Distributed Mode Launch

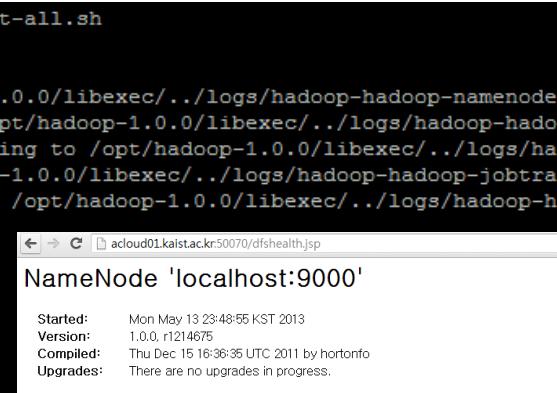
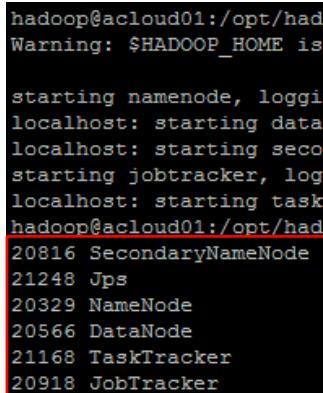
- The `$HADOOP_INSTALL/hadoop/bin` directory contains some scripts used to launch Hadoop DFS and Hadoop Map/Reduce daemons
 - **stop-mapred.sh**: Stops the Hadoop Map/Reduce daemons.
 - **start-all.sh**: Starts all Hadoop daemons, the namenode, datanodes, the jobtracker and tasktrackers. Deprecated; use `start-dfs.sh` then `start-mapred.sh`
 - **stop-all.sh**: Stops all Hadoop daemons. Deprecated; use `stop-mapred.sh` then `stop-dfs.sh`

Pseudo-Distributed Mode Launch²³

- Hadoop Launch command
 - \$ bin/start-all.sh
- Can verify the operation of hadoop with
 - jps (Java Virtual Machine Process Status Tool) command
 - <http://acloud01.kaist.ac.kr:50070>: Namenode web interface
 - <http://acloud01.kaist.ac.kr:50030>: JobTracker web interface

```
hadoop@acloud01:/opt/hadoop-1.0.0$ bin/start-all.sh
Warning: $HADOOP_HOME is deprecated.

starting namenode, logging to /opt/hadoop-1.0.0/libexec/../logs/hadoop-hadoop-namenode-acloud01.out
localhost: starting datanode, logging to /opt/hadoop-1.0.0/libexec/../logs/hadoop-hadoop-datanode-acloud01.out
localhost: starting secondarynamenode, logging to /opt/hadoop-1.0.0/libexec/../logs/hadoop-hadoop-secondarynamenode-acloud01.out
starting jobtracker, logging to /opt/hadoop-1.0.0/libexec/../logs/hadoop-hadoop-jobtracker-acloud01.out
localhost: starting tasktracker, logging to /opt/hadoop-1.0.0/libexec/../logs/hadoop-hadoop-tasktracker-acloud01.out
hadoop@acloud01:/opt/hadoop-1.0.0$ jps
20816 SecondaryNameNode
21248 Jps
20329 NameNode
20566 DataNode
21168 TaskTracker
20918 JobTracker
```



Pseudo-Distributed Mode Launch

NameNode 'localhost:9000'

Started: Mon May 13 23:48:55 KST 2013
Version: 1.0.0, r1214675
Compiled: Thu Dec 15 16:36:35 UTC 2011 by hortonfo
Upgrades: There are no upgrades in progress.

[Browse the filesystem](#)

[Namenode Logs](#)

[Go back to DFS home](#)

Live Datanodes : 1

Node	Last Contact	Admin State	Configured Capacity (GB)	Used (GB)	Non DFS Used (GB)	Remaining (GB)	Used (%)	Used (%)	Remaining (%)	Blocks
acloud01	0	In Service	559.19	0	311.17	248.02	0	0	44.35	1

This is Apache Hadoop release 1.0.0

localhost Hadoop Machine List

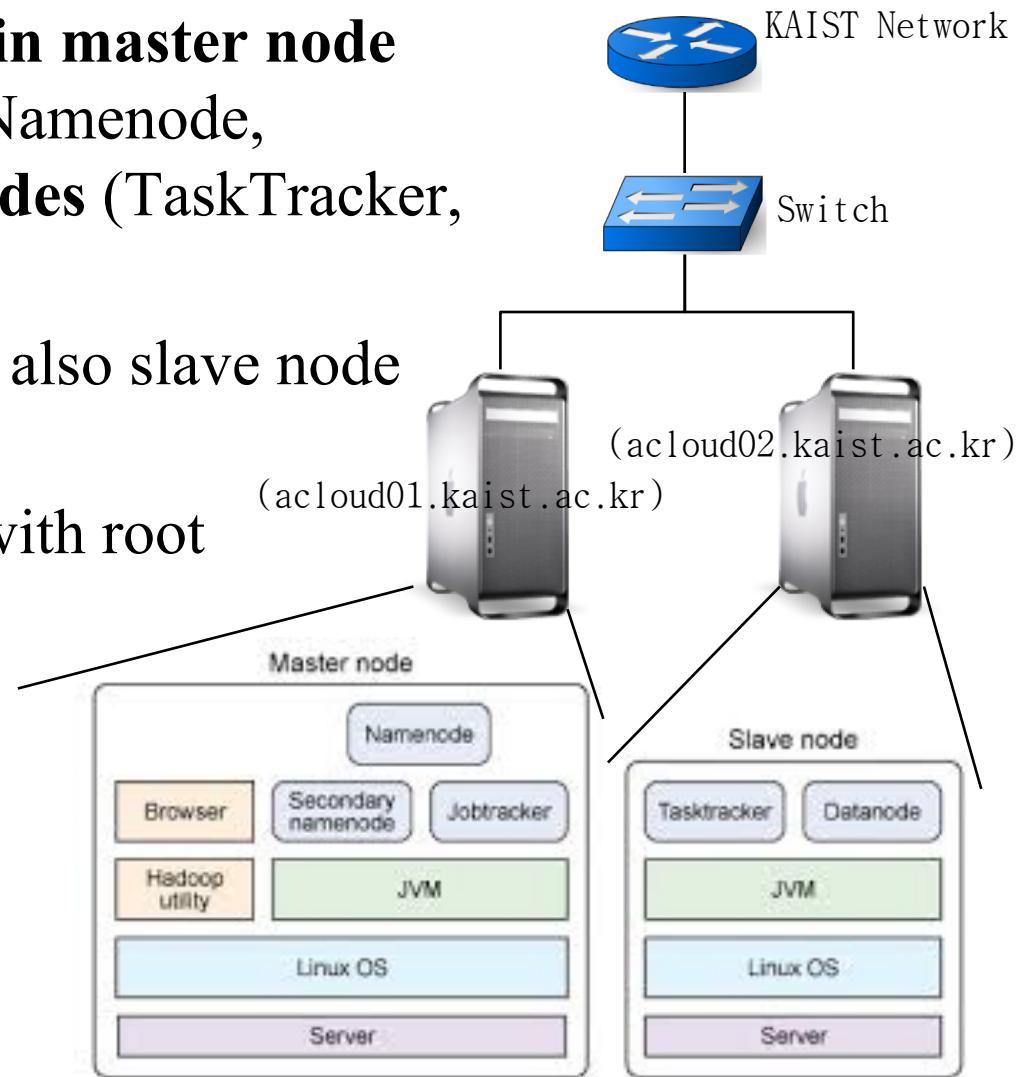
Active Task Trackers

Task Trackers														
Name	Host	# running tasks	Max Map Tasks	Max Reduce Tasks	Failures	Node Health Status	Seconds Since Node Last Healthy	Total Tasks Since Start	Succeeded Tasks Since Start	Total Tasks Last Day	Succeeded Tasks Last Day	Total Tasks Last Hour	Succeeded Tasks Last Hour	Seconds since heartbeat
tracker_acloud01.kaist.ac.kr:localhost/127.0.0.1:55112	acloud01.kaist.ac.kr	0	2	2	0	N/A	0	0	0	0	0	0	0	0

This is Apache Hadoop release 1.0.0

Fully-Distributed Mode Installation

- Hadoop processes runs in master node (JobTracker, Secondary Namenode, Namenode) and slave nodes (TaskTracker, Datanode) dispersedly**
 - We will use acloud01 as also slave node (resource waste)
1. Prepare linux machine with root authentication
 2. Java Installation
 3. Hadoop Download
 4. SSH configuration
 5. Hadoop Configuration



Fully-Distributed Mode Installation

1. Prepare linux machine with root authentication for every node
 - Prepare user account ‘Hadoop’ with root authentication

```
# User privilege specification
root    ALL=(ALL:ALL)  ALL
hadoop  ALL=(ALL:ALL)  ALL
```

- For Fully-Distributed mode installation, we will use ‘acloud01’ and ‘acloud02’ as hostname respectively

Fully-Distributed Mode Installation

1. Prepare linux machine with root authentication for every node

- Can modify the local machine's hostname in /etc/hostname and can modify the remote machine's hostname in /etc/hosts

```
hadoop@acloud01:~$ cat /etc/hostname
acloud01
```

```
hadoop@acloud01:~$ cat /etc/hosts
127.0.0.1      localhost

143.248.152.16 accloud01.kaist.ac.kr      accloud01
143.248.152.17 accloud02.kaist.ac.kr      accloud02

143.248.6.161   anclabvm1.kaist.ac.kr    avm1

# The following lines are desirable for IPv6 capable hosts
::1      ip6-localhost ip6-loopback
fe00::0  ip6-localnet
ff00::0  ip6-mcastprefix
ff02::1  ip6-allnodes
ff02::2  ip6-allrouters
```

```
hadoop@accloud02:~$ cat /etc/hostname
acloud02
```

```
hadoop@accloud02:~$ cat /etc/hosts
127.0.0.1      localhost

143.248.152.16 accloud01.kaist.ac.kr      accloud01
143.248.152.17 accloud02.kaist.ac.kr      accloud02

# The following lines are desirable for IPv6 capable hosts
::1      ip6-localhost ip6-loopback
fe00::0  ip6-localnet
ff00::0  ip6-mcastprefix
ff02::1  ip6-allnodes
ff02::2  ip6-allrouters
```

Fully-Distributed Mode Installation

2. Same as Pseudo-Distributed Mode Installation
3. Same as Pseudo-Distributed Mode Installation
4. SSH Configuration (Refer the Lab 11)
 - Can access to the hosts named ‘acloud01’ and ‘acloud02’ without password
 - Get the ‘id_rsa.pub’ file’s contents from every node and append the contents to ‘authorized_keys’ under `~/.ssh` directory
 - Copy ‘authorized_keys’ to every node

```
hadoop@acloud01:~/ssh$ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQC6hbJlE1g875BE5X06a/Zv+A2w7LbuAhrzNS5H71KC

ntpcoMCVg3RZsJwWD55OY0WT1AgEpK/8FxLBGXg8tH1fGIpBTPYcL1KY9Fbr4HHhowhhiuKetNRR3K8c
VSBY4M6fz8fvEiSTeutxh1/O1W5hZ13vY01pPfFGxTJRcOR+qK9ElldN8fFoz hadoop@acloud01
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQDmx1VDqCiIrw3A5WsVarE/4JNTvLhvMOqffzQRTo9

xi4ar3qhTTsMXGPrkDH0F2Xg+kbtuDYsYxB/8GOSArj3KV34H79QqS5p/sm1vAMrxFZiMXx1PvbQx1UM
gmkx0VAC1IXEOxvkghz1N3k8G80WwdbbjNDWERvCoQtZtPdaMc7wHF4TD6j3 hadoop@acloud02
```

Fully-Distributed Mode Installation

- Hadoop Configuration for Pseudo-Distributed Mode
 - masters, slaves

```
hadoop@acloud01:/opt/hadoop-1.0.0/conf$ cat masters
acloud01
hadoop@acloud01:/opt/hadoop-1.0.0/conf$ cat slaves
acloud01
acloud02
```
 - To launch Secondary Namenode module in acloud01 node and to launch slave modules in acloud01/acloud02, modify **conf/masters** and **conf/slaves** file
 - hadoop-env.sh

```
# The java implementation to use. Required.
export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-amd64/jre
```
 - Specify the location of Java installation in **conf/hadoop-env.sh**

Fully-Distributed Mode Installation

- Hadoop Configuration for Pseudo-Distributed Mode
 - conf/core-site.xml:

```
<configuration>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/opt/hdfs</value>
  </property>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://acloud01:9000 </value>
  </property>
</configuration>
```

- Above configuration define the local file system directory for HDFS storage
- Also define the namenode and its port for RPC

Fully-Distributed Mode Installation

- Hadoop Configuration for Pseudo-Distributed Mode

- conf/hdfs-site.xml:

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>2</value>
  </property>
</configuration>
```

- Above configuration define the number of replication for failure control

- conf/mapred-site.xml:

```
<configuration>
  <property>
    <name>mapred.job.tracker</name>
    <value>acloud01:9001</value>
  </property>
</configuration>
```

- Above configuration define the JobTracker node and its port for RPC

Detail options:

http://hadoop.apache.org/docs/stable/cluster_setup.html

C.H. Youn (Nov. 21, 2017)

Fully-Distributed Mode Launch

- HDFS (namenode) format
\$ bin/hadoop namenode -format

```

hadoop@acloud01:/opt/hadoop-1.0.0$ bin/hadoop namenode -format
Warning: $HADOOP_HOME is deprecated.

13/05/14 00:46:43 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = acloud01/143.248.152.16
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 1.0.0
STARTUP_MSG:   build = https://svn.apache.org/repos/asf/hadoop/common/branches/b
ranch-1.0 -r 1214675; compiled by 'hortonfo' on Thu Dec 15 16:36:35 UTC 2011
*****
13/05/14 00:46:43 INFO util.GSet: VM type      = 64-bit
13/05/14 00:46:43 INFO util.GSet: 2% max memory = 17.77875 MB
13/05/14 00:46:43 INFO util.GSet: capacity      = 2^21 = 2097152 entries
13/05/14 00:46:43 INFO util.GSet: recommended=2097152, actual=2097152
13/05/14 00:46:44 INFO namenode.FSNamesystem: fsOwner=hadoop
13/05/14 00:46:44 INFO namenode.FSNamesystem: supergroup=supergroup
13/05/14 00:46:44 INFO namenode.FSNamesystem: isPermissionEnabled=true
13/05/14 00:46:44 INFO namenode.FSNamesystem: dfs.block.invalidate.limit=100
13/05/14 00:46:44 INFO namenode.FSNamesystem: isAccessTokenEnabled=false accessTokenUpdateInterval=0 min(s), accessTokenLifetime=0 min(s)
13/05/14 00:46:44 INFO namenode.NameNode: Caching file names occurring more than
10 times
13/05/14 00:46:44 INFO common.Storage: Image file of size 112 saved in 0 seconds
.
13/05/14 00:46:44 INFO common.Storage: Storage directory /opt/hdfs/dfs/name has
been successfully formatted.
13/05/14 00:46:44 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at acloud01/143.248.152.16
*****/

```

Fully-Distributed Mode Launch

- Hadoop Launch command
 - \$ bin/start-all.sh
 - Slave nodes also launched with command master node using SSH remote command
- Can verify the operation of hadoop with
 - jps (Java Virtual Machine Process Status Tool) command
 - acloud01: Secondary Namenode, JobTracker, Namenode, TaskTraker, Datanode
 - acloud02: TaskTracker, Datanode

```

hadoop@acloud01:/opt/hadoop-1.0.0$ bin/start-all.sh
Warning: $HADOOP_HOME is deprecated.

starting namenode, logging to /opt/hadoop-1.0.0/libexec/../logs/hadoop-hadoop-na
menode-acloud01.out
acloud01: starting datanode, logging to /opt/hadoop-1.0.0/libexec/../logs/hadoop-
-hadoop-datanode-acloud01.out
acloud02: starting datanode, logging to /opt/hadoop-1.0.0/libexec/../logs/hadoop-
-hadoop-datanode-acloud02.out
acloud01: starting secondarynamenode, logging to /opt/hadoop-1.0.0/libexec/../lo
gs/hadoop-hadoop-secondarynamenode-acloud01.out
starting jobtracker, logging to /opt/hadoop-1.0.0/libexec/../logs/hadoop-hadoop-
jobtracker-acloud01.out
acloud01: starting tasktracker, logging to /opt/hadoop-1.0.0/libexec/../logs/had
oop-hadoop-tasktracker-acloud01.out
acloud02: starting tasktracker, logging to /opt/hadoop-1.0.0/libexec/../logs/had
oop-hadoop-tasktracker-acloud02.out
hadoop@accloud01:/opt/hadoop-1.0.0$ jps
880 SecondaryNameNode
1252 TaskTracker
988 JobTracker
370 NameNode
1351 Jps
617 DataNode

```

```

hadoop@accloud02:/opt$ jps
8516 TaskTracker
8390 DataNode
8593 Jps

```

Fully-Distributed Mode Launch

NameNode 'acloud01.kaist.ac.kr:9000'

Started: Tue May 14 01:12:11 KST 2013
Version: 1.0.0, r1214675
Compiled: Thu Dec 15 16:36:35 UTC 2011 by hortonfo
Upgrades: There are no upgrades in progress.

[Browse the filesystem](#)
[Namenode Logs](#)
[Go back to DFS home](#)

<http://acloud01.kaist.ac.kr:50070>: Namenode web interface
<http://acloud01.kaist.ac.kr:50030>: JobTracker web interface

Live Datanodes : 2

Node	Last Contact	Admin State	Configured Capacity (GB)	Used (GB)	Non DFS Used (GB)	Remaining (GB)	Used (%)	Used (%)	Remaining (%)	Blocks
acloud01	1	In Service	559.19	0	311.35	247.85	0	0	44.32	2
acloud02	0	In Service	174.27	0	120.6	53.68	0	0	30.8	1

This is Apache Hadoop release 1.0.0

acloud01 Hadoop Machine List

Active Task Trackers

Task Trackers														
Name	Host	# running tasks	Max Map Tasks	Max Reduce Tasks	Failures	Node Health Status	Seconds Since Node Last Healthy	Total Tasks Since Start	Succeeded Tasks Since Start	Total Tasks Last Day	Succeeded Tasks Last Day	Total Tasks Last Hour	Succeeded Tasks Last Hour	Seconds since heartbeat
tracker_acloud01.kaist.ac.kr:localhost/127.0.0.1:46498	acloud01.kaist.ac.kr	0	2	2	0	N/A	0	0	0	0	0	0	0	1
tracker_acloud02.kaist.ac.kr:localhost/127.0.0.1:33005	acloud02.kaist.ac.kr	0	2	2	0	N/A	0	0	0	0	0	0	0	1

This is Apache Hadoop release 1.0.0

Hadoop Web Interface

- HDFS: <http://acloud01.kaist.ac.kr:50070/dfshealth.jsp>
- JobTracker: <http://acloud01.kaist.ac.kr:50030/jobtracker.jsp>
- Can see the status of HDFS, MapReduce and their nodes
- Can browse the HDFS file system
- Also can see the job processing status

File: [/user/soc/output/part-00000](#)

Goto : [/user/soc/output](#) [go](#)

[Go back to dir listing](#)

[Advanced view/download options](#)

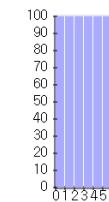
```
Bye      1
Goodbye  1
Hadoop   2
Hello    2
World    2
to       1
```

Hadoop job_201305140112_0005 on [acloud01](#)

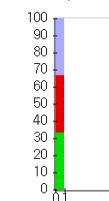
User: hadoop
 Job Name: wordcount
 Job File: https://acloud01:9000/opt/hdfs/mapred/staging/hadoop_staging/job_201305140112_0005/job.xml
 Submit Host: acloud01
 Submit Host Address: 143.248.152.16
 Job-ACLs: All users are allowed
 Job Setup: Successful
 Status: Succeeded
 Started at: Tue May 14 15:53:51 KST 2013
 Finished at: Tue May 14 15:54:24 KST 2013
 Finished in: 32sec
 Job Cleanup: Successful

Kind	% Complete	Num Tasks	Pending	Running	Complete	Killed	Failed/Killed Task Attempts
map	100.00%	3	0	0	3	0	0 / 0
reduce	100.00%	1	0	0	1	0	0 / 0

Map Completion Graph - [close](#)



Reduce Completion Graph - [close](#)



HDFS Commands

HDFS commands

- http://hadoop.apache.org/docs/r1.1.2/file_system_shell.html
 - Try each listed command
- Each student gets a home directory in HDFS
 - /user/your_account_name
- Examples:
 - hadoop fs –ls HDFS_directory
 - hadoop fs –put local_file HDFS_file
 - hadoop fs –get HDFS_file local_file
 - Other: -cat, -mkdir, -rm, -rmdir

Hadoop Distributed File System (HDFS)

- HDFS-shell guide

Comm.	Descriptions	examples
mkdir	Takes path uri's as argument and creates directories	hadoop fs -mkdir /user/hadoop/dir1
cp	Copy files from source to destination	hadoop fs -cp /user/hadoop/file1 /user/hadoop/file2
count	Copy files from source to destination	hadoop fs -cp /user/hadoop/file1 /user/hadoop/file2
get	Copy files to the local file system.	hadoop fs -get /user/hadoop/file localfile
put	Copy single src, or multiple srcs from local file system to the destination file system	hadoop fs -put localfile /user/hadoop/hadoopfile
cat	Copies source paths to stdout	hadoop fs -cat hdfs://nn1.example.com/file1
ls (lsr)	For a file returns	hadoop fs -ls /user
rm (rmr)	Delete files specified as args	hadoop fs -rm hdfs://nn.example.com/file

HDFS commands

- **mkdir**

Usage: hadoop fs -mkdir <paths>

Takes path uri's as argument and creates directories. The behavior is much like unix mkdir -p creating parent directories along the path.

- Example:

```
hadoop fs -mkdir /user/hadoop/dir1 /user/hadoop/dir2  
hadoop fs -mkdir hdfs://nn1.example.com/user/hadoop/dir  
hdfs://nn2.example.com/user/hadoop/dir
```

- Exit Code:

Returns 0 on success and -1 on error.

```
hadoop@acloud01:/opt/hadoop-1.0.0/bin$ hadoop fs -mkdir /user/soc7011  
hadoop@acloud01:/opt/hadoop-1.0.0/bin$ hadoop fs -ls /user  
Found 3 items  
drwxr-xr-x  - hadoop supergroup          0 2013-05-14 15:06 /user/hadoop  
drwxr-xr-x  - hadoop supergroup          0 2013-05-14 17:00 /user/soc  
drwxr-xr-x  - hadoop supergroup          0 2013-05-16 01:50 /user/soc7011
```

HDFS commands

- **ls**

Usage: hadoop fs -ls <args>

- For a file returns stat on the file with the following format:

permissions number_of_replicas userid groupid filesize
modification_date modification_time filename

- For a directory it returns list of its direct children as in unix. A directory is listed as:

permissions userid groupid modification_date
modification_time dirname

- Example:

```
hadoop fs -ls /user/hadoop/file1
```

- Exit Code:

Returns 0 on success and -1 on error.

```
hadoop@accloud01:/opt/hadoop-1.0.0/bin$ hadoop fs -ls /user/soc/input
Found 2 items
-rw-r--r--  2 hadoop supergroup          22 2013-05-14 15:51 /user/soc/input/file0
-rw-r--r--  2 hadoop supergroup          31 2013-05-14 15:51 /user/soc/input/file1
```

HDFS commands

- **lsr**

Usage: hadoop fs -lsr <args>

Recursive version of ls. Similar to Unix ls -R.

```
hadoop@acloud01:~/temp$ hadoop fs -lsr /user/soc
drwxr-xr-x  - hadoop supergroup          0 2013-05-14 17:00 /user/soc/ii_output
-rw-r--r--  2 hadoop supergroup          0 2013-05-14 17:00 /user/soc/ii_output/
drwxr-xr-x  - hadoop supergroup          0 2013-05-14 17:00 /user/soc/ii_output/
drwxr-xr-x  - hadoop supergroup          0 2013-05-14 17:00 /user/soc/ii_output/
-rw-r--r--  2 hadoop supergroup        19337 2013-05-14 17:00 /user/soc/ii_output/
-rw-r--r--  2 hadoop supergroup        19923 2013-05-14 17:00 /user/soc/ii_output/
-rw-r--r--  2 hadoop supergroup          91 2013-05-14 17:00 /user/soc/ii_output/p
drwxr-xr-x  - hadoop supergroup          0 2013-05-16 02:10 /user/soc/input
-rw-r--r--  2 hadoop supergroup         22 2013-05-16 02:10 /user/soc/input/file0
-rw-r--r--  2 hadoop supergroup         31 2013-05-14 15:51 /user/soc/input/file1
drwxr-xr-x  - hadoop supergroup          0 2013-05-14 15:54 /user/soc/output
-rw-r--r--  2 hadoop supergroup          0 2013-05-14 15:54 /user/soc/output/_SUO
drwxr-xr-x  - hadoop supergroup          0 2013-05-14 15:53 /user/soc/output/_log
drwxr-xr-x  - hadoop supergroup          0 2013-05-14 15:53 /user/soc/output/_log
-rw-r--r--  2 hadoop supergroup        19332 2013-05-14 15:53 /user/soc/output/_log
-rw-r--r--  2 hadoop supergroup        20228 2013-05-14 15:53 /user/soc/output/_log
-rw-r--r--  2 hadoop supergroup          46 2013-05-14 15:54 /user/soc/output/part
```

HDFS commands

- **count**

Usage: hadoop fs -count [-q] <paths>

Count the number of directories, files and bytes under the paths that match the specified file pattern.

- The output columns with -count are:

DIR_COUNT, FILE_COUNT, CONTENT_SIZE FILE_NAME

- The output columns with -count -q are:

QUOTA, REMAINING_QUATA, SPACE_QUOTA,

REMAINING_SPACE_QUOTA, DIR_COUNT, FILE_COUNT,

CONTENT_SIZE, FILE_NAME

- Example:

```
hadoop fs -count hdfs://nn1.example.com/file1
```

```
hdfs://nn2.example.com/file2
```

```
hadoop fs -count -q hdfs://nn1.example.com/file1
```

- Exit Code:

Returns 0 on success and -1 on error.

```
hadoop@acloud01:/opt/hadoop-1.0.0/bin$ hadoop fs -count /user/soc
8          10          79010 hdfs://acloud01:9000/user/soc
```

HDFS commands

- **get**

Usage: hadoop fs -get [-ignorecrc] [-crc] <src> <localdst>

Copy files to the local file system. Files that fail the CRC check may be copied with the `-ignorecrc` option. Files and CRCs may be copied using the `-crc` option.

- Example:

```
hadoop fs -get /user/hadoop/file localfile
```

```
hadoop fs -get hdfs://nn.example.com/user/hadoop/file  
localfile
```

- Exit Code:

Returns 0 on success and -1 on error.

```
hadoop@acloud01:~/temp$ ls  
hadoop@acloud01:~/temp$ hadoop fs -get /user/soc/input/file0 file0  
hadoop@acloud01:~/temp$ ls  
file0
```

HDFS commands

- **put**

Usage: hadoop fs -put <localsrc> ... <dst>

Copy single src, or multiple srcs from local file system to the destination file system. Also reads input from stdin and writes to destination file system.

- Example:

```
hadoop fs -put localfile /user/hadoop/hadoopfile  
hadoop fs -put localfile1 localfile2  
/user/hadoop/hadoopdir  
hadoop fs -put localfile  
hdfs://nn.example.com/hadoop/hadoopfile
```

- Exit Code:

```
Returns 0 on success and 1 on error  
hadoop@acloud01:~/temp$ hadoop fs -ls /user/soc/input  
Found 1 items  
-rw-r--r-- 2 hadoop supergroup 31 2013-05-14 15:51 /user/soc/input/file1  
hadoop@acloud01:~/temp$ ls  
file0  
hadoop@acloud01:~/temp$ hadoop fs -put file0 /user/soc/input  
hadoop@accloud01:~/temp$ hadoop fs -ls /user/soc/input  
Found 2 items  
-rw-r--r-- 2 hadoop supergroup 22 2013-05-16 02:10 /user/soc/input/file0  
-rw-r--r-- 2 hadoop supergroup 31 2013-05-14 15:51 /user/soc/input/file1
```

HDFS commands

- **rm**

Usage: hadoop fs -rm [-skipTrash] URI [URI ...]

Delete files specified as args. Only deletes non empty directory and files. If the -skipTrash option is specified, the trash, if enabled, will be bypassed and the specified file(s) deleted immediately. This can be useful when it is necessary to delete files from an over-quota directory. Refer to rmr for recursive deletes.

- Example:

```
hadoop fs -rm hdfs://nn.example.com/file  
/user/hadoop/emptydir
```

- Exit Code:

Returns 0 on success and -1 on error.

```
hadoop@acloud01:~/temp$ hadoop fs -ls /user/soc/input  
Found 2 items  
-rw-r--r-- 2 hadoop supergroup 22 2013-05-14 15:51 /user/soc/input/file0  
-rw-r--r-- 2 hadoop supergroup 31 2013-05-14 15:51 /user/soc/input/file1  
hadoop@acloud01:~/temp$ hadoop fs -rm /user/soc/input/file0  
Deleted hdfs://acloud01:9000/user/soc/input/file0  
hadoop@acloud01:~/temp$ hadoop fs -ls /user/soc/input  
Found 1 items  
-rw-r--r-- 2 hadoop supergroup 31 2013-05-14 15:51 /user/soc/input/file1
```

HDFS commands

- **rmr**

Usage: `hadoop fs -rmr [-skipTrash] URI [URI ...]`

Recursive version of delete. If the `-skipTrash` option is specified, the trash, if enabled, will be bypassed and the specified file(s) deleted immediately. This can be useful when it is necessary to delete files from an over-quota directory.

- Example:

```
hadoop fs -rmr /user/hadoop/dir
```

```
hadoop fs -rmr hdfs://nn.example.com/user/hadoop/dir
```

- Exit Code:

Returns 0 on success and -1 on error.

```
hadoop@acloud01:~/temp$ hadoop fs -ls /user
Found 3 items
drwxr-xr-x  - hadoop supergroup          0 2013-05-14 15:06 /user/hadoop
drwxr-xr-x  - hadoop supergroup          0 2013-05-14 17:00 /user/soc
drwxr-xr-x  - hadoop supergroup          0 2013-05-16 01:50 /user/soc7011
hadoop@acloud01:~/temp$ hadoop fs -rmr /user/soc7011
Deleted hdfs://acloud01:9000/user/soc7011
hadoop@acloud01:~/temp$ hadoop fs -ls /user
Found 2 items
drwxr-xr-x  - hadoop supergroup          0 2013-05-14 15:06 /user/hadoop
drwxr-xr-x  - hadoop supergroup          0 2013-05-14 17:00 /user/soc
```

Execution of MapReduce Example

Execution of MapReduce Example

- Hadoop gives example MapRecude source code and execution file
 - hadoop-examples-1.0.0.jar
 - src/examples/org/apache/hadoop/examples

```
hadoop@acloud01:/opt/hadoop-1.0.0$ ls
bin          conf          hadoop-core-1.0.0.jar    ivy      LICENSE.txt  sbin
build.xml    contrib        hadoop-examples-1.0.0.jar  ivy.xml  logs       share
c++          docs          hadoop-test-1.0.0.jar   lib      NOTICE.txt  src
CHANGES.txt  hadoop-ant-1.0.0.jar  hadoop-tools-1.0.0.jar libexec README.txt  webapps

hadoop@accloud01:/opt/hadoop-1.0.0$ cd src/examples/org/apache/hadoop/examples/
hadoop@accloud01:/opt/hadoop-1.0.0/src/examples/org/apache/hadoop/examples$ ls
AggregateWordCount.java      Grep.java           RandomTextWriter.java  terasort
AggregateWordHistogram.java  Join.java           RandomWriter.java    WordCount.java
dancing                      MultiFileWordCount.java SecondarySort.java
DBCountPageView.java         package.html       SleepJob.java
ExampleDriver.java           PiEstimator.java  Sort.java
```

Execution of MapReduce Example

- Hadoop gives number of examples listed below

```
$ hadoop jar hadoop-examples-1.0.0.jar
```

```
hadoop@acloud01:/opt/hadoop-1.0.0$ hadoop jar hadoop-examples-1.0.0.jar
An example program must be given as the first argument.
Valid program names are:
aggregatewordcount: An Aggregate based map/reduce program that counts the words in the input files.
aggregatewordhist: An Aggregate based map/reduce program that computes the histogram of the words in the input files.
dbcount: An example job that count the pageview counts from a database.
grep: A map/reduce program that counts the matches of a regex in the input.
join: A job that effects a join over sorted, equally partitioned datasets
multifilewc: A job that counts words from several files.
pentomino: A map/reduce tile laying program to find solutions to pentomino problems.
pi: A map/reduce program that estimates Pi using monte-carlo method.
randomtextwriter: A map/reduce program that writes 10GB of random textual data per node.
randomwriter: A map/reduce program that writes 10GB of random data per node.
secondariesort: An example defining a secondary sort to the reduce.
sleep: A job that sleeps at each map and reduce task.
sort: A map/reduce program that sorts the data written by the random writer.
sudoku: A sudoku solver.
teragen: Generate data for the terasort
terasort: Run the terasort
teravalidate: Checking results of terasort
wordcount: A map/reduce program that counts the words in the input files.
```

Execution of MapReduce Example

- Hadoop jar execution command

\$ hadoop jar [local jar file] [execution class with package name] [arguments]

\$ hadoop jar hadoop-examples-1.0.0.jar pi

```
hadoop@accloud01:/opt/hadoop-1.0.0$ hadoop jar hadoop-examples-1.0.0.jar pi
Usage: org.apache.hadoop.examples.PiEstimator <nMaps> <nSamples>
Generic options supported are
-conf <configuration file>      specify an application configuration file
-D <property=value>             use value for given property
-fs <local|namenode:port>       specify a namenode
-jt <local|jobtracker:port>     specify a job tracker
-files <comma separated list of files>  specify comma separated files to be copied to the map reduce cluster
-libjars <comma separated list of jars>   specify comma separated jar files to include in the classpath.
-archives <comma separated list of archives>  specify comma separated archives to be unarchived on the compute machines.

The general command line syntax is
bin/hadoop command [genericOptions] [commandOptions]
```

Needs more argument <nMaps> <nSamples>

Execution of MapReduce Example

- Hadoop jar execution command
 - \$ hadoop jar [local jar file] [execution class with package name] [arguments]
 - \$ hadoop jar hadoop-examples-1.0.0.jar pi 4 10000

```
hadoop@acloud01:/opt/hadoop-1.0.0$ hadoop jar hadoop-examples-1.0.0.jar pi 4 10000
Number of Maps = 4
Samples per Map = 10000
Wrote input for Map #0
Wrote input for Map #1
Wrote input for Map #2
Wrote input for Map #3
Starting Job
13/05/16 02:34:25 INFO mapred.FileInputFormat: Total input paths to process : 4
13/05/16 02:34:25 INFO mapred.JobClient: Running job: job_201305160148_0001
13/05/16 02:34:26 INFO mapred.JobClient: map 0% reduce 0%
13/05/16 02:34:41 INFO mapred.JobClient: map 100% reduce 0%
13/05/16 02:34:54 INFO mapred.JobClient: map 100% reduce 100%
13/05/16 02:34:59 INFO mapred.JobClient: Job complete: job_201305160148_0001
```

```
Job Finished in 33.686 seconds  
Estimated value of Pi is 3.14140000000000000000
```

Execution of MapReduce Example

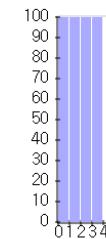
- Job Status Monitoring with web interface
<http://acloud01.kaist.ac.kr:50030/jobtracker.jsp>

Hadoop job_201305160148_0001 on [acloud01](#)

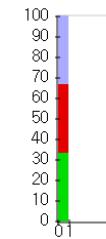
User: hadoop
 Job Name: PiEstimator
 Job File: http://acloud01:9000/opt/hdfs/mapred/staging/hadoop/.staging/job_201305160148_0001/job.xml
 Submit Host: acloud01
 Submit Host Address: 143.248.152.16
 Job-ACLs: All users are allowed
 Job Setup: [Successful](#)
 Status: Succeeded
 Started at: Thu May 16 02:34:25 KST 2013
 Finished at: Thu May 16 02:34:58 KST 2013
 Finished in: 32sec
 Job Cleanup: [Successful](#)

Kind	% Complete	Num Tasks	Pending	Running	Complete	Killed	Failed/Killed Task Attempts
map	 100.00%	4	0	0	4	0	0 / 0
reduce	 100.00%	1	0	0	1	0	0 / 0

Map Completion Graph - [close](#)



Reduce Completion Graph - [close](#)



copy

sort

reduce

Lab 13-1 Requirements

- Please install hadoop framework in VM environment with Fully-Distributed Mode Installation configuration
 - Refer the page 24
- Please compare the distributed computing frameworks. Figure out the differences between MPI framework and Hadoop framework (PiEstimator example using monte-carlo method will be helpful)

Lab 13-1 Requirements

- Please explain the all job status metrics which is shown in right figure. Also analysis specific MapReduce example using the metrics

	Counter	Map	Reduce	Total
Job Counters	SLOTS_MILLIS_MAPS	0	0	15,539
	Launched reduce tasks	0	0	1
	Total time spent by all reduces waiting after reserving slots (ms)	0	0	0
	Total time spent by all maps waiting after reserving slots (ms)	0	0	0
	Launched map tasks	0	0	2
	Data-local map tasks	0	0	2
	SLOTS_MILLIS_REDUCES	0	0	9,955
File Output Format Counters	Bytes Written	0	46	46
File Input Format Counters	Bytes Read	53	0	53
FileSystemCounters	FILE_BYTES_READ	0	88	88
	HDFS_BYTES_READ	265	0	265
	FILE_BYTES_WRITTEN	43,080	21,550	64,630
	HDFS_BYTES_WRITTEN	0	46	46
Map-Reduce Framework	Map output materialized bytes	94	0	94
	Map input records	2	0	2
	Reduce shuffle bytes	0	0	0
	Spilled Records	7	7	14
	Map output bytes	89	0	89
	CPU time spent (ms)	360	800	1,160
	Total committed heap usage (bytes)	369,295,360	200,998,912	570,294,272
	Combine input records	9	0	9
	SPLIT_RAW_BYTES	212	0	212
	Reduce input records	0	7	7
	Reduce input groups	0	6	6
	Combine output records	7	0	7
	Physical memory (bytes) snapshot	396,623,872	112,578,560	509,202,432
	Reduce output records	0	6	6
	Virtual memory (bytes) snapshot	4,646,756,352	2,628,759,552	7,275,515,904
	Map output records	9	0	9

LAB 13-2) HADOOP PROGRAMMING

WordCount

```

public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable>{
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException{
        String line = value.toString();
        StringTokenizer tokenizer = new StringTokenizer(line);
        while (tokenizer.hasMoreTokens()){
            word.set(tokenizer.nextToken());
            output.collect(word, one);
        }
    }
}

```

mapper

shuffle
(Grouping and sorting)

```

public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable>{
    public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {
        int sum=0;
        while (values.hasNext()){
            sum += values.next().get();
        }
        output.collect(key, new IntWritable(sum));
    }
}

```

partitioner

combiner

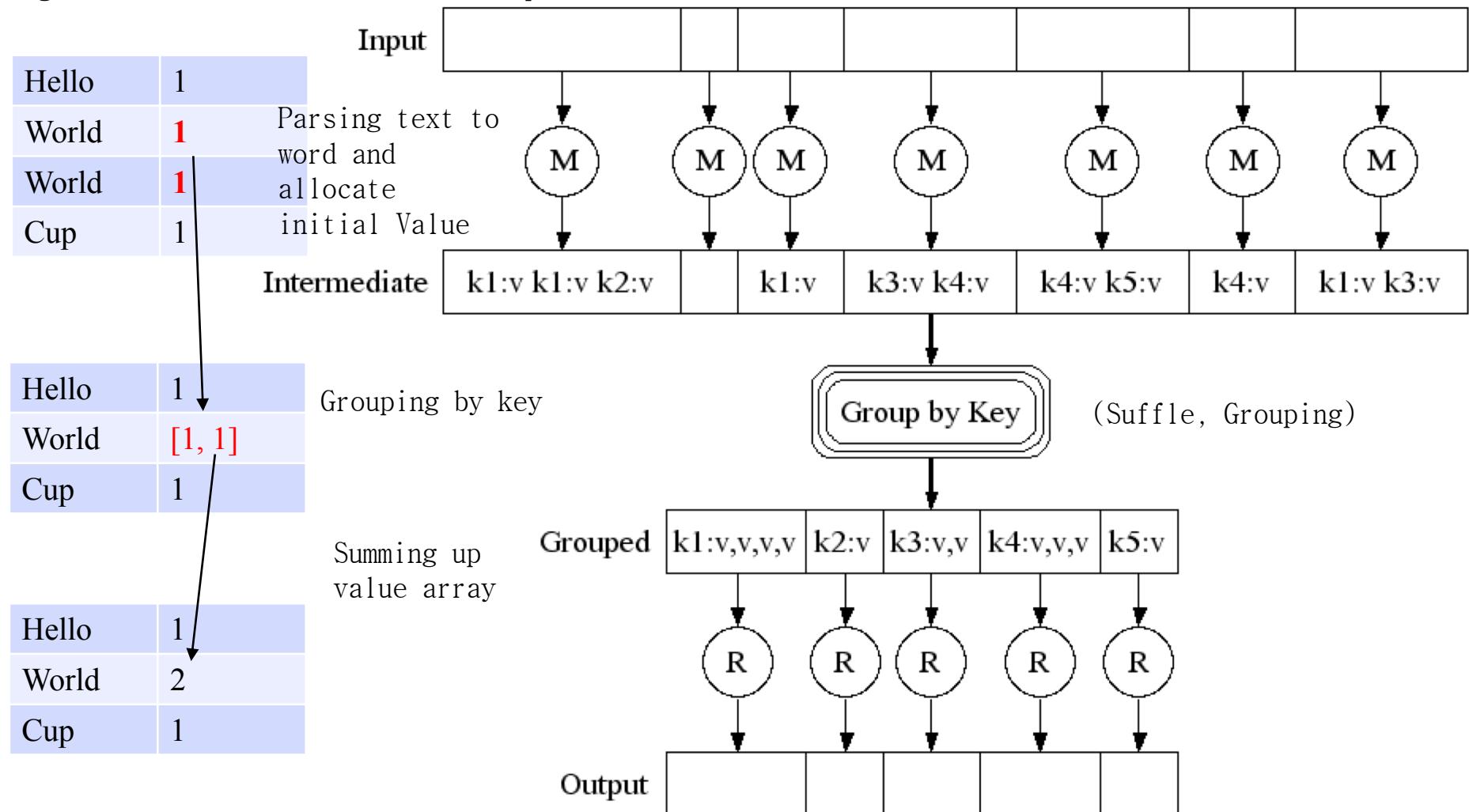
reducer

WordCount

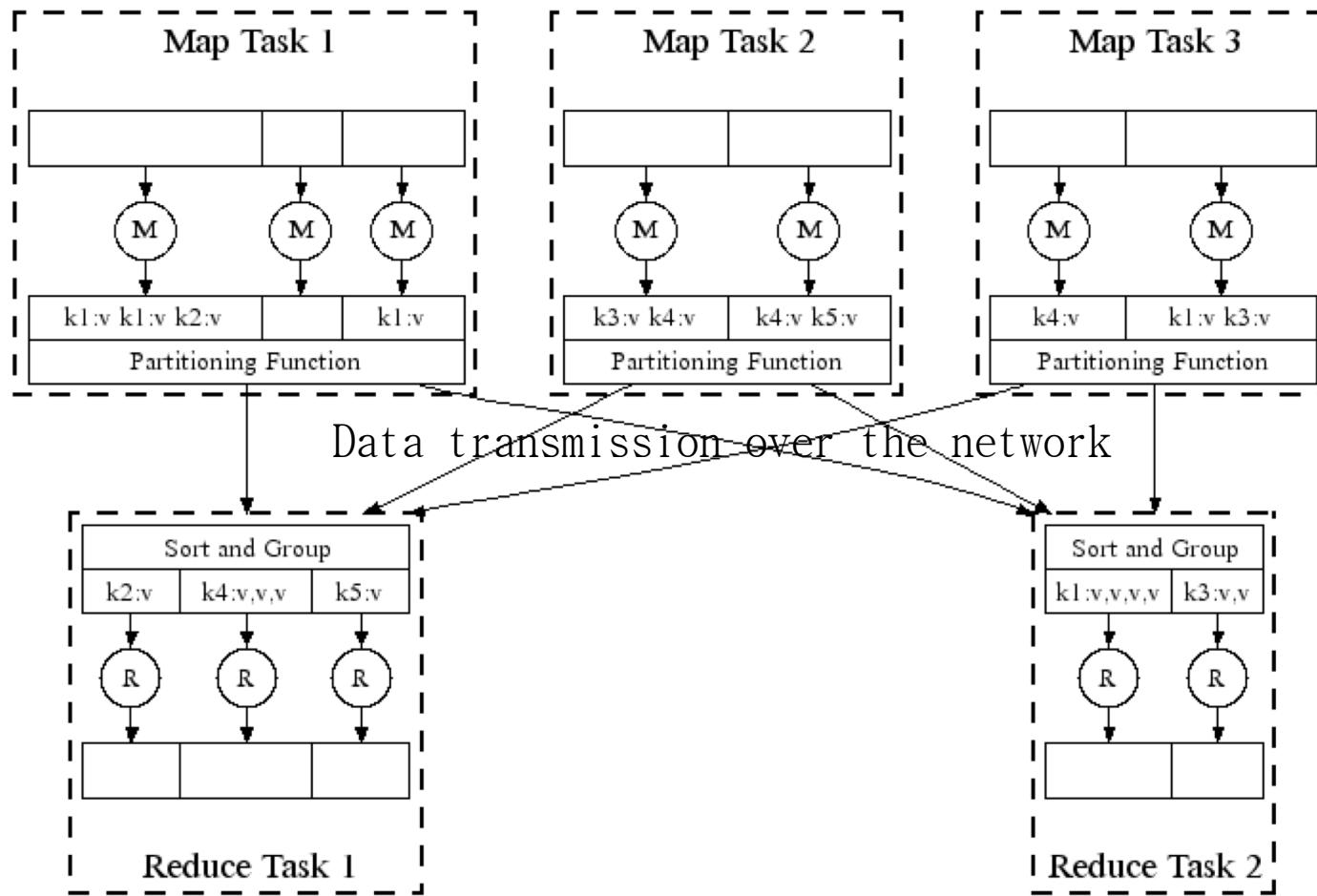
```
public static void main(String[] args) throws Exception{  
    JobConf conf = new JobConf(WordCount.class);  
    conf.setJobName("wordcount");  
    conf.setOutputKeyClass(Text.class);  
    conf.setOutputValueClass(IntWritable.class);  
    conf.setMapperClass(Map.class);  
    conf.setCombinerClass(Reduce.class);  
    conf.setReducerClass(Reduce.class);  
    conf.setInputFormat(TextInputFormat.class);  
    conf.setOutputFormat(TextOutputFormat.class);  
  
    FileInputFormat.setInputPaths(conf, new Path(args[0]));  
    FileOutputFormat.setOutputPath(conf, new Path(args[1]));  
  
    RunningJob job = JobClient.runJob(conf);  
}
```

Execution

Original Data - Hello World World Cup



Parallel Execution



Combiners

- Simply it can be said as “local reducer” for cutting down the data transmission over Map to Reduce task
- Often a map task will produce many pairs of the form (k, v_1) , (k, v_2) , ... for the same key k
 - E.g., popular words in Word Count
- Can save network time by pre-aggregating at mapper
 - $\text{combine}(k_1, \text{list}(v_1)) \rightarrow v_2$
 - Usually same as reduce function
- Works only if reduce function is commutative and associative

Partition Function

- Inputs to map tasks are created by contiguous splits of input file
- For reduce, we need to ensure that records with the same intermediate key end up at the same worker
- System uses a default partition function e.g.,
 $\text{hash}(\text{key}) \bmod R$
- Sometimes useful to override
 - E.g., $\text{hash}(\text{hostname(URL)}) \bmod R$ ensures URLs from a host end up in the same output file

WordCount

1. Build word count application

```
$ javac -classpath /opt/hadoop-1.0.0-core.jar -d wordcount_classes  
WordCount.java
```

2. Make it to archived file

```
$ jar -cvf WordCount.jar -C wordcount_classes/ .
```

3. Setting up input files

4. Execute WordCount Job in MapReduce

```
hadoop@accloud01:~$ hadoop jar WordCount.jar org.myorg.WordCount /user/soc/input /user/soc/output  
13/05/14 15:53:50 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. Applications should implement Tool for the same.  
13/05/14 15:53:51 INFO mapred.FileInputFormat: Total input paths to process : 2  
13/05/14 15:53:51 INFO mapred.JobClient: Running job: job_201305140112_0005  
13/05/14 15:53:52 INFO mapred.JobClient: map 0% reduce 0%  
13/05/14 15:54:07 INFO mapred.JobClient: map 100% reduce 0%  
13/05/14 15:54:19 INFO mapred.JobClient: map 100% reduce 100%  
13/05/14 15:54:24 INFO mapred.JobClient: Job complete: job_201305140112_0005
```

5. Execution Result

```
hadoop@accloud01:~$ hadoop fs -cat /user/soc/output/part-00000  
Bye      1  
Goodbye  1  
Hadoop   2  
Hello    2  
World    2  
to       1
```

WordCount

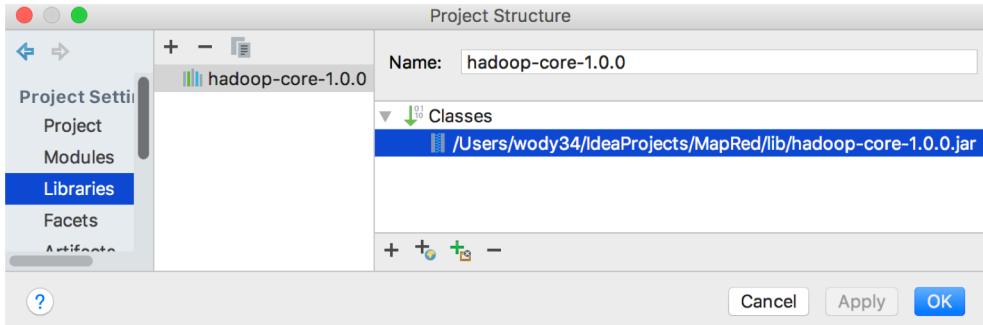
- hadoop@ee6130\$ hadoop fs -put test_wc.txt /user/20137011
- hadoop@ee6130\$ hadoop jar WordCount.jar WordCount /user/20137011 /user/20137011/wc_result
12/05/14 22:06:50 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. Applications should implement Tool for the same.
12/05/14 22:06:50 INFO mapred.FileInputFormat: Total input paths to process : 100
12/05/14 22:06:51 INFO mapred.JobClient: Running job: job_201205142002_0008
12/05/14 22:06:52 INFO mapred.JobClient: map 0% reduce 0%
12/05/14 22:06:59 INFO mapred.JobClient: map 1% reduce 0%
12/05/14 22:07:01 INFO mapred.JobClient: map 3% reduce 0%
12/05/14 22:07:02 INFO mapred.JobClient: map 6% reduce 0%
12/05/14 22:07:04 INFO mapred.JobClient: map 7% reduce 0%
12/05/14 22:07:05 INFO mapred.JobClient: map 9% reduce 0%
12/05/14 22:07:07 INFO mapred.JobClient: map 11% reduce 0%
12/05/14 22:07:08 INFO mapred.JobClient: map 14% reduce 0%
12/05/14 22:07:10 INFO mapred.JobClient: map 16% reduce 0%
12/05/14 22:07:11 INFO mapred.JobClient: map 18% reduce 3%
12/05/14 22:07:13 INFO mapred.JobClient: map 20% reduce 3%
12/05/14 22:07:14 INFO mapred.JobClient: map 22% reduce 3%
12/05/14 22:07:16 INFO mapred.JobClient: map 24% reduce 3%
12/05/14 22:07:17 INFO mapred.JobClient: map 26% reduce 3%
12/05/14 22:08:23 INFO mapred.JobClient: map 100% reduce 100%
- hadoop@ee6130\$ hadoop fs -get /user/20137011/wc_result/part-00000 wc_result
- hadoop@ee6130\$ vi wc_result

<http://143.248.146.155:50030/jobtracker.jsp>

<http://143.248.146.155:50070/dfshealth.jsp>

Development with IntelliJ

- Download Project File in
<https://github.com/wody34/ee614/raw/master/LabMaterials/MapRed.zip>
- [Setup library] Setup project library in project setting



- [Build Project] Build → Build project
- [JAR Generation] Build Artifacts for JAR generation: Build → Build Artifacts → MapRed → Build
MapRed.jar will generated under out/artifacts/MapRed/MapRed.jar
- [Move executable] Copy MapRed.jar into the machine (143.248.146.155)
- [Execute] hadoop jar MapRed.jar WordCount /user/20137011/test_wc.txt
/user/20137011/wc_result
- [Download Result] hadoop fs -get /user/20137011/wc_result wc_result
- [Validate Result] cat wc_result/part-00000

Inverted Index

- An inverted index is a mapping of words to their location in a set of documents. Most modern search engines utilize some form of an inverted index to process user-submitted queries. In its most basic form, an inverted index is a simple hash table which maps words in the documents to some sort of document identifier. For example, if given the following 2 documents:

```
Doc1: Buffalo buffalo Buffalo buffalo buffalo Buffalo buffalo.  
Doc2: Buffalo are mammals.
```

- we could construct the following inverted file index:

```
Buffalo -> Doc1, Doc2  
buffalo -> Doc1  
buffalo. -> Doc1  
are -> Doc2  
mammals. -> Doc2
```

Inverted Index

- Setting up input files

```

hadoop@acloud01:~$ hadoop fs -put file0 /user/soc/input
hadoop@acloud01:~$ hadoop fs -put file1 /user/soc/input
hadoop@acloud01:~$ hadoop fs -ls /user/soc/input
Found 2 items
-rw-r--r-- 2 hadoop supergroup          22 2013-05-14 15:51 /user/soc/input/file0
-rw-r--r-- 2 hadoop supergroup          31 2013-05-14 15:51 /user/soc/input/file1
hadoop@accloud01:~$ hadoop fs -cat /user/soc/input/file0
Hello World Bye World
hadoop@accloud01:~$ hadoop fs -cat /user/soc/input/file1
Hello Hadoop Goodbye to Hadoop

```

- Execute InvertedIndex Job in MapReduce

```

hadoop@accloud01:~/InvertedIndexer$ hadoop jar InvertedIndexer.jar InvertedIndexer /user/soc/input /user/soc/ii_output
13/05/14 17:00:29 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. Applications should implement Tool for the same.
13/05/14 17:00:30 INFO mapred.FileInputFormat: Total input paths to process : 2
13/05/14 17:00:30 INFO mapred.JobClient: Running job: job_201305140112_0006
13/05/14 17:00:31 INFO mapred.JobClient: map 0% reduce 0%
13/05/14 17:00:44 INFO mapred.JobClient: map 66% reduce 0%
13/05/14 17:00:46 INFO mapred.JobClient: map 100% reduce 0%
13/05/14 17:00:56 INFO mapred.JobClient: map 100% reduce 100%
13/05/14 17:01:01 INFO mapred.JobClient: Job complete: job_201305140112_0006

```

- Execution Result

```

hadoop@accloud01:~/InvertedIndexer$ hadoop fs -cat /user/soc/ii_output/part-00000
bye      file0
goodbye  file1
hadoop   file1, file1
hello    file1, file0
to       file1
world   file0, file0

```

Inverted Index

```

public class InvertedIndexer {
    public static class InvertedIndexMapper extends MapReduceBase implements Mapper<LongWritable, Text, Text, Text> {
        private final static Text word = new Text();
        private final static Text location = new Text();

        public void map(LongWritable key, Text val, OutputCollector<Text, Text> output, Reporter reporter) throws IOException {
            FileSplit fileSplit = (FileSplit)reporter.getInputSplit();
            String fileName = fileSplit.getPath().getName();
            location.set(fileName);

            String line = val.toString();
            StringTokenizer itr = new StringTokenizer(line.toLowerCase());
            while (itr.hasMoreTokens()) {
                word.set(itr.nextToken());
                output.collect(word, location);
            }
        }
    }
}

public static class InvertedIndexReducer extends MapReduceBase implements Reducer<Text, Text, Text, Text> {
    public void reduce(Text key, Iterator<Text> values, OutputCollector<Text, Text> output, Reporter reporter) throws IOException {
        boolean first = true;
        StringBuilder toReturn = new StringBuilder();
        while (values.hasNext()){
            if (!first)
                toReturn.append(", ");
            first=false;
            toReturn.append(values.next().toString());
        }
        output.collect(key, new Text(toReturn.toString()));
    }
}

```

Original Data - file1: Hello World
file 2: World Cup

Hello	file1	
World	file1	Parsing text to word and allocate
World	file2	
Cup	file2	location Value

↓
Hello file1
World [file1, file2]
Cup file2

Hello	file1	
World	[file1, file2]	Grouping by key
Cup	file2	

Hello	'file1'	
World	'file1, file2'	Append string with value array
Cup	'file2'	

Inverted Index

```
/**  
 * The actual main() method for our program; this is the  
 * "driver" for the MapReduce job.  
 */  
public static void main(String[] args) {  
    JobClient client = new JobClient();  
    JobConf conf = new JobConf(InvertedIndexer.class);  
  
    conf.setJobName("InvertedIndexer");  
  
    conf.setOutputKeyClass(Text.class);  
    conf.setOutputValueClass(Text.class);  
  
    FileInputFormat.addInputPath(conf, new Path(args[0]));  
    FileOutputFormat.setOutputPath(conf, new Path(args[1]));  
  
    conf.setMapperClass(InvertedIndexMapper.class);  
    conf.setReducerClass(InvertedIndexReducer.class);  
  
    client.setConf(conf);  
  
    try {  
        JobClient.runJob(conf);  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
}
```

PiEstimator

- A Map-reduce program to estimate the value of Pi using quasi-Monte Carlo method
- Mapper: Generate points in a unit square and then count points inside/outside of the inscribed circle of the square
- Reducer: Accumulate points inside/outside results from the mappers
- Let $\text{numTotal} = \text{numInside} + \text{numOutside}$. The fraction $\text{numInside}/\text{numTotal}$ is a rational approximation of the value $(\text{Area of the circle})/(\text{Area of the square})$, where the area of the inscribed circle is $\pi/4$ and the area of unit square is 1
- Then, Pi is estimated value to be $4(\text{numInside}/\text{numTotal})$

PiEstimator

```
66 public class PiEstimator extends Configured implements Tool {  
  
    //tmp directory for input/output  
67  
68    static private final Path TMP_DIR = new Path(  
69        PiEstimator.class.getSimpleName() + "_TMP_3_141592654");  
  
    //2-dimensional Halton sequence {H(i)}, where H(i) is a 2-dimensional point and i >= 1 is the index. Halton sequence is used to generate  
    sample points for Pi estimation.  
74  
75    private static class HaltonSequence {  
  
        //Bases  
76  
77        static final int[] P = {2, 3};  
  
        //Maximum number of digits allowed  
78  
79        static final int[] K = {63, 40};  
80  
81        private long index;  
82        private double[] x;  
83        private double[][] q;  
84        private int[][] d;
```

PiEstimator

//Initialize to H(startindex), so the sequence begins with H(startindex+1).

```

88
89 HaltonSequence(long startindex) {
90     index = startindex;                                //Compute next point. Assume the current point is H(index). Compute H(index+1).
91     x = new double[K.length];                          //Returns:
92     q = new double[K.length][];                      //a 2-dimensional point with coordinates in [0,1)^2
93     d = new int[K.length][];                         117
94     for(int i = 0; i < K.length; i++) {               118    double[] nextPoint() {
95         q[i] = new double[K[i]];                     119        index++;
96         d[i] = new int[K[i]];                       120        for(int i = 0; i < K.length; i++) {
97     }                                              121            for(int j = 0; j < K[i]; j++) {
98     for(int i = 0; i < K.length; i++) {             122                d[i][j]++;
99         long k = index;                           123                x[i] += q[i][j];
100        x[i] = 0;                                 124                if (d[i][j] < P[i]) {
101        for(int j = 0; j < K[i]; j++) {           125                    break;
102            q[i][j] = (j == 0? 1.0: q[i][j-1])/P[i]; 126                }
103            d[i][j] = (int)(k % P[i]);              127                d[i][j] = 0;
104            k = (k - d[i][j])/P[i];                 128                x[i] -= (j == 0? 1.0: q[i][j-1]);
105            x[i] += d[i][j] * q[i][j];               129        }
106        }                                            130    }
107    }                                              131    return x;
108 }                                              132 }
109 }                                              133 }
```

PiEstimator

//Mapper class for Pi estimation. Generate points in a unit square and then count points inside/outside of the inscribed circle of the square.

139

```
140 public static class PiMapper extends MapReduceBase
141   implements Mapper<LongWritable, LongWritable, BooleanWritable,
LongWritable> {
```

//Map method.

//Parameters:

//offset - samples starting from the (offset+1)th sample.

//size - the number of samples for this map

//out - output {ture->numInside, false->numOutside}

//reporter -

148

```
149   public void map(LongWritable offset,
150     LongWritable size,
151     OutputCollector<BooleanWritable, LongWritable> out,
152     Reporter reporter) throws IOException {
153
```

154 final HaltonSequence haltonsequence = new HaltonSequence(offset.get());

155 long numInside = 0L;

156 long numOutside = 0L;

157

158 for(long i = 0; i < size.get();) {

159 //generate points in a unit square

160 final double[] point = haltonsequence.nextPoint();

161

//count points inside/outside of the inscribed circle of the square

163 final double x = point[0] - 0.5;

164 final double y = point[1] - 0.5;

165 if (x*x + y*y > 0.25) {

166 numOutside++;

167 } else {

168 numInside++;

169 }

170

171 //report status

172 i++;

173 if (i % 1000 == 0) {

174 reporter.setStatus("Generated " + i + " samples.");

175 }

176 }

177

178 //output map results

179 out.collect(new BooleanWritable(true), new
LongWritable(numInside));

180 out.collect(new BooleanWritable(false), new
LongWritable(numOutside));

181 }

182 }

PiEstimator

```

//Reducer class for Pi estimation. Accumulate points inside/outside results from the
mappers.
187
188 public static class PiReducer extends MapReduceBase
189 implements Reducer<BooleanWritable, LongWritable,
WritableComparable<?>, Writable> {
190
191 private long numInside = 0;
192 private long numOutside = 0;
193 private JobConf conf; //configuration for accessing the file system
194

//Store job configuration.
195
196 @Override
197 public void configure(JobConf job) {
198     conf = job;
199 }
```

```

//Accumulate number of points inside/outside results from the mappers.
//Parameters:
//isInside - Is the points inside?
//values - An iterator to a list of point counts
//output - dummy, not used here.
//reporter -
207
208     public void reduce(BooleanWritable isInside,
209                         Iterator<LongWritable> values,
210                         OutputCollector<WritableComparable<?>, Writable>
211                         output,
212                         Reporter reporter) throws IOException {
213         if (isInside.get()) {
214             for(; values.hasNext(); numInside += values.next().get());
215         } else {
216             for(; values.hasNext(); numOutside += values.next().get());
217         }
218     }

Reduce task done, write output to a file.
221
222     @Override
223     public void close() throws IOException {
224         //write output to a file
225         Path outDir = new Path(TMP_DIR, "out");
226         Path outFile = new Path(outDir, "reduce-out");
227         FileSystem fileSys = FileSystem.get(conf);
228         SequenceFile.Writer writer = SequenceFile.createWriter(fileSys, conf,
229                     outFile, LongWritable.class, LongWritable.class,
230                     CompressionType.NONE);
231         writer.append(new LongWritable(numInside), new
232                     LongWritable(numOutside));
233         writer.close();
234     }

```

PiEstimator

```
//Run a map/reduce job for estimating Pi.  
//Returns:  
//the estimated value of Pi  
240  
241 public static BigDecimal estimate(int numMaps, long numPoints, JobConf  
jobConf  
242     ) throws IOException {  
243     //setup job conf  
244     jobConf.setJobName(PiEstimator.class.getSimpleName());  
245  
246     jobConf.setInputFormat(SequenceFileInputFormat.class);  
247  
248     jobConf.setOutputKeyClass(BooleanWritable.class);  
249     jobConf.setOutputValueClass(LongWritable.class);  
250     jobConf.setOutputFormat(SequenceFileOutputFormat.class);  
251  
252     jobConf.setMapperClass(PiMapper.class);  
253     jobConf.setNumMapTasks(numMaps);  
254  
255     jobConf.setReducerClass(PiReducer.class);  
256     jobConf.setNumReduceTasks(1);  
257  
258     // turn off speculative execution, because DFS doesn't handle  
259     // multiple writers to the same file.  
260     jobConf.setSpeculativeExecution(false);  
261  
262     //setup input/output directories  
263     final Path inDir = new Path(TMP_DIR, "in");  
264     final Path outDir = new Path(TMP_DIR, "out");  
265     FileInputFormat.setInputPaths(jobConf, inDir);  
266     FileOutputFormat.setOutputPath(jobConf, outDir);  
267  
268     final FileSystem fs = FileSystem.get(jobConf);  
269     if(fs.exists(TMP_DIR)) {  
270         throw new IOException("Tmp directory " + fs.makeQualified(TMP_DIR)  
+ " already exists. Please remove it first.");  
271     }  
272     if (!fs.mkdirs(inDir)) {  
273         throw new IOException("Cannot create input directory " + inDir);  
274     }  
275     try {  
276         //generate an input file for each map task
```

```
279     for(int i=0; i < numMaps; ++i) {  
280         final Path file = new Path(inDir, "part"+i);  
281         final LongWritable offset = new LongWritable(i * numPoints);  
282         final LongWritable size = new LongWritable(numPoints);  
283         final SequenceFile.Writer writer = SequenceFile.createWriter(  
284             fs, jobConf, file,  
285             LongWritable.class, LongWritable.class, CompressionType.NONE);  
286         try {  
287             writer.append(offset, size);  
288         } finally {  
289             writer.close();  
290         }  
291         System.out.println("Wrote input for Map #" + i);  
292     }  
293  
294     //start a map/reduce job  
295     System.out.println("Starting Job");  
296     final long startTime = System.currentTimeMillis();  
297     JobClient.runJob(jobConf);  
298     final double duration = (System.currentTimeMillis() - startTime)/1000.0;  
299     System.out.println("Job Finished in " + duration + " seconds");  
300  
301     //read outputs  
302     Path inFile = new Path(outDir, "reduce-out");  
303     LongWritable numInside = new LongWritable();  
304     LongWritable numOutside = new LongWritable();  
305     SequenceFile.Reader reader = new SequenceFile.Reader(fs, inFile, jobConf);  
306     try {  
307         reader.next(numInside, numOutside);  
308     } finally {  
309         reader.close();  
310     }  
311  
312     //compute estimated value  
313     return BigDecimal.valueOf(4).setScale(20)  
314         .multiply(BigDecimal.valueOf(numInside.get()))  
315         .divide(BigDecimal.valueOf(numMaps))  
316         .divide(BigDecimal.valueOf(numPoints));  
317     } finally {  
318         fs.delete(TMP_DIR, true);  
319     }  
320 }
```

PiEstimator

```
//Parse arguments and then runs a map/reduce job. Print output in standard out.  
//Returns:  
//a non-zero if there is an error. Otherwise, return 0.  
327  
328 public int run(String[] args) throws Exception {  
329     if (args.length != 2) {  
330         System.err.println("Usage: "+getClass().getName()+" <nMaps> <nSamples>");  
331         ToolRunner.printGenericCommandUsage(System.err);  
332         return -1;  
333     }  
334  
335     final int nMaps = Integer.parseInt(args[0]);  
336     final long nSamples = Long.parseLong(args[1]);  
337  
338     System.out.println("Number of Maps = " + nMaps);  
339     System.out.println("Samples per Map = " + nSamples);  
340  
341     final JobConf jobConf = new JobConf(getConf(), getClass());  
342     System.out.println("Estimated value of Pi is "  
343             + estimate(nMaps, nSamples, jobConf));  
344     return 0;  
345 }
```



```
//main method for running it as a stand alone command.  
349  
350 public static void main(String[] argv) throws Exception {  
351     System.exit(ToolRunner.run(null, new PiEstimator(), argv));  
352 }  
353}
```

PiEstimator

```

hadoop@acloud01:/opt/hadoop-1.0.0$ hadoop jar hadoop-examples-1.0.0.jar pi 6 1000
Warning: $HADOOP_HOME is deprecated.

Number of Maps = 6
Samples per Map = 1000
Wrote input for Map #0
Wrote input for Map #1
Wrote input for Map #2
Wrote input for Map #3
Wrote input for Map #4
Wrote input for Map #5
Starting Job
13/05/14 15:06:18 INFO mapred.FileInputFormat: Total input paths to process : 6
13/05/14 15:06:18 INFO mapred.JobClient: Running job: job_201305140112_0002
13/05/14 15:06:19 INFO mapred.JobClient: map 0% reduce 0%
13/05/14 15:06:32 INFO mapred.JobClient: map 66% reduce 0%
13/05/14 15:06:38 INFO mapred.JobClient: map 100% reduce 0%
13/05/14 15:06:41 INFO mapred.JobClient: map 100% reduce 22%
13/05/14 15:06:50 INFO mapred.JobClient: map 100% reduce 100%
13/05/14 15:06:55 INFO mapred.JobClient: Job complete: job_201305140112_0002
13/05/14 15:06:55 INFO mapred.JobClient: Counters: 30
13/05/14 15:06:55 INFO mapred.JobClient: Job Counters
13/05/14 15:06:55 INFO mapred.JobClient: Launched reduce tasks=1
13/05/14 15:06:55 INFO mapred.JobClient: SLOTS_MILLIS_MAPS=32176
13/05/14 15:06:55 INFO mapred.JobClient: Total time spent by all reduces waiting after reserving slots (ms)=0
13/05/14 15:06:55 INFO mapred.JobClient: Total time spent by all maps waiting after reserving slots (ms)=0
13/05/14 15:06:55 INFO mapred.JobClient: Launched map tasks=6
13/05/14 15:06:55 INFO mapred.JobClient: Data-local map tasks=6
13/05/14 15:06:55 INFO mapred.JobClient: SLOTS_MILLIS_REDUCES=16216
13/05/14 15:06:55 INFO mapred.JobClient: File Input Format Counters
13/05/14 15:06:55 INFO mapred.JobClient: Bytes Read=708
13/05/14 15:06:55 INFO mapred.JobClient: File Output Format Counters
13/05/14 15:06:55 INFO mapred.JobClient: Bytes Written=97
13/05/14 15:06:55 INFO mapred.JobClient: FileSystemCounters
13/05/14 15:06:55 INFO mapred.JobClient: FILE_BYTES_READ=138
13/05/14 15:06:55 INFO mapred.JobClient: HDFS_BYTES_READ=1440
13/05/14 15:06:55 INFO mapred.JobClient: FILE_BYTES_WRITTEN=151083
13/05/14 15:06:55 INFO mapred.JobClient: HDFS_BYTES_WRITTEN=215
13/05/14 15:06:55 INFO mapred.JobClient: Map-Reduce Framework
13/05/14 15:06:55 INFO mapred.JobClient: Map output materialized bytes=168
13/05/14 15:06:55 INFO mapred.JobClient: Map input records=6
13/05/14 15:06:55 INFO mapred.JobClient: Reduce shuffle bytes=140
13/05/14 15:06:55 INFO mapred.JobClient: Spilled Records=24
13/05/14 15:06:55 INFO mapred.JobClient: Map output bytes=108
13/05/14 15:06:55 INFO mapred.JobClient: Total committed heap usage (bytes)=1341587456
13/05/14 15:06:55 INFO mapred.JobClient: CPU time spent (ms)=1860
13/05/14 15:06:55 INFO mapred.JobClient: Map input bytes=144
13/05/14 15:06:55 INFO mapred.JobClient: SPLIT_RAW_BYTES=732
13/05/14 15:06:55 INFO mapred.JobClient: Combine input records=0
13/05/14 15:06:55 INFO mapred.JobClient: Reduce input records=12
13/05/14 15:06:55 INFO mapred.JobClient: Reduce input groups=12
13/05/14 15:06:55 INFO mapred.JobClient: Combine output records=0
13/05/14 15:06:55 INFO mapred.JobClient: Physical memory (bytes) snapshot=1315565568
13/05/14 15:06:55 INFO mapred.JobClient: Reduce output records=0
13/05/14 15:06:55 INFO mapred.JobClient: Virtual memory (bytes) snapshot=16225529856
13/05/14 15:06:55 INFO mapred.JobClient: Map output records=12
Job Finished in 37.665 seconds
Estimated value of Pi is 3.13800000000000000000

```

Can compare the error and performance with variation of input arguments

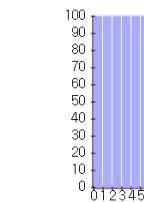
- Number of Map
- Samples per Map

Hadoop job_201305140112_0002 on [acloud01](#)

User: hadoop
 Job Name: PiEstimator
 Job File: https://acloud01:9000/opt/hdfs/mapred/staging/hadoop/.staging/job_201305140112_0002/job.xml
 Submit Host: acloud01
 Submit Host Address: 143.248.152.16
 Job-ACLs: All users are allowed
 Job Setup: Successful
 Status: Succeeded
 Started at: Tue May 14 15:06:18 KST 2013
 Finished at: Tue May 14 15:06:54 KST 2013
 Finished in: 36sec
 Job Cleanup: Successful

Kind	% Complete	Num Tasks	Pending	Running	Complete	Killed	Failed/Killed Task Attempts
map	100.00%	6	0	0	6	0	0 / 0
reduce	100.00%	1	0	0	1	0	0 / 0

Map Completion Graph - [close](#).



Reduce Completion Graph - [close](#).



Graph reversal

- Given a directed graph as an adjacency list:
src1: dest11, dest12, ...
src2: dest21, dest22, ...
Ex) $11 \rightarrow 2, 10$
 $7 \rightarrow 8, 11$
 $8 \rightarrow 9$
...
- Construct the graph in which all the links are reversed
dest1: src11, src21, src31, ...
dest2: src12, src22, src32, ...
Ex) $2 \leftarrow 11$
 $10 \leftarrow 3, 11$
 $9 \leftarrow 8, 11$

Search Application Pseudo Code

- Input: (line number, line) records
- Output: line number whose line value is same as pattern
- Map
 - if(line matches pattern)
 output(line)
- Reduce
 - Identity function

Sort Application Pseudo Code

- Input: (key[word], value[]) record
- Output: sorted record order by key[word]
- Map
 - Identity function
- Reduce
 - Identity function
- Partitioning function
 - Define partitioning function $H()$ which satisfy if $\text{key1} < \text{key2}$ then $H(\text{key1}) < H(\text{key2})$

Integration Pseudo Code

- Input: (start, end) which define the range of Integration
- Output: Integration result
- Map

```
def map(start, end)
    sum = 0
    for(x=start; x<end; x+=step) :
        sum += f(x) * step
        output("", sum)
```

- Reduce

```
def reduce(key, values) :
    output(key, sum(values))
```

Lab 13-2 Requirements

- Implement and test hadoop applications which is introduced in Lab 6
 - Wordcount
 - Inverted Index