电子科技大学

课 程 报 告

词频统计

学　　院：信息与软件工程学院

小组成员 1：201821090124 徐增

学生姓名 2：201822090316 杨庆

学生姓名 3：201852090632 张晟铭

学生姓名 4：2015220304002 敬鑫

学生姓名 5：201822090313 邹洋

指导教师：林迪

目录

MapReduce 及 Spark 词频统计

1　实验内容

（1）安装部署 Hadoop、HDFS、MapReduce、Spark，配置分布式开发环境；

（2）分别在 MapReduce 及 Spark 执行词频统计，；

（3）比较不同文本大小在 MapReduce 及 Spark 执行效率异同；

（4）从软件体系架构角度解释分析实验结果。

2　实验环境

（1）操作系统：Ubuntu16.04；

（2）系统内存：24G；

（3）小组分工：

姓名	学号	任务
徐增	201821090124	实验结果分析，Spark 实验
杨庆	201822090316	环境搭建，MapReduce 实验
张晟铭	201852090632	问题解决，实验分析
敬鑫	2015220304002	资料收集，撰写 PPT
邹洋	201822090313	实验汇总，撰写实验报告

3　Hadoop 介绍

3.1　Hadoop 产生背景

Hadoop 最早起源于 Nutch。Nutch 的设计目标是构建一个大型的全网搜索引擎，包括网页抓取、索引、查询等功能，但随着抓取网页数量的增加，遇到了严重的可扩展性问题——如何解决数十亿网页的存储和索引问题。

2003 年、2004 年谷歌发表的两篇论文为该问题提供了可行的解决方案。

（1）分布式文件系统（GFS），可用于处理海量网页的存储

（2）分布式计算框架（MapReduce），可用于处理海量网页的索引计算问题。

Nutch 的开发人员完成了相应的开源实现 HDFS 和 MapReduce，并从 Nutch 中剥离成为独立项目 Hadoop，到 2008 年 1 月，Hadoop 成为 Apache 顶级项目，迎来了它的快速发展期。

3.2 Hadoop 简介

Hadoop 是 Apache 旗下的一套开源软件平台，Hadoop 是利用服务器集群对数据进行存储，根据用户的自定义业务逻辑，对海量数据进行分布式计算。广义上来说，Hadoop 通常是指一个更广泛的概念——Hadoop 生态圈。

Hadoop 解决了海量数据的存储（HDFS）、海量数据的技术（MapReduce）、资源调度（YARN）等问题。其中重点组件包括：

（1）HDFS：分布式文件系统；

（2）MapReduce：分布式运算程序开发框架；

（3）YARN:资源调度系统。

（4）ZOOKEEPER：分布式协调服务基础组件；

（5）HIVE：SQL 数据仓库工具；

（6）HBASE：基于 Hadoop 的分布式海量数据库；

（7）Sqoop：数据迁移工具；

（8）Flume：日志数据采集框架；

3.3 Hadoop 架构

3.3.1 分布式架构简介

1.单机的问题

（1）存储能力有限；

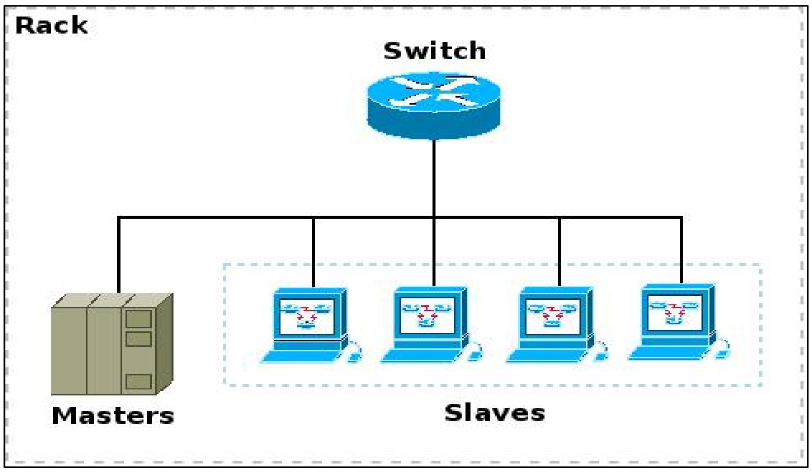
（2）计算能力有限；

（3）有单点故障等。

2.分布式架构解决了单机的问题

3.经典分布式主从架构（Master-Slave）

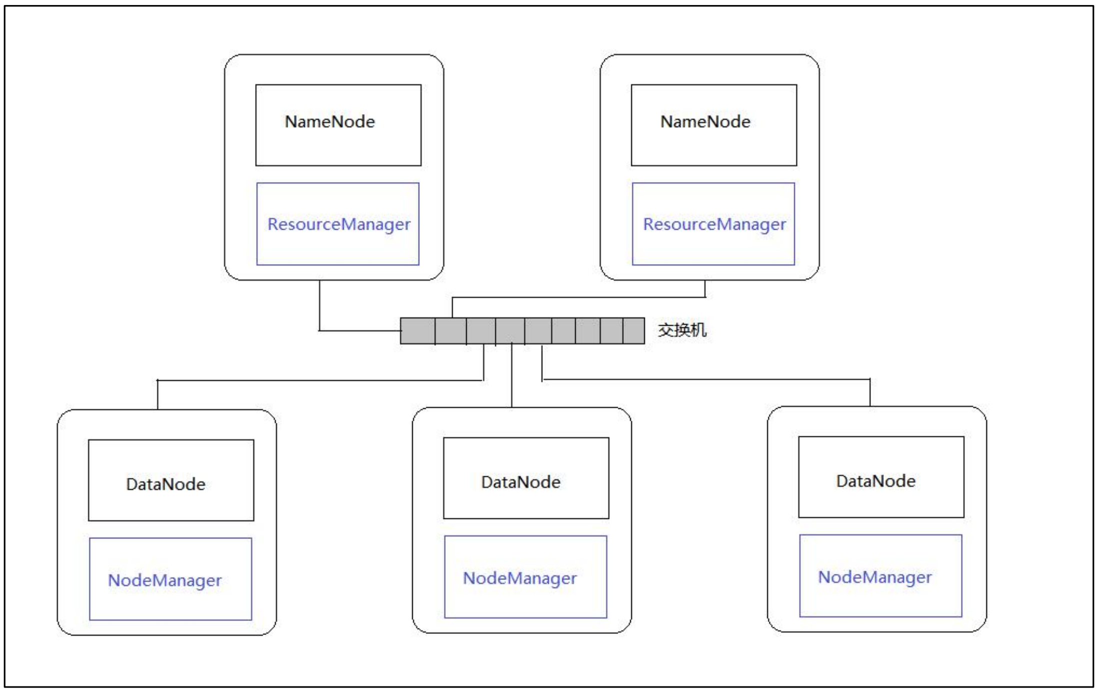
4.Master 负责管理，且可以有多个，防止单点故障的发生。Slave 负责干活，Slave 有多个，并且可以动态的添加或移除。



3.3.2 Hadoop2.0

（1）HDFS ：NameNode（老大） DataNode（小弟）

（2）YARN ：ResourceManager（老大） NodeManager（小组长）



3.3.3 伪分布式架构

（1）NameNode：HDFS 的管理节点，负责 DataNode 的管理和元数据管理；

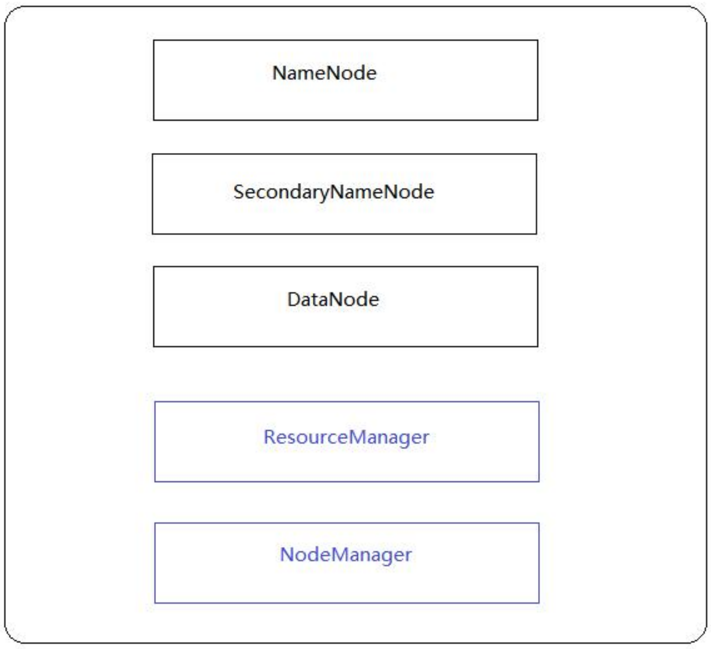
（2）SecondaryNameNode：NameNode 的一个助理，帮助 NameNode 管理元数据，防止元数据丢失；

（3）DataNode：负责数据存储；

（4）ResourceManager：YARN 的管理节点，负责 NodeManager 的管理、任务

调度等；

（5）NodeManager：YARN 的节点管理器，负责向 ResourceManager 汇报当前节点的状态和启动计算任务进程（YarnChild）并监控 YarnChild。



4 环境搭建

4.1 ubuntu16.04 配置 Hadoop 伪分布式

4.1.1 实验环境

（1）操作系统：Ubuntu16.04

（2）Java 环境：jdk1.8.0_181

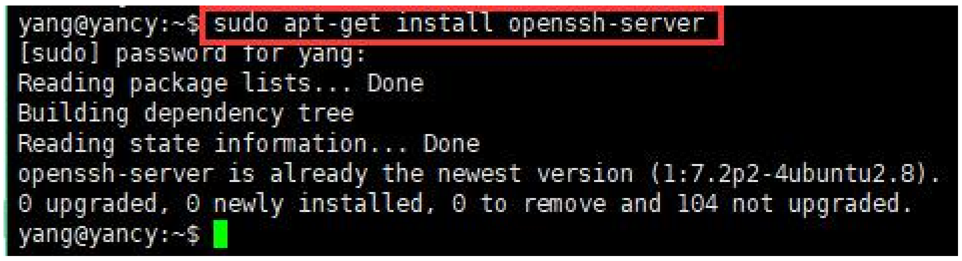
（3）Hadoop 版本：hadoop-2.7.6

（4）Scala 版本：scala-2.12.8

（5）Spark 版本：spark-2.4.1-bin-hadoop2.7

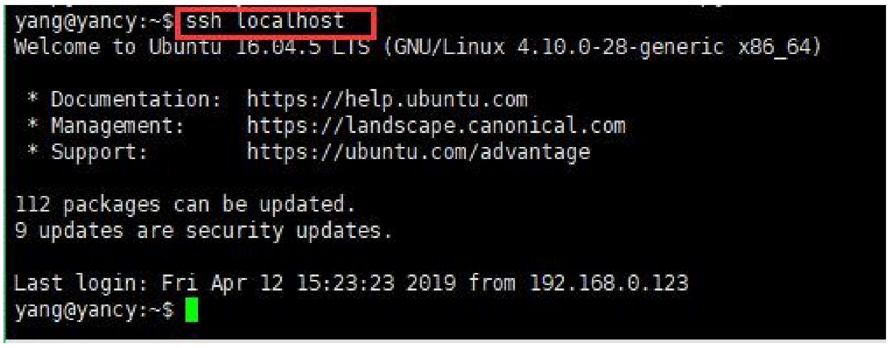
4.1.2 SSH 免密码登录

（1）输入：sudo apt-get install openssh-server，安装 SSH server；



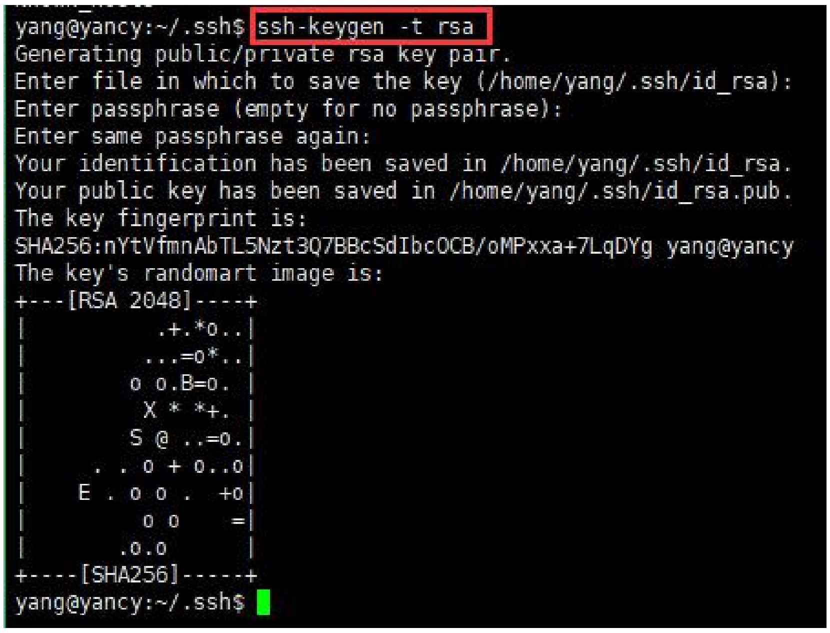
```
yang@yancy:~$ sudo apt-get install openssh-server
[sudo] password for yang:
Reading package lists... Done
Building dependency tree
Reading state information... Done
openssh-server is already the newest version (1:7.2p2-4ubuntu2.8).
0 upgraded, 0 newly installed, 0 to remove and 104 not upgraded.
yang@yancy:~$
```

（2）输入：cd ~/.ssh/，如果没法进入该目录，执行一次 ssh localhost；



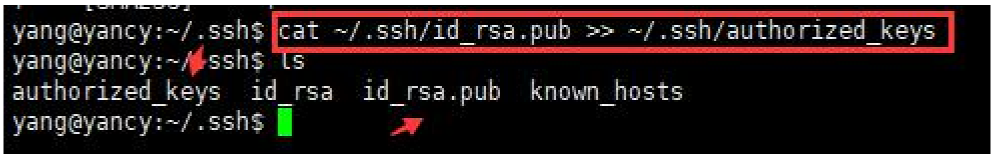
```
yang@yancy:~$ ssh localhost
Welcome to Ubuntu 16.04.5 LTS (GNU/Linux 4.10.0-28-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

112 packages can be updated.
9 updates are security updates.

Last login: Fri Apr 12 15:23:23 2019 from 192.168.0.123
yang@yancy:~$
```

（3）输入：ssh-keygen -t rsa，三次回车后，该目录下将会产生 id_rsa，id_rsa.pub 文件；



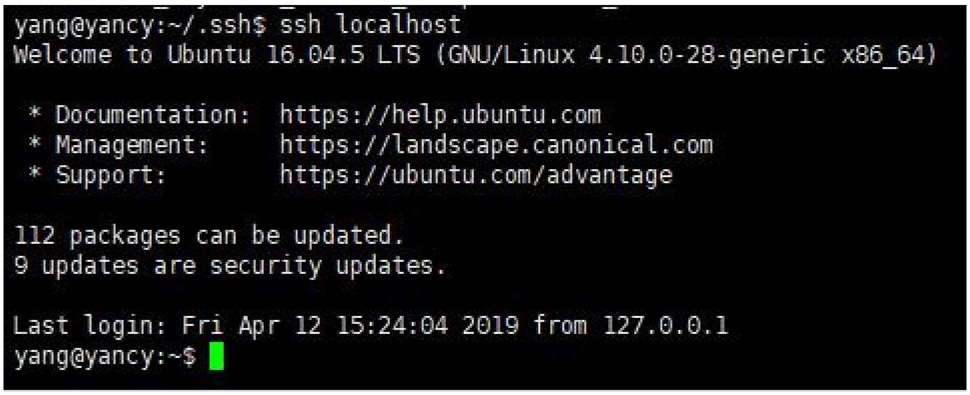
```
yang@yancy:~/.ssh$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/yang/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/yang/.ssh/id_rsa.
Your public key has been saved in /home/yang/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:nYtVfmnAbTL5Nzt3Q7BBcSdIbcOCB/oMPxxa+7LqDYg yang@yancy
The key's randomart image is:
+---[RSA 2048]----+
|          .+.*o..|
|         ...=o*..|
|        o o.B=o. |
|         X * *+. |
|        S @ ..=o.|
|     . . o + o..o|
|    E . o o .  +o|
|         o o    =|
|       .o.o      |
+----[SHA256]-----+
yang@yancy:~/.ssh$
```

（4）输入：cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys 加入授权；



```
yang@yancy:~/.ssh$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
yang@yancy:~/.ssh$ ls
authorized_keys  id_rsa  id_rsa.pub  known_hosts
yang@yancy:~/.ssh$
```

（5）输入：ssh localhost，如果不提示输入密码则 SSH 无密登陆配置成功；



```
yang@yancy:~/.ssh$ ssh localhost
Welcome to Ubuntu 16.04.5 LTS (GNU/Linux 4.10.0-28-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

112 packages can be updated.
9 updates are security updates.

Last login: Fri Apr 12 15:24:04 2019 from 127.0.0.1
yang@yancy:~$
```

4.1.3 安装 java1.8.0_181

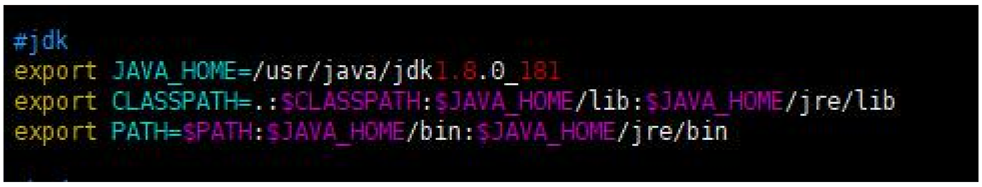
（1）https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html 下载 jdk-8u181-linux-x64.tar.gz，输入：sudo tar zxvf jdk-8u201-linux-x64.tar.gz -C /usr/java/jdk1.8.0_181 将 jdk-8u181-linux-x64.tar.gz 解压到/usr/java/目录下；

（2）输入：sudo vim ~/.bashrc 配置环境变量，在最后添加三行：

export JAVA_HOME=/usr/java/jdk1.8.0_181

export CLASSPATH=.:$CLASSPATH:$JAVA_HOME/lib:$JAVA_HOME/jre/lib

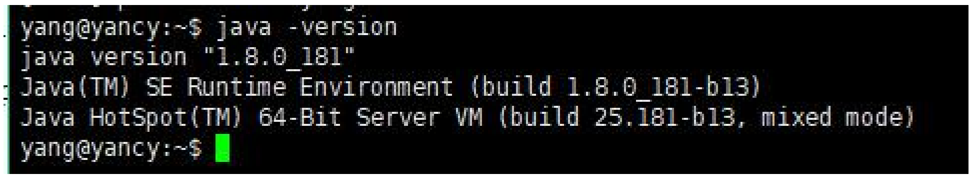
export PATH=$PATH:$JAVA_HOME/bin:$JAVA_HOME/jre/bin



```
#jdk
export JAVA_HOME=/usr/java/jdk1.8.0_181
export CLASSPATH=.:$CLASSPATH:$JAVA_HOME/lib:$JAVA_HOME/jre/lib
export PATH=$PATH:$JAVA_HOME/bin:$JAVA_HOME/jre/bin
```

（3）输入：source ~/.bashrc，使新配置的环境变量生效；

（4）输入：java -version，查看 Java 版本，检测是否安装成功；



```
yang@yancy:~$ java -version
java version "1.8.0_181"
Java(TM) SE Runtime Environment (build 1.8.0_181-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.181-b13, mixed mode)
yang@yancy:~$
```

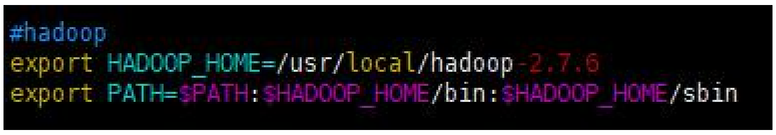
4.1.4 安装 hadoop-2.7.6

（1）在 https://mirrors.tuna.tsinghua.edu.cn/apache/hadoop/common/ 下载 hadoop-2.7.6.tar.gz，输入：sudo tar zxvf hadoop-2.7.6.tar.gz -C /usr/local/hadoop-2.7.6 将 hadoop-2.7.6.tar.gz 解压到/usr/local/目录下；

（2）输入：sudo vim ~/.bashrc 添加如下两行，然后输入 source ~./bashrc；

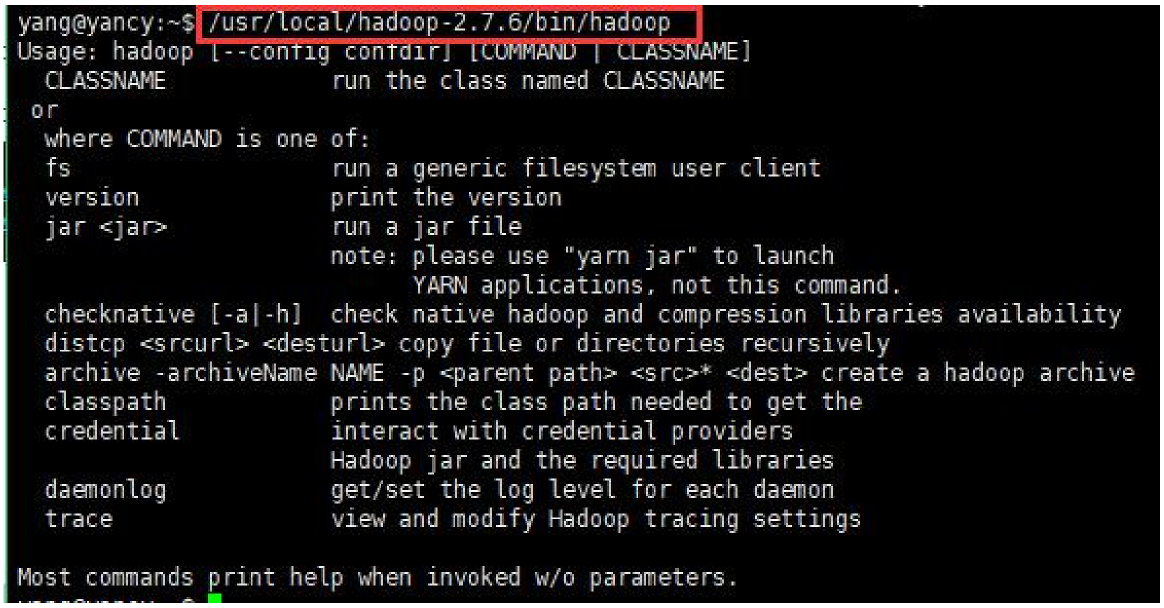
export HADOOP_HOME=/usr/local/hadoop-2.7.6

export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin



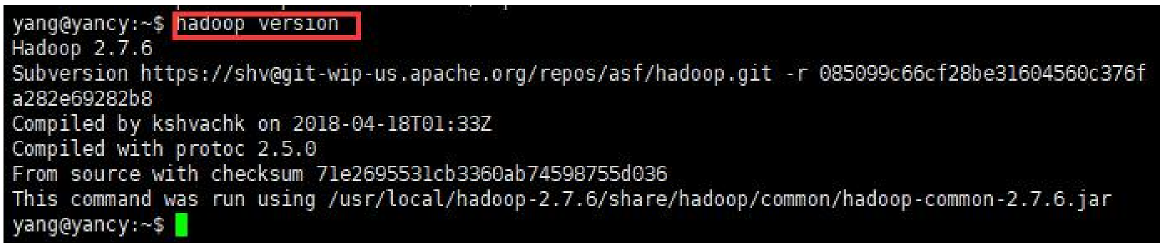
```
#hadoop
export HADOOP_HOME=/usr/local/hadoop-2.7.6
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```

（3）输入：/usr/local/hadoop-2.7.6/bin/hadoop 查看 hadoop 命令相关使用信息；



```
yang@yancy:~$ /usr/local/hadoop-2.7.6/bin/hadoop
Usage: hadoop [--config confdir] [COMMAND | CLASSNAME]
  CLASSNAME            run the class named CLASSNAME
 or
  where COMMAND is one of:
  fs                   run a generic filesystem user client
  version              print the version
  jar <jar>            run a jar file
                       note: please use "yarn jar" to launch
                             YARN applications, not this command.
  checknative [-a|-h]  check native hadoop and compression libraries availability
  distcp <srcurl> <desturl> copy file or directories recursively
  archive -archiveName NAME -p <parent path> <src>* <dest> create a hadoop archive
  classpath            prints the class path needed to get the
  credential           interact with credential providers
                       Hadoop jar and the required libraries
  daemonlog            get/set the log level for each daemon
  trace                view and modify Hadoop tracing settings

Most commands print help when invoked w/o parameters.
```

（4）输入：hadoop version 查看版本。



```
yang@yancy:~$ hadoop version
Hadoop 2.7.6
Subversion https://shv@git-wip-us.apache.org/repos/asf/hadoop.git -r 085099c66cf28be31604560c376f
a282e69282b8
Compiled by kshvachk on 2018-04-18T01:33Z
Compiled with protoc 2.5.0
From source with checksum 71e2695531cb3360ab74598755d036
This command was run using /usr/local/hadoop-2.7.6/share/hadoop/common/hadoop-common-2.7.6.jar
yang@yancy:~$
```

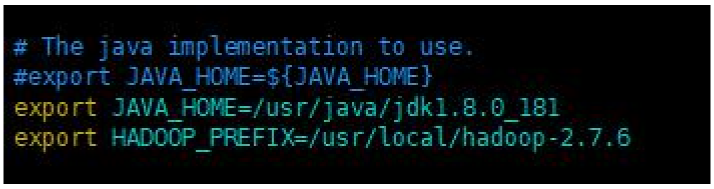
4.1.5 伪分布式配置

Hadoop 可以在单节点上以伪分布式的方式运行，Hadoop 进程以分离的 Java 进程来运行，节点既作为 NameNode 也作为 DataNode，同时，读取的是 HDFS 中的文件。Hadoop 的配置文件位于 /usr/local/hadoop-2.7.6/etc/hadoop/ 中，伪分布式需要修改 2 个配置文件 core-site.xml 和 hdfs-site.xml 。Hadoop 的配置文件是 xml 格式，每个配置以声明 property 的 name 和 value 的方式来实现。此处我们另外修改了配置文件。

（1）JAVA_HOME 位于/usr/java/jdk1.8.0_181，Hadoop 在/usr/local/hadoop-2.7.6。

输入：sudo vim /usr/local/hadoop-2.7.6/etc/hadoop/hadoop-env.sh 添加两行参数：

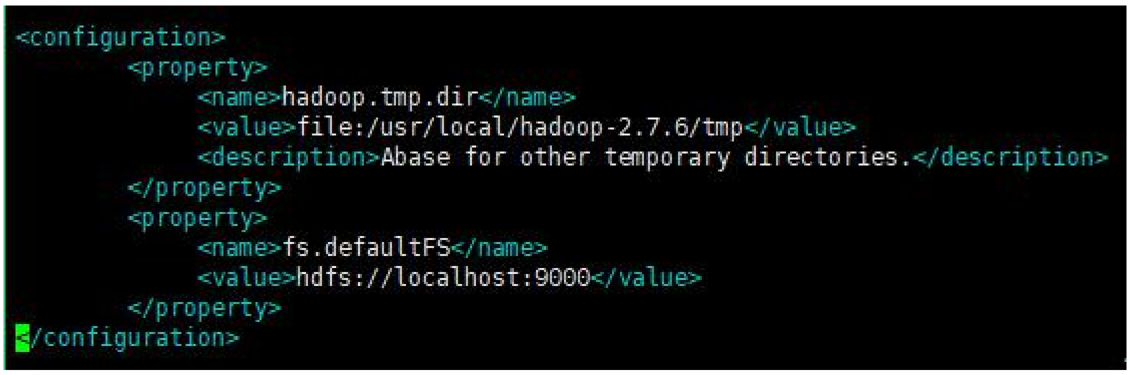
```
export JAVA_HOME=/usr/java/jdk1.8.0_181
export HADOOP_PREFIX=/usr/local/hadoop-2.7.6
```



```
# The java implementation to use.
#export JAVA_HOME=${JAVA_HOME}
export JAVA_HOME=/usr/java/jdk1.8.0_181
export HADOOP_PREFIX=/usr/local/hadoop-2.7.6
```

（2）输入：sudo vim /usr/local/hadoop-2.7.6/etc/hadoop/core-site.xml 修改 core-site.xml 文件添加如下内容：

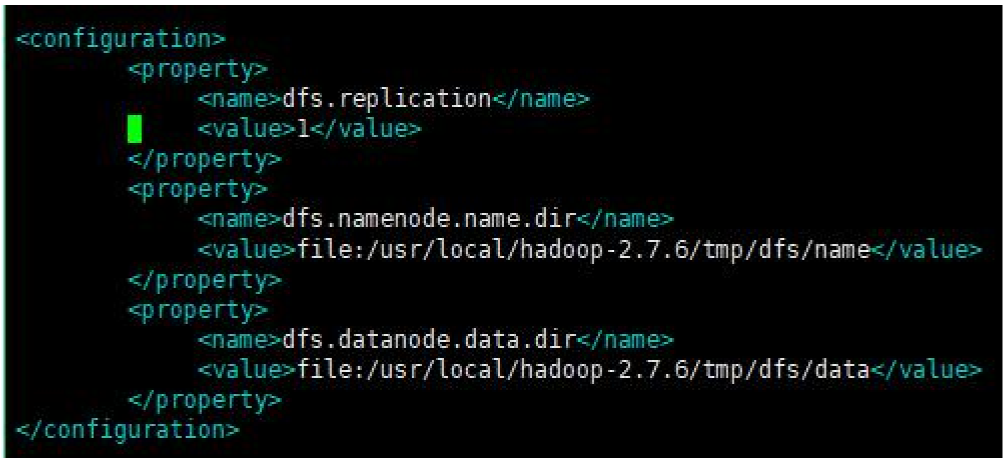
```
<configuration>
        <!-- 配置 Hadoop 运行时产生数据的存储目录，不是临时的数据 -->
        <property>
              <name>hadoop.tmp.dir</name>
              <value>file:/usr/local/hadoop-2.7.6/tmp</value>
              <description>Abase for other temporary directories.</description>
        </property>
        <!-- 配置 hdfs 的 Namenode（老大）的地址 -->
        <property>
              <name>fs.defaultFS</name>
              <value>hdfs://localhost:9000</value>
        </property>
</configuration>
```



```
<configuration>
        <property>
             <name>hadoop.tmp.dir</name>
             <value>file:/usr/local/hadoop-2.7.6/tmp</value>
             <description>Abase for other temporary directories.</description>
        </property>
        <property>
             <name>fs.defaultFS</name>
             <value>hdfs://localhost:9000</value>
        </property>
</configuration>
```

（3）输入：sudo vim /usr/local/hadoop-2.7.6/etc/hadoop/hdfs-site.xml 修改配置文件 hdfs-site.xml 添加如下内容：

```
<configuration>
        <!-- 指定 HDFS 存储数据的副本数量 -->
        <property>
             <name>dfs.replication</name>
             <value>1</value>
        </property>
        <property>
             <name>dfs.namenode.name.dir</name>
             <value>file:/usr/local/hadoop-2.7.6/tmp/dfs/name</value>
        </property>
        <property>
             <name>dfs.datanode.data.dir</name>
             <value>file:/usr/local/hadoop-2.7.6/tmp/dfs/data</value>
        </property>
</configuration>
```



```
<configuration>
        <property>
             <name>dfs.replication</name>
             <value>1</value>
        </property>
        <property>
             <name>dfs.namenode.name.dir</name>
             <value>file:/usr/local/hadoop-2.7.6/tmp/dfs/name</value>
        </property>
        <property>
             <name>dfs.datanode.data.dir</name>
             <value>file:/usr/local/hadoop-2.7.6/tmp/dfs/data</value>
        </property>
</configuration>
```

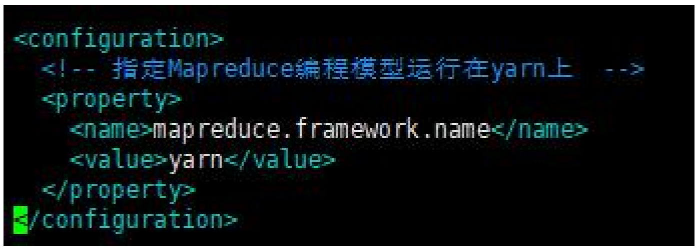
（4）将 mapred-site.xml.template 重命名为 mapred-site.xml，

输入：sudo mv mapred-site.xml.template mapred-site.xml

然后输入：sudo vim /usr/local/hadoop-2.7.6/etc/hadoop/mapred-site.xml 修改配置文件 mapred-site.xml 添加如下内容：

```
<configuration>
  <!-- 指定 Mapreduce 编程模型运行在 yarn 上 -->
  <property>
```

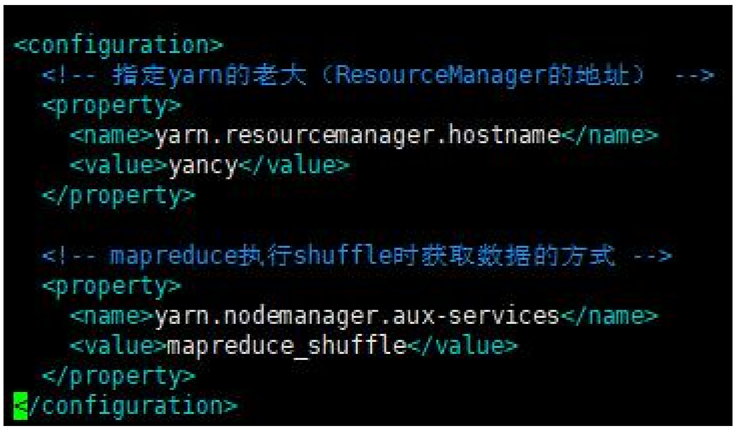
```
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
      </property>
    </configuration>
```



```
<configuration>
  <!-- 指定Mapreduce编程模型运行在yarn上  -->
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

（5）输入：sudo vim /usr/local/hadoop-2.7.6/etc/hadoop/yarn-site.xml 修改配置文件 yarn-site.xml 添加如下内容：

```
<configuration>
  <!-- 指定 yarn 的老大（ResourceManager 的地址）  -->
  <property>
    <name>yarn.resourcemanager.hostname</name>
    <value>yancy</value>
  </property>

  <!-- mapreduce 执行 shuffle 时获取数据的方式  -->
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
</configuration>
```



```
<configuration>
  <!-- 指定yarn的老大（ResourceManager的地址） -->
  <property>
    <name>yarn.resourcemanager.hostname</name>
    <value>yancy</value>
  </property>

  <!-- mapreduce执行shuffle时获取数据的方式 -->
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
</configuration>
```

4.1.6 运行 Hadoop

Hadoop 的运行方式是由配置文件决定的（运行 Hadoop 时会读取配置文件），因此如果需要从伪分布式模式切换回非分布式模式，需要删除 core-site.xml 中的配置项。此外，伪分布式虽然只需要配置 fs.defaultFS 和 dfs.replication 就可以运行（可参考官方教程），不过若没有配置 hadoop.tmp.dir 参数，则默认使用的临时目录为 /tmp/hadoo-hadoop，而这个目录在重启时有可能被系统清理掉，导致必须重新执行 format 才行。所以我们进行了设置，同时也指定 dfs.namenode.name.dir 和 dfs.datanode.data.dir，否则在接下来的步骤中可能会出错。

在 Hadoop 安装包目录下有几个比较重要的目录：

1）sbin：启动或停止 Hadoop 相关服务的脚本；

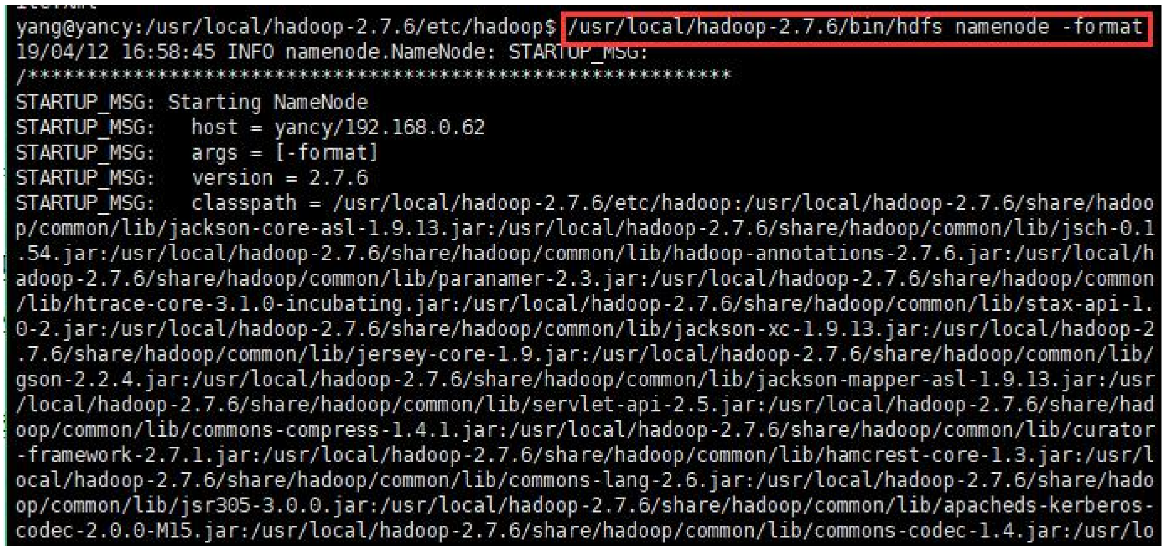
2）bin：对 Hadoop 相关服务（HDFS,YARN）进行操作的脚本；

3）etc：Hadoop 的配置文件目录；

4）share：Hadoop 的依赖 jar 包和文档，文档可以被删掉；

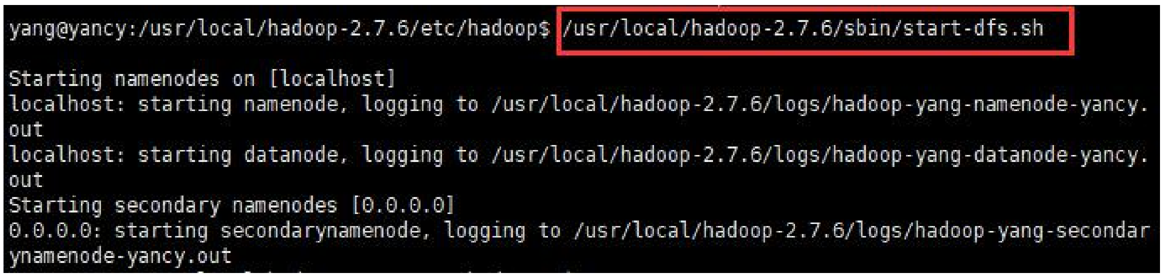
5）lib：Hadoop 的本地库（对数据进行压缩解压缩功能的）。

（1）输入：/usr/local/hadoop-2.7.6/bin/hdfs namenode -format 执行 NameNode 的格式化；



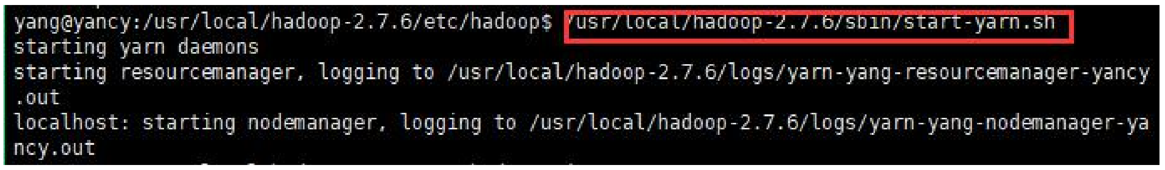
```
yang@yancy:/usr/local/hadoop-2.7.6/etc/hadoop$ /usr/local/hadoop-2.7.6/bin/hdfs namenode -format
19/04/12 16:58:45 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = yancy/192.168.0.62
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 2.7.6
STARTUP_MSG:   classpath = /usr/local/hadoop-2.7.6/etc/hadoop:/usr/local/hadoop-2.7.6/share/hadoo
p/common/lib/jackson-core-asl-1.9.13.jar:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/jsch-0.1
.54.jar:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/hadoop-annotations-2.7.6.jar:/usr/local/h
adoop-2.7.6/share/hadoop/common/lib/paranamer-2.3.jar:/usr/local/hadoop-2.7.6/share/hadoop/common
/lib/htrace-core-3.1.0-incubating.jar:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/stax-api-1.
0-2.jar:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/jackson-xc-1.9.13.jar:/usr/local/hadoop-2
.7.6/share/hadoop/common/lib/jersey-core-1.9.jar:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/
gson-2.2.4.jar:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/jackson-mapper-asl-1.9.13.jar:/usr
/local/hadoop-2.7.6/share/hadoop/common/lib/servlet-api-2.5.jar:/usr/local/hadoop-2.7.6/share/had
oop/common/lib/commons-compress-1.4.1.jar:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/curator
-framework-2.7.1.jar:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/hamcrest-core-1.3.jar:/usr/l
ocal/hadoop-2.7.6/share/hadoop/common/lib/commons-lang-2.6.jar:/usr/local/hadoop-2.7.6/share/hado
op/common/lib/jsr305-3.0.0.jar:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/apacheds-kerberos-
codec-2.0.0-M15.jar:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/commons-codec-1.4.jar:/usr/lo
```

（2）输入：/usr/local/hadoop-2.7.6/sbin/start-dfs.sh 启动 NameNode 和 DataNode 进程，并查看启动结果；



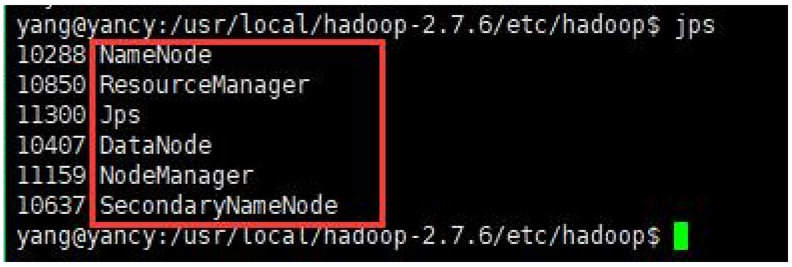
```
yang@yancy:/usr/local/hadoop-2.7.6/etc/hadoop$ /usr/local/hadoop-2.7.6/sbin/start-dfs.sh

Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/local/hadoop-2.7.6/logs/hadoop-yang-namenode-yancy.
out
localhost: starting datanode, logging to /usr/local/hadoop-2.7.6/logs/hadoop-yang-datanode-yancy.
out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop-2.7.6/logs/hadoop-yang-secondar
ynamenode-yancy.out
```

（3）输入：/usr/local/hadoop-2.7.6/sbin/start-yarn.sh 启动 ResourceManager 和 NodeManager；



```
yang@yancy:/usr/local/hadoop-2.7.6/etc/hadoop$ /usr/local/hadoop-2.7.6/sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop-2.7.6/logs/yarn-yang-resourcemanager-yancy
.out
localhost: starting nodemanager, logging to /usr/local/hadoop-2.7.6/logs/yarn-yang-nodemanager-ya
ncy.out
```

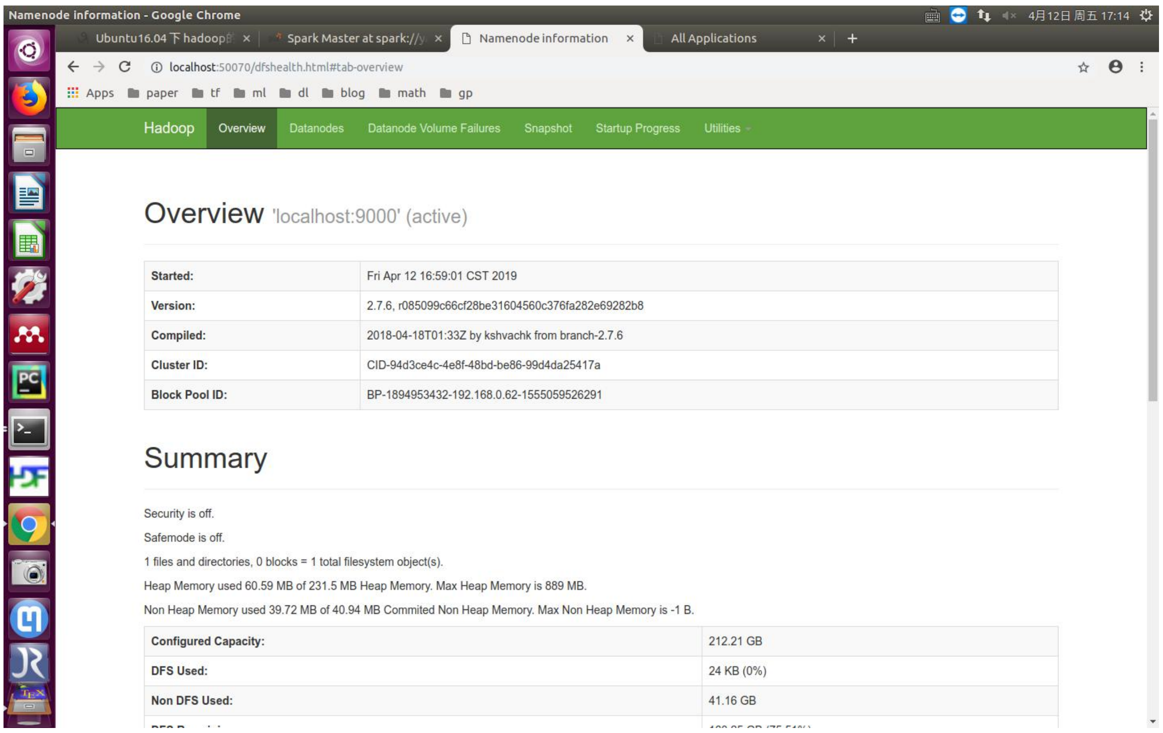
（4）输入：jps，判断是否成功启动，若成功启动则会列出如下进程: “NameNode”、”DataNode”、“SecondaryNameNode”、“ResourceManager”和 NodeManager；



```
yang@yancy:/usr/local/hadoop-2.7.6/etc/hadoop$ jps
10288 NameNode
10850 ResourceManager
11300 Jps
10407 DataNode
11159 NodeManager
10637 SecondaryNameNode
yang@yancy:/usr/local/hadoop-2.7.6/etc/hadoop$
```

（5）访问 HDFS 的管理界面：在浏览器访问 http://localhost:50070 查看

NameNode 和 DataNode 的相关信息，还可以在线查看 HDFS 中的文件；



Overview 'localhost:9000' (active)

Started:	Fri Apr 12 16:59:01 CST 2019
Version:	2.7.6, r085099c66cf28be31604560c376fa282e69282b8
Compiled:	2018-04-18T01:33Z by kshvachk from branch-2.7.6
Cluster ID:	CID-94d3ce4c-4e8f-48bd-be86-99d4da25417a
Block Pool ID:	BP-1894953432-192.168.0.62-1555059526291

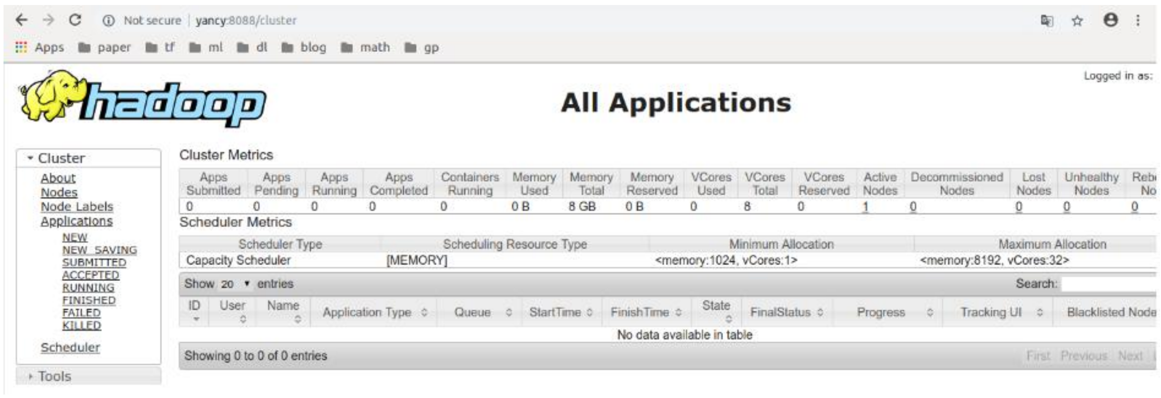
Summary

Security is off.
Safemode is off.
1 files and directories, 0 blocks = 1 total filesystem object(s).
Heap Memory used 60.59 MB of 231.5 MB Heap Memory. Max Heap Memory is 889 MB.
Non Heap Memory used 39.72 MB of 40.94 MB Commited Non Heap Memory. Max Non Heap Memory is -1 B.

Configured Capacity:	212.21 GB
DFS Used:	24 KB (0%)
Non DFS Used:	41.16 GB

Namenode information - Google Chrome
Ubuntu16.04 下 hadoop
Spark Master at spark://
Namenode information
All Applications
4月12日 周五 17:14
localhost:50070/dfshealth.html#tab-overview
Apps paper tf ml dl blog math gp
Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities

（6）访问 YARN 的管理界面：在浏览器访问 http://yancy:8088 查看 Cluster 相关信息。



All Applications

Logged in as:

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
0	0	0	0	0	0 B	8 GB	0 B	0	8	0	1	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:32>

Show 20 entries Search:

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI	Blacklisted Node
No data available in table											

Showing 0 to 0 of 0 entries

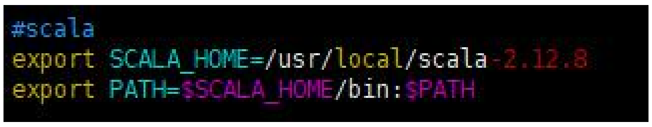
4.2 ubuntu16.04 配置 spark

4.2.1 ubuntu16.04 安装 scala-2.12.8

（1）输入：sudo tar -xzvf scala-2.12.8.tgz -C /usr/local，解压 scala 到/usr/local。网址为：https://www.scala-lang.org/download/；

（2）输入：sudo vim ~/.bashrc，在最后添加下面内容：

```
export SCALA_HOME=/usr/local/scala-2.12.8
export PATH=$SCALA_HOME/bin:$PATH
```



```
#scala
export SCALA_HOME=/usr/local/scala-2.12.8
export PATH=$SCALA_HOME/bin:$PATH
```

（3）输入：source ~/.bashrc，使新配置的环境变量生效；

（4）输入：scala -version 查看版本。

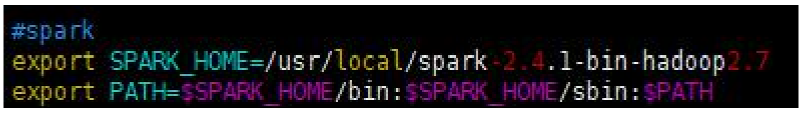
4. 2. 2 ubuntu16. 04 安装 spark-2. 4. 1-bin-hadoop2. 7

（1）输入：sudo tar -zxf spark-2.4.1-bin-hadoop2.7.tgz -C /usr/local，解压下载的 spark 文件。网址为：http://spark.apache.org/downloads.html

（2）输入：sudo vim ~/.bashrc，在最后添加下面内容：

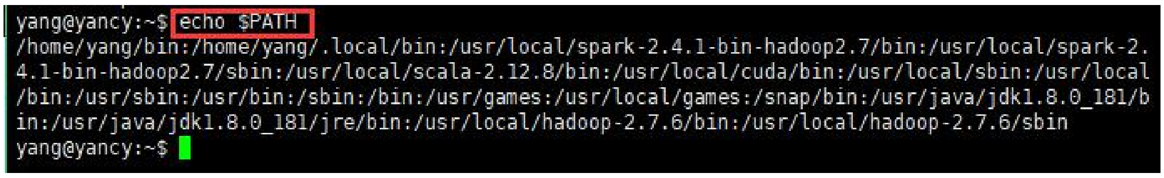
export SPARK_HOME=/usr/local/spark-2.4.1-bin-hadoop2.7

export PATH=$SPARK_HOME/bin:$SPARK_HOME/sbin:$PATH



```
#spark
export SPARK_HOME=/usr/local/spark-2.4.1-bin-hadoop2.7
export PATH=$SPARK_HOME/bin:$SPARK_HOME/sbin:$PATH
```

（3）输入：source ~/.bashrc，使新配置的环境变量生效；



```
yang@yancy:~$ echo $PATH
/home/yang/bin:/home/yang/.local/bin:/usr/local/spark-2.4.1-bin-hadoop2.7/bin:/usr/local/spark-2.
4.1-bin-hadoop2.7/sbin:/usr/local/scala-2.12.8/bin:/usr/local/cuda/bin:/usr/local/sbin:/usr/local
/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/usr/java/jdk1.8.0_181/b
in:/usr/java/jdk1.8.0_181/jre/bin:/usr/local/hadoop-2.7.6/bin:/usr/local/hadoop-2.7.6/sbin
yang@yancy:~$ 
```

（4）拷贝配置文件：

cd /usr/local/spark-2.4.1-bin-hadoop2.7

sudo cp ./conf/spark-env.sh.template ./conf/spark-env.sh

（5）输入：sudo vim /usr/local/spark-2.4.1-bin-hadoop2.7/conf/spark-env.sh，修改配置文件，添加下面一行：

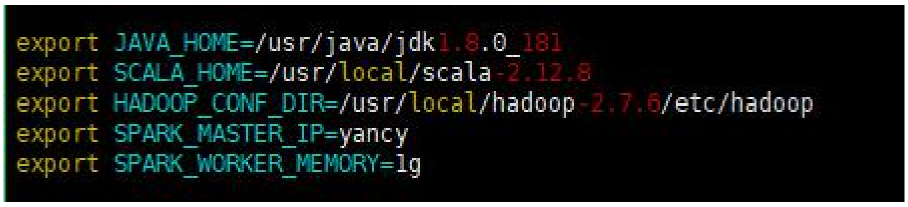
export JAVA_HOME=/usr/java/jdk1.8.0_181

export SCALA_HOME=/usr/local/scala-2.12.8

export HADOOP_CONF_DIR=/usr/local/hadoop-2.7.6/etc/hadoop

export SPARK_MASTER_IP=yancy

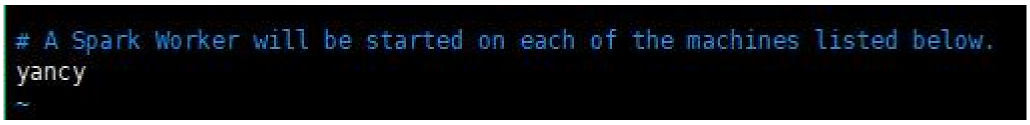
export SPARK_WORKER_MEMORY=1g



```
export JAVA_HOME=/usr/java/jdk1.8.0_181
export SCALA_HOME=/usr/local/scala-2.12.8
export HADOOP_CONF_DIR=/usr/local/hadoop-2.7.6/etc/hadoop
export SPARK_MASTER_IP=yancy
export SPARK_WORKER_MEMORY=1g
```

（6）输 入 ： sudo cp /usr/local/spark-2.4.1-bin-hadoop2.7/conf/slaves.template slaves，将 slaves.template 重命名为 slaves；

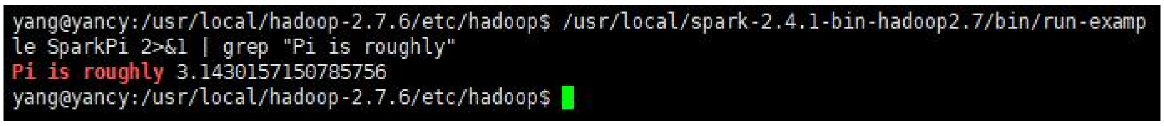
（7）输入：sudo vim /usr/local/spark-2.4.1-bin-hadoop2.7/conf/slaves，将 slaves 中的 localhost 修改为主机名，我的是 yancy。



```
# A Spark Worker will be started on each of the machines listed below.
yancy
~
```

（8）运行简单示例：

/usr/local/spark-2.4.1-bin-hadoop2.7/bin/run-example SparkPi 2>&1 | grep "Pi is roughly"



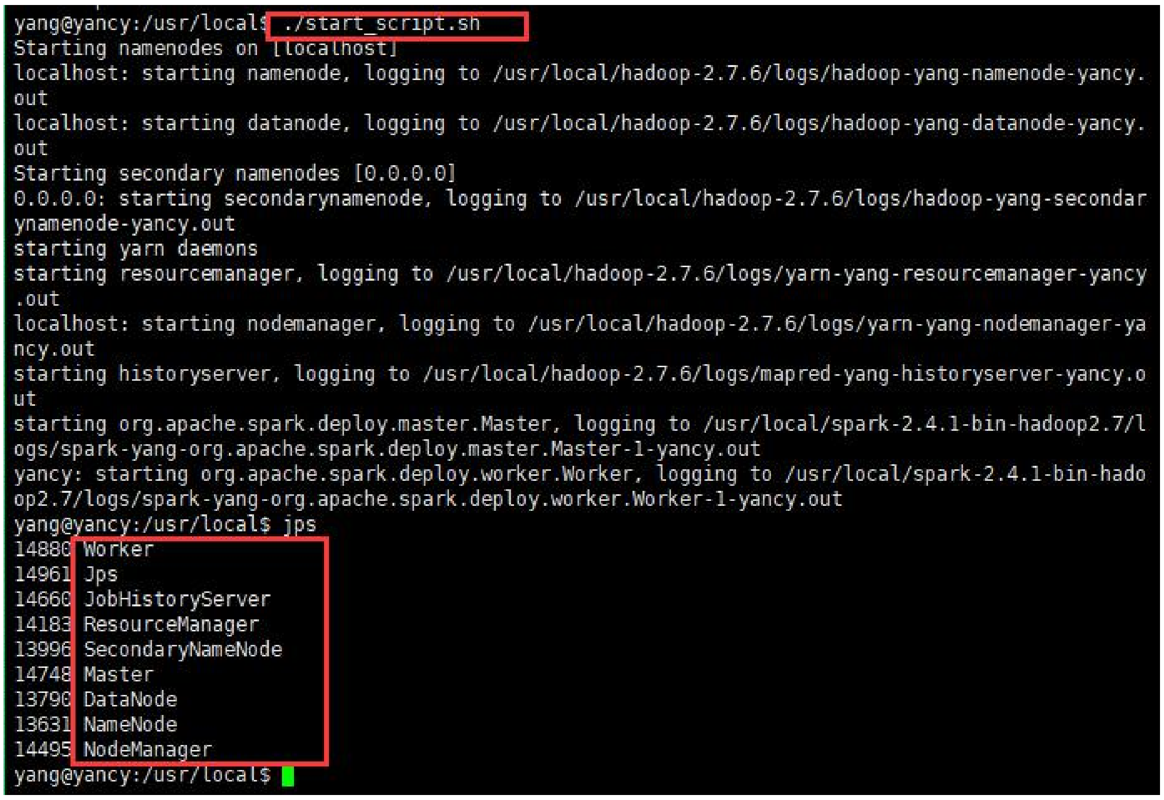
```
yang@yancy:/usr/local/hadoop-2.7.6/etc/hadoop$ /usr/local/spark-2.4.1-bin-hadoop2.7/bin/run-examp
le SparkPi 2>&1 | grep "Pi is roughly"
Pi is roughly 3.1430157150785756
yang@yancy:/usr/local/hadoop-2.7.6/etc/hadoop$
```

（9）输入；sudo chown -R yang:yang spark-2.4.1-bin-hadoop2.7/，修改权限；

（10）输入：/usr/local/spark-2.4.1-bin-hadoop2.7/sbin/start-all.sh，启动 Spark；

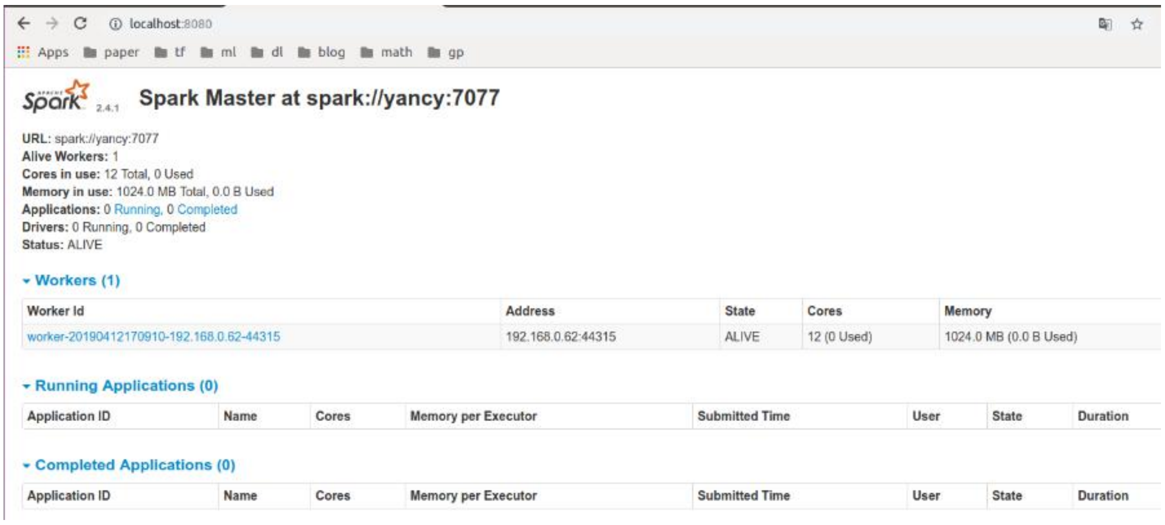
（11）编写启动脚本 start_script.sh 启动 Hadoop 以及 Spark：

```
#!/bin/bash
start-dfs.sh # 启动 Hadoop
start-yarn.sh # 启动 Yarn
mr-jobhistory-daemon.sh start historyserver # 启动历史服务器,以便在 Web 中查看任务运行情况。
/usr/local/spark-2.4.1-bin-hadoop2.7/sbin/start-all.sh # 启动 Spark
```



```
yang@yancy:/usr/local$ ./start_script.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/local/hadoop-2.7.6/logs/hadoop-yang-namenode-yancy.
out
localhost: starting datanode, logging to /usr/local/hadoop-2.7.6/logs/hadoop-yang-datanode-yancy.
out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop-2.7.6/logs/hadoop-yang-secondar
ynamenode-yancy.out
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop-2.7.6/logs/yarn-yang-resourcemanager-yancy
.out
localhost: starting nodemanager, logging to /usr/local/hadoop-2.7.6/logs/yarn-yang-nodemanager-ya
ncy.out
starting historyserver, logging to /usr/local/hadoop-2.7.6/logs/mapred-yang-historyserver-yancy.o
ut
starting org.apache.spark.deploy.master.Master, logging to /usr/local/spark-2.4.1-bin-hadoop2.7/l
ogs/spark-yang-org.apache.spark.deploy.master.Master-1-yancy.out
yancy: starting org.apache.spark.deploy.worker.Worker, logging to /usr/local/spark-2.4.1-bin-hado
op2.7/logs/spark-yang-org.apache.spark.deploy.worker.Worker-1-yancy.out
yang@yancy:/usr/local$ jps
14880 Worker
14961 Jps
14660 JobHistoryServer
14183 ResourceManager
13996 SecondaryNameNode
14748 Master
13790 DataNode
13631 NameNode
14495 NodeManager
yang@yancy:/usr/local$
```

（12）通过 WEB 页面查看：浏览器中输入地址：localhost:8080。



Spark Master at spark://yancy:7077

URL: spark://yancy:7077
Alive Workers: 1
Cores in use: 12 Total, 0 Used
Memory in use: 1024.0 MB Total, 0.0 B Used
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (1)

Worker Id	Address	State	Cores	Memory
worker-20190412170910-192.168.0.62-44315	192.168.0.62:44315	ALIVE	12 (0 Used)	1024.0 MB (0.0 B Used)

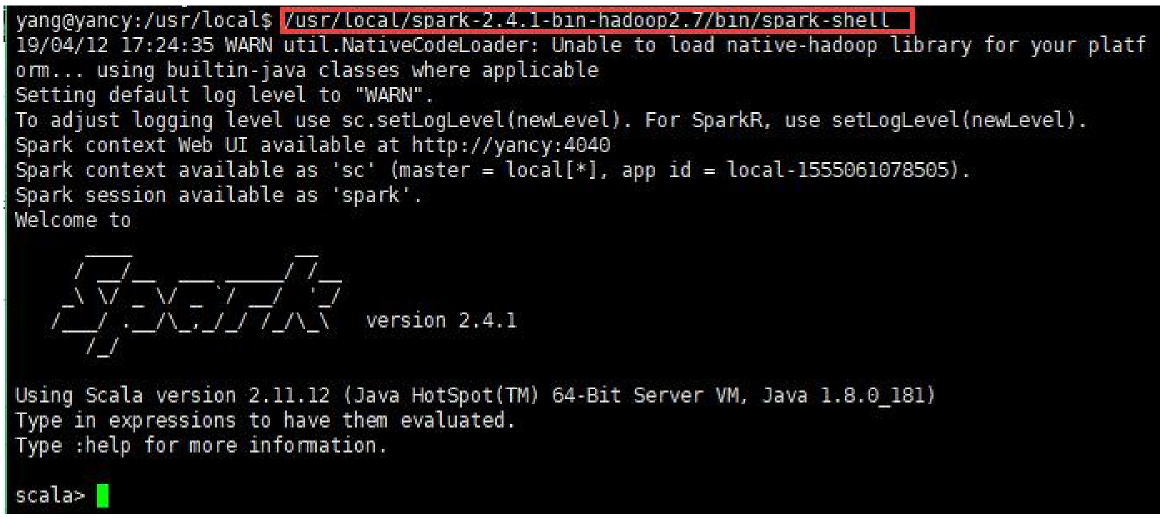
Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration

（13）输 入 ： /usr/local/spark-2.4.1-bin-hadoop2.7/bin/spark-shell ， 启 动 SparkContext。



```
yang@yancy:/usr/local$ /usr/local/spark-2.4.1-bin-hadoop2.7/bin/spark-shell
19/04/12 17:24:35 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platf
orm... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://yancy:4040
Spark context available as 'sc' (master = local[*], app id = local-1555061078505).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.4.1
      /_/

Using Scala version 2.11.12 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_181)
Type in expressions to have them evaluated.
Type :help for more information.

scala> 
```

（14）编写停止脚本 stop_script.sh 停止 Hadoop 以及 Spark：

#!/bin/bash

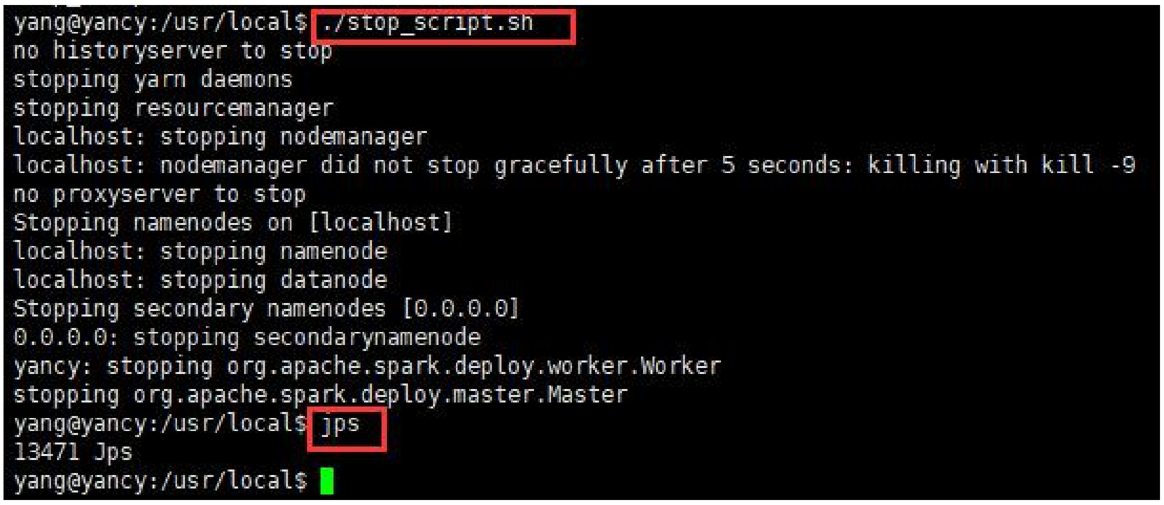
mr-jobhistory-daemon.sh stop historyserver # 停止历史服务器

stop-yarn.sh # 停止 Yarn

stop-dfs.sh # 停止 Hadoop

#/usr/local/hadoop/sbin/stop-all.sh # 停止 Hadoop 以及 yarn

/usr/local/spark-2.4.1-bin-hadoop2.7/sbin/stop-all.sh # 停止 Spark



```
yang@yancy:/usr/local$ ./stop_script.sh
no historyserver to stop
stopping yarn daemons
stopping resourcemanager
localhost: stopping nodemanager
localhost: nodemanager did not stop gracefully after 5 seconds: killing with kill -9
no proxyserver to stop
Stopping namenodes on [localhost]
localhost: stopping namenode
localhost: stopping datanode
Stopping secondary namenodes [0.0.0.0]
0.0.0.0: stopping secondarynamenode
yancy: stopping org.apache.spark.deploy.worker.Worker
stopping org.apache.spark.deploy.master.Master
yang@yancy:/usr/local$ jps
13471 Jps
yang@yancy:/usr/local$ 
```

4.3 ubuntu16.04 配置 Hadoop 全分布式集群

搭建完全分布式集群需准备三台主机，一个主节点 yancy 和两个从节点 Slave、Slave2，首先需要对主机名进行修改。

4.3.1 配置 hosts 文件

（1）输入：sudo vim /etc/hostname，修改主机名为 yancy；

（2）输入：sudo vim /etc/hosts，添加各主机 IP 地址如下：

```
192.168.0.62      yancy
192.168.0.104     Slave1
192.168.0.50      Slave2
```

（3）其它两台从节点也都要修改 hostname 和 hosts 文件。配置完 hosts 后三台主机就可以进行通信了，可以互相 ping 通，是可以 ping 通的。

4.3.2 SSH 免密码登录

（1）输入：dpkg --list|grep ssh，查看安装的 openssh-server；

（2）输入：ssh-keygen -t rsa，三次回车后，该目录下将会产生 id_rsa，id_rsa.pub 文件；

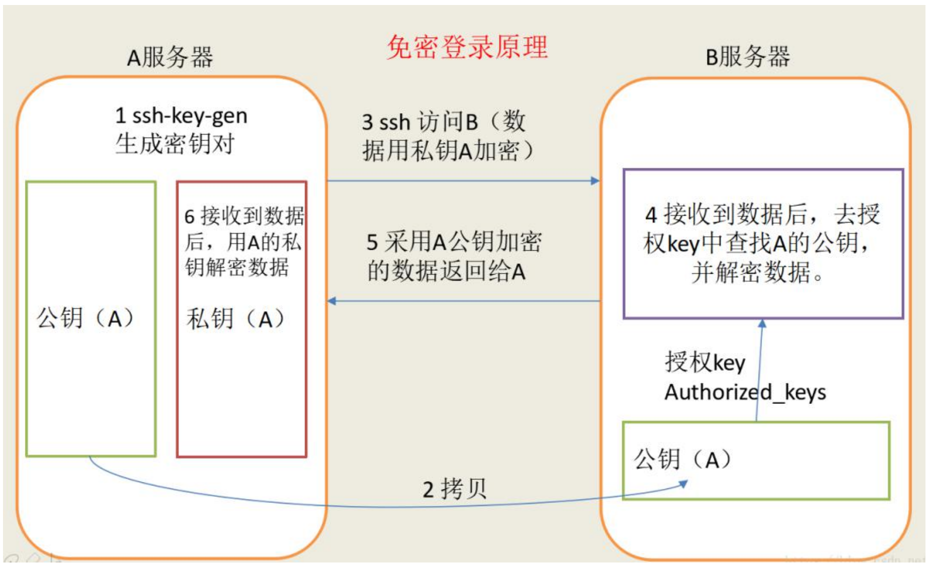
（3）输入：cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys 加入授权；

（4）输入：scp /home/yang/.ssh/id_rsa.pub qxxhemu@Slave1:~/.ssh/，将公钥复制到其他从机，或 ssh-copy-id -i ~/.ssh/id_rsa.pub Slave1；

（5）输入：scp /home/yang/.ssh/id_rsa.pub long@Slave2:~/.ssh/

（6）输入：cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys 加入授权；

（7）输入：ssh qxxhemu@Slave1，第一次需要密码，之后 exit 退出，再 ssh qxxhemu@Slave1 就不需要密码登录成功。


免密登录原理
A服务器
B服务器
1 ssh-key-gen
生成密钥对
3 ssh 访问B（数据用私钥A加密）
6 接收到数据后，用A的私钥解密数据
5 采用A公钥加密的数据返回给A
4 接收到数据后，去授权key中查找A的公钥，并解密数据。
公钥（A）
私钥（A）
授权key
Authorized_keys
公钥（A）
2 拷贝

4.3.3 修改 slaves 文件

（1）输入：sudo vim /usr/local/hadoop-2.7.6/etc/hadoop/slaves，添加内容：Slave1 和 Slave2；

4.3.4 配置 Hadoop 及 spark 文件

其余步骤类似，不再详述。

5 实验过程

5.1 WordCount 词频统计

5.1.1 MapReduce 词频统计

下面为一些 HDFS 常用命令：

hadoop fs -mkdir /tmp/input 在 HDFS 上目录/tmp/input；

hadoop fs -put input1.txt /tmp/input 把本地文件 input1.txt 传到 HDFS 的/tmp/input 目录下；

hadoop fs -get input1.txt /tmp/input/input1.txt 把 HDFS 文件拉到本地；

hadoop fs -ls /tmp/output 列出 HDFS 的目录/tmp/output；

hadoop fs -cat /tmp/ouput/output1.txt 查看 HDFS 上文件/tmp/ouput/output1.txt；

hadoop fs -rmr /tmp/intput 删除 HDFS 上的目录/tmp/intput；

hadoop dfsadmin -report 查看 HDFS 状态，比如每个 DataNode 的情况；

hadoop dfsadmin -safemode leave 离开安全模式；

hadoop dfsadmin -safemode enter 进入安全模式。

WordCount 词频统计如下步骤：

（1）输入：start-all.sh，启动 HDFS；

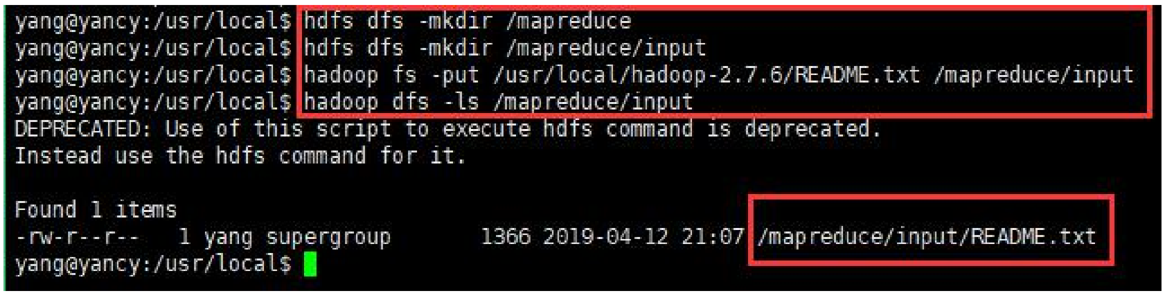
（2）输入：hadoop dfs -ls /，查看 HDFS 下面包含的文件目录，第一次运行 hdfs 什么文件都没有；

（3）输入：hdfs dfs -mkdir -p /mapreduce/input，在 HDFS 中创建一个文件目录/mapreduce/input；

（4）输入：hadoop fs -put /usr/local/hadoop-2.7.6/README.txt /mapreduce/input，将/usr/local/hadoop-2.7.6/README.txt 上传至/mapreduce/input 中；

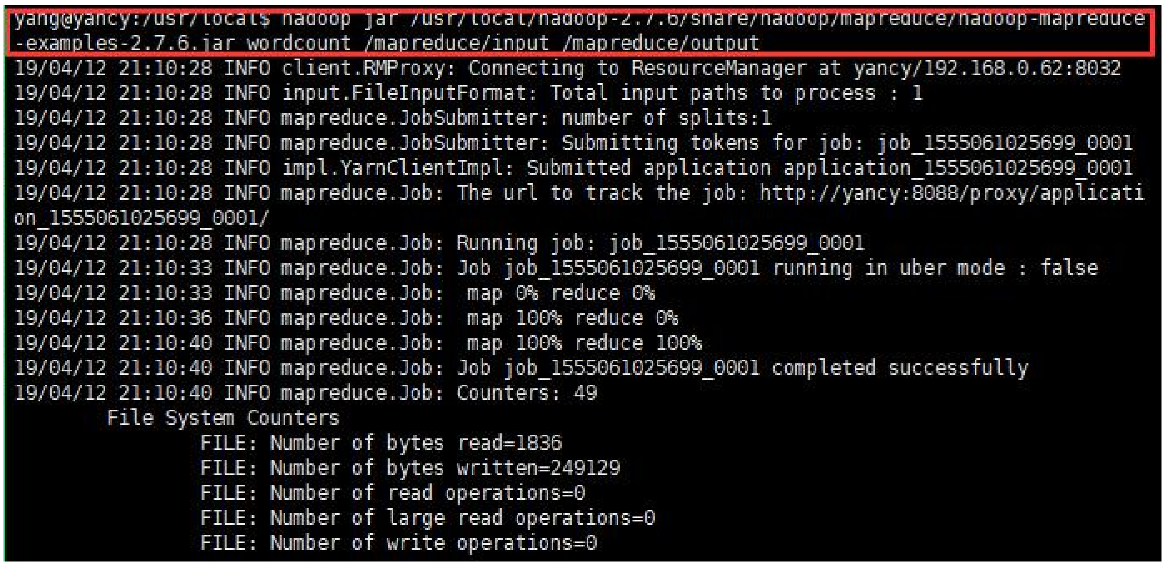
（5）输入：hadoop dfs -ls /mapreduce/input 查看/mapreduce/input 下多了一个

README.txt；

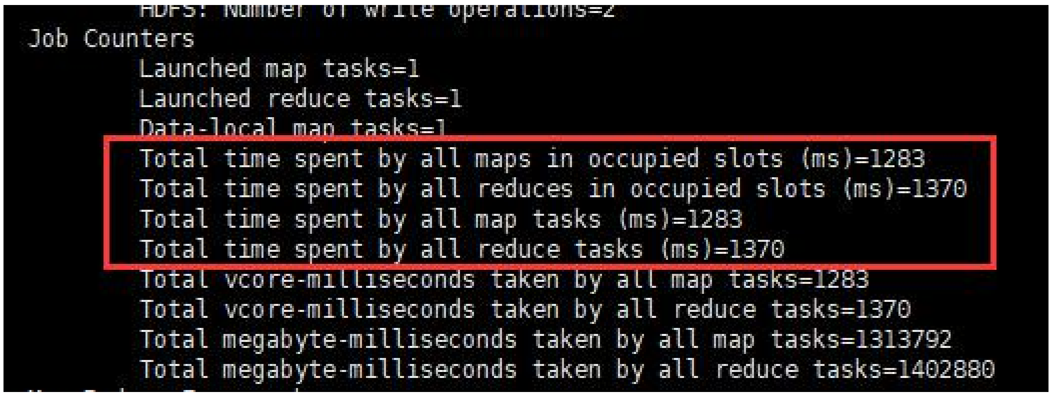


```
yang@yancy:/usr/local$ hdfs dfs -mkdir /mapreduce
yang@yancy:/usr/local$ hdfs dfs -mkdir /mapreduce/input
yang@yancy:/usr/local$ hadoop fs -put /usr/local/hadoop-2.7.6/README.txt /mapreduce/input
yang@yancy:/usr/local$ hadoop dfs -ls /mapreduce/input
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

Found 1 items
-rw-r--r--   1 yang supergroup       1366 2019-04-12 21:07 /mapreduce/input/README.txt
yang@yancy:/usr/local$ 
```

（6）执 行 如 下 命 令 运 行 wordcount 并 将 结 果 输 到 output ： hadoop jar /usr/local/hadoop-2.7.6/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.6.jar wordcount /mapreduce/input /mapreduce/output；

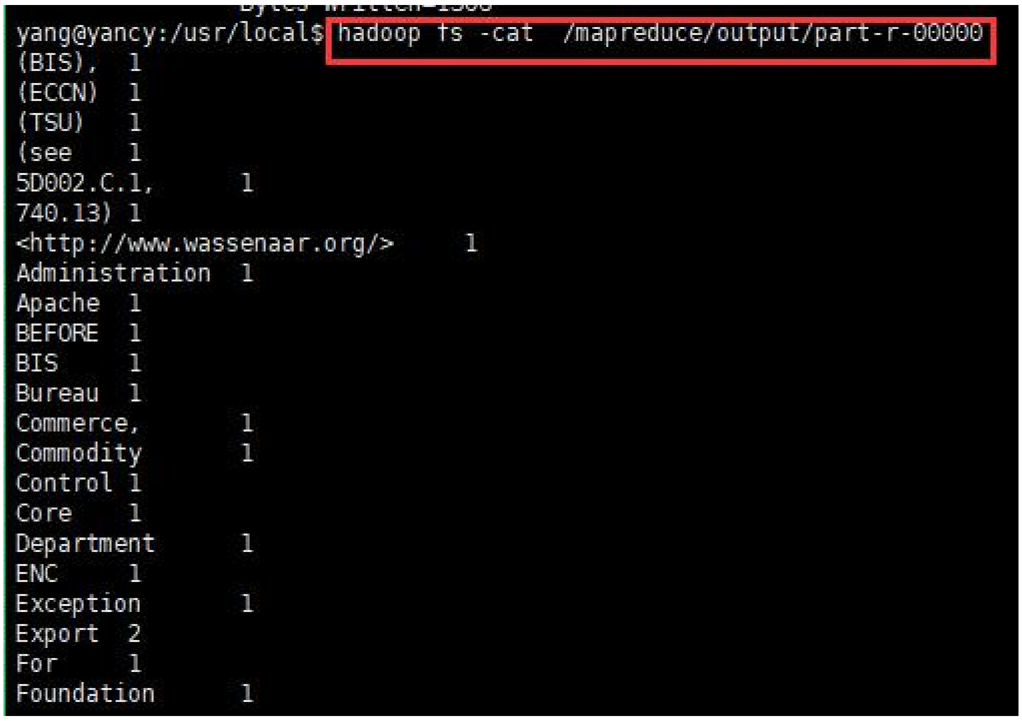


```
yang@yancy:/usr/local$ hadoop jar /usr/local/hadoop-2.7.6/share/hadoop/mapreduce/hadoop-mapreduce
-examples-2.7.6.jar wordcount /mapreduce/input /mapreduce/output
19/04/12 21:10:28 INFO client.RMProxy: Connecting to ResourceManager at yancy/192.168.0.62:8032
19/04/12 21:10:28 INFO input.FileInputFormat: Total input paths to process : 1
19/04/12 21:10:28 INFO mapreduce.JobSubmitter: number of splits:1
19/04/12 21:10:28 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1555061025699_0001
19/04/12 21:10:28 INFO impl.YarnClientImpl: Submitted application application_1555061025699_0001
19/04/12 21:10:28 INFO mapreduce.Job: The url to track the job: http://yancy:8088/proxy/applicati
on_1555061025699_0001/
19/04/12 21:10:28 INFO mapreduce.Job: Running job: job_1555061025699_0001
19/04/12 21:10:33 INFO mapreduce.Job: Job job_1555061025699_0001 running in uber mode : false
19/04/12 21:10:33 INFO mapreduce.Job:  map 0% reduce 0%
19/04/12 21:10:36 INFO mapreduce.Job:  map 100% reduce 0%
19/04/12 21:10:40 INFO mapreduce.Job:  map 100% reduce 100%
19/04/12 21:10:40 INFO mapreduce.Job: Job job_1555061025699_0001 completed successfully
19/04/12 21:10:40 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=1836
                FILE: Number of bytes written=249129
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
```

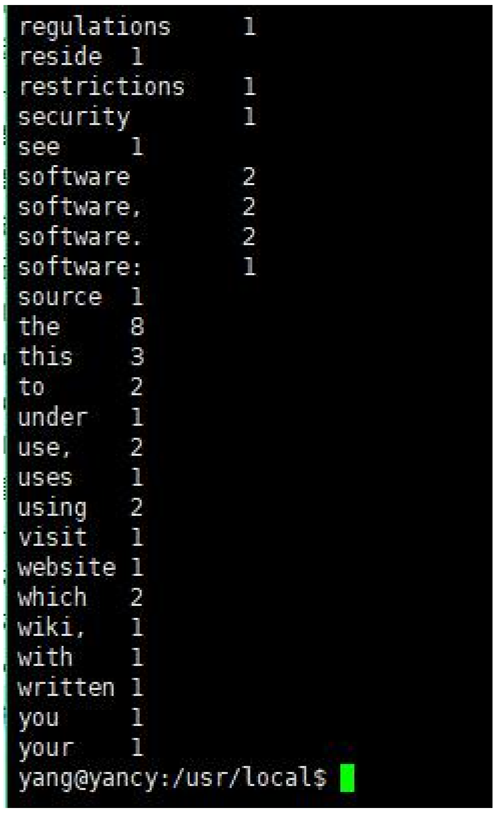


```
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=1283
                Total time spent by all reduces in occupied slots (ms)=1370
                Total time spent by all map tasks (ms)=1283
                Total time spent by all reduce tasks (ms)=1370
                Total vcore-milliseconds taken by all map tasks=1283
                Total vcore-milliseconds taken by all reduce tasks=1370
                Total megabyte-milliseconds taken by all map tasks=1313792
                Total megabyte-milliseconds taken by all reduce tasks=1402880
```

（7）执行成功后 output 目录下会生成两个文件：一个是_SUCCESS 成功标志的文件，里面没有内容，另一个是 part-r-00000 ，通过以下命令查看执行的结果：hadoop fs -cat　/mapreduce/output/part-r-00000。



```
yang@yancy:/usr/local$ hadoop fs -cat  /mapreduce/output/part-r-00000
(BIS),  1
(ECCN)  1
(TSU)   1
(see    1
5D002.C.1,      1
740.13) 1
<http://www.wassenaar.org/>     1
Administration  1
Apache  1
BEFORE  1
BIS     1
Bureau  1
Commerce,       1
Commodity       1
Control 1
Core    1
Department      1
ENC     1
Exception       1
Export  2
For     1
Foundation      1
```



```
regulations     1
reside  1
restrictions    1
security        1
see     1
software        2
software,       2
software.       2
software:       1
source  1
the     8
this    3
to      2
under   1
use,    2
uses    1
using   2
visit   1
website 1
which   2
wiki,   1
with    1
written 1
you     1
your    1
yang@yancy:/usr/local$
```

5.1.2 Spark 词频统计

（1）输入：/usr/local/spark-2.4.1-bin-hadoop2.7/bin/spark-shell，启动 spark shell；

（2）或者：/usr/local/spark-2.4.1-bin-hadoop2.7/bin/spark-shell \

\> --master spark://yancy:7077 \

> --executor-memory 500m

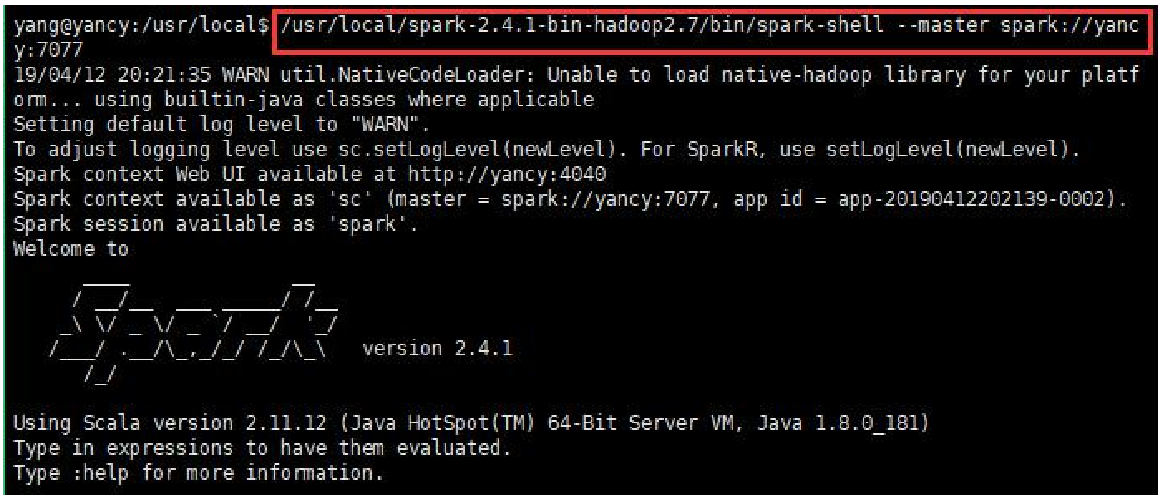
> --total-executor-cores 1

参数说明：

--master spark://yancy:7077 ：指定 Master 的地址；

--executor-memory 500m ：指定每个 worker 可用内存为 500m；

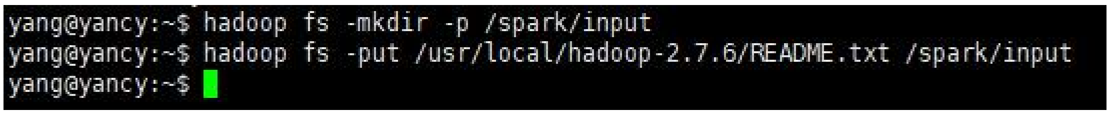
--total-executor-cores 1 ：指定整个集群使用的 CPU 核数为 1 个；



```
yang@yancy:/usr/local$ /usr/local/spark-2.4.1-bin-hadoop2.7/bin/spark-shell --master spark://yanc
y:7077
19/04/12 20:21:35 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platf
orm... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://yancy:4040
Spark context available as 'sc' (master = spark://yancy:7077, app id = app-20190412202139-0002).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.4.1
      /_/

Using Scala version 2.11.12 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_181)
Type in expressions to have them evaluated.
Type :help for more information.
```

（3）输入：hadoop fs -mkdir -p /spark/input，创建/spark/input 文件夹；

（4）输入：hadoop fs -put /usr/local/hadoop-2.7.6/README.txt /spark/input，将/usr/local/hadoop-2.7.6/README.txt 上传至/spark/input 中；

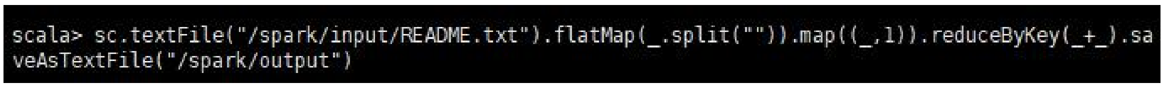


```
yang@yancy:~$ hadoop fs -mkdir -p /spark/input
yang@yancy:~$ hadoop fs -put /usr/local/hadoop-2.7.6/README.txt /spark/input
yang@yancy:~$ 
```

（5）在 spark shell 中用 scala 编写 spark 程序，按空格分割数据，输入：

sc.textFile("/spark/input/README.txt").flatMap(_.split("

")).map((_,1)).reduceByKey(_+_).saveAsTextFile("/spark/output")；



```
scala> sc.textFile("/spark/input/README.txt").flatMap(_.split("")).map((_,1)).reduceByKey(_+_).sa
veAsTextFile("/spark/output")
```

说明：

sc 是 SparkContext 对象，该对象是提交 spark 程序的入口；

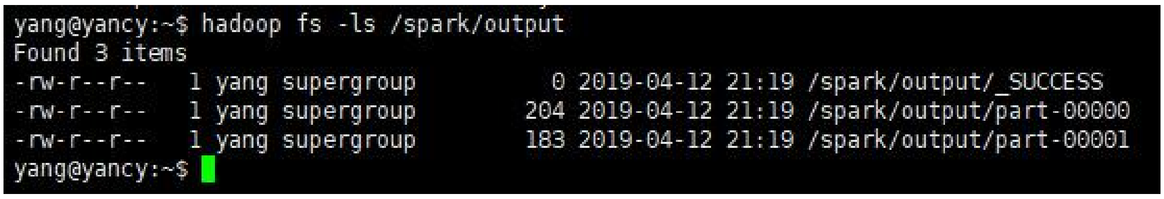
textFile("/spark/input/README.txt") 是从 hdfs 中读取数据；

flatMap(_.split(" ")) 先 map 再压平；

map((_,1)) 将单词和 1 构成元组；

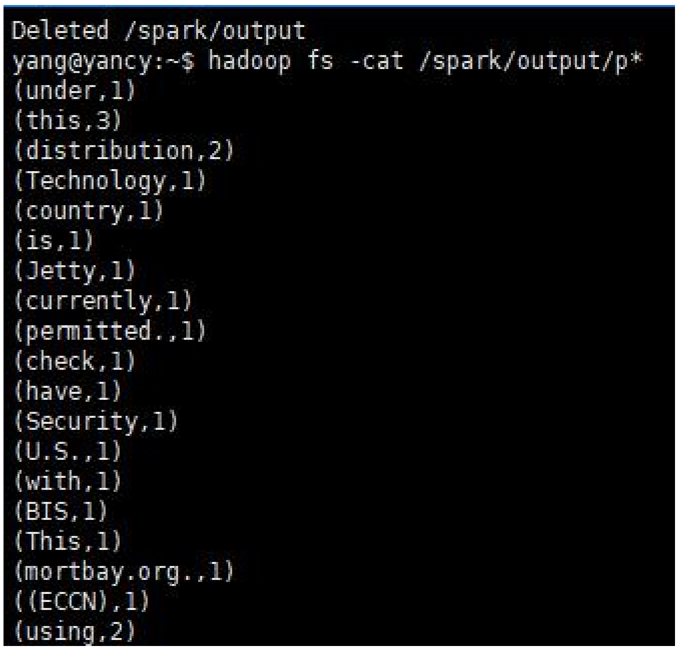
reduceByKey(_+_) 按照 key 进行 reduce，并将 value 累加；

saveAsTextFile("/spark/output") 将结果写入到 hdfs 中。



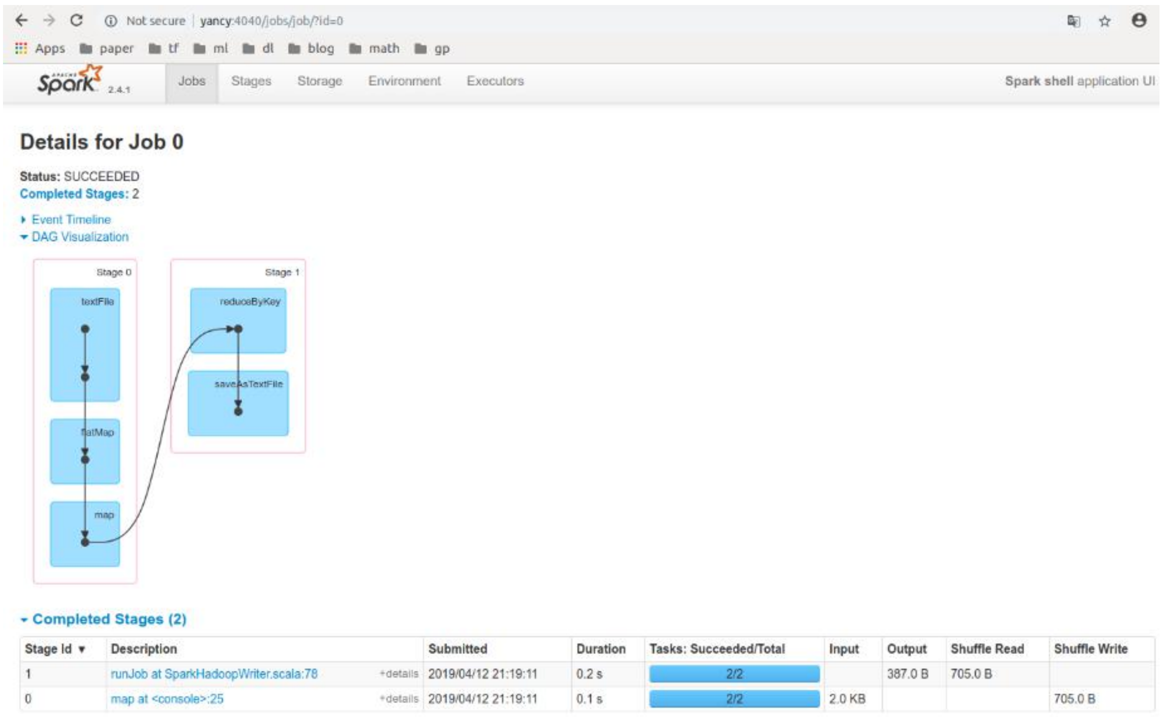
```
yang@yancy:~$ hadoop fs -ls /spark/output
Found 3 items
-rw-r--r--   1 yang supergroup          0 2019-04-12 21:19 /spark/output/_SUCCESS
-rw-r--r--   1 yang supergroup        204 2019-04-12 21:19 /spark/output/part-00000
-rw-r--r--   1 yang supergroup        183 2019-04-12 21:19 /spark/output/part-00001
yang@yancy:~$
```

（6）输入：hadoop fs -cat /spark/output/p*，查看 hdfs 的执行结果：



```
Deleted /spark/output
yang@yancy:~$ hadoop fs -cat /spark/output/p*
(under,1)
(this,3)
(distribution,2)
(Technology,1)
(country,1)
(is,1)
(Jetty,1)
(currently,1)
(permitted.,1)
(check,1)
(have,1)
(Security,1)
(U.S.,1)
(with,1)
(BIS,1)
(This,1)
(mortbay.org.,1)
((ECCN),1)
(using,2)
```

（7）在浏览器输入：yancy:4040，查看 spark UI 。


Not secure | yancy:4040/jobs/job/?id=0
Apps paper tf ml dl blog math gp
Spark 2.4.1
Jobs Stages Storage Environment Executors
Spark shell application UI
Stage 0
textFile
flatMap
map
Stage 1
reduceByKey
saveAsTextFile

Details for Job 0

Status: SUCCEEDED
Completed Stages: 2

Event Timeline
DAG Visualization

Completed Stages (2)

Stage Id ▾	Description		Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
1	runJob at SparkHadoopWriter.scala:78	+details	2019/04/12 21:19:11	0.2 s	2/2		387.0 B	705.0 B	
0	map at <console>:25	+details	2019/04/12 21:19:11	0.1 s	2/2	2.0 KB			705.0 B

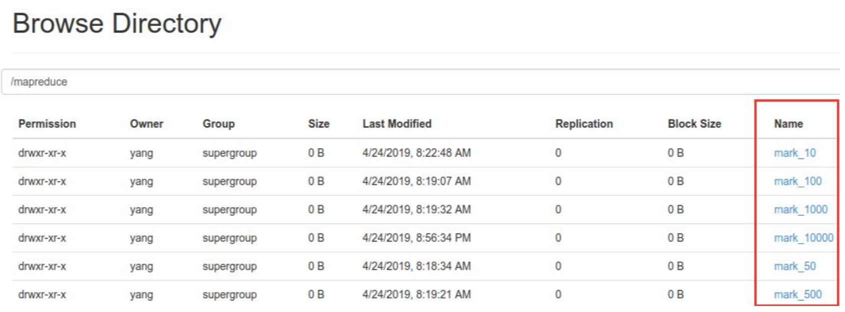
6　实验结果对比分析

6.1　MapReduce 执行效率

本次实验选取的实验样本为五个大小分别为 10M、50M、100M、500M、100M、10000M 的英文文本文件，使用 MapReduce、Spark 框架分别运行不同大小文本进行对比分析相关指标，进而统计各自的运行时间、运行时 CPU 占用率、内存占用率等执行效率和容错性、通用性的对比。

6.1.1 运行时间

（1）查看 10M、50M、100M、500M、100M、10000M 大小文本：

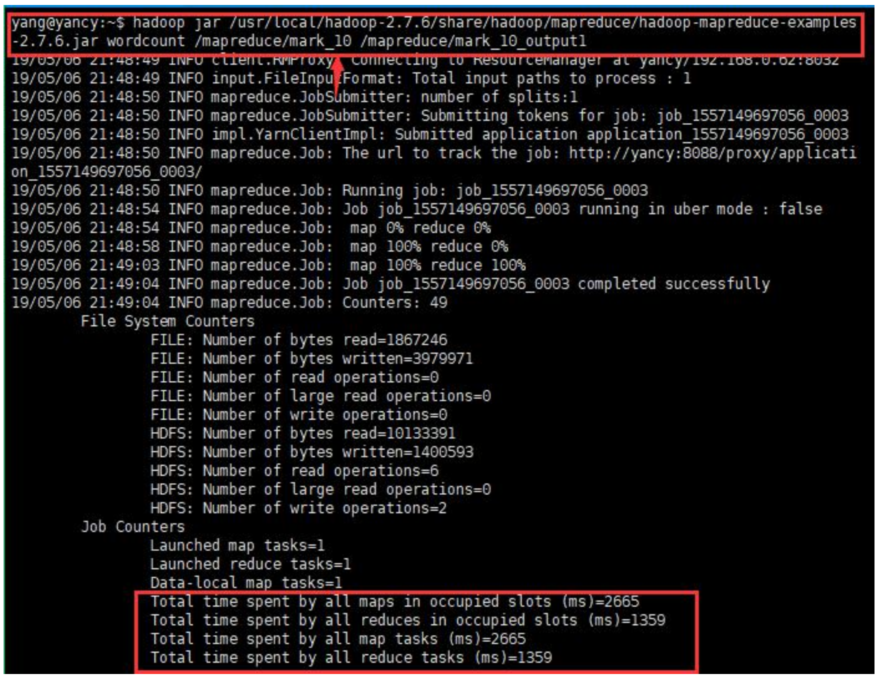


Browse Directory

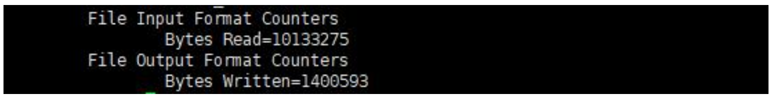
/mapreduce

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	yang	supergroup	0 B	4/24/2019, 8:22:48 AM	0	0 B	mark_10
drwxr-xr-x	yang	supergroup	0 B	4/24/2019, 8:19:07 AM	0	0 B	mark_100
drwxr-xr-x	yang	supergroup	0 B	4/24/2019, 8:19:32 AM	0	0 B	mark_1000
drwxr-xr-x	yang	supergroup	0 B	4/24/2019, 8:56:34 PM	0	0 B	mark_10000
drwxr-xr-x	yang	supergroup	0 B	4/24/2019, 8:18:34 AM	0	0 B	mark_50
drwxr-xr-x	yang	supergroup	0 B	4/24/2019, 8:19:21 AM	0	0 B	mark_500

（2）10M 大小文本执行 Map、Reduce 时间及文件大小输入输出结果图：

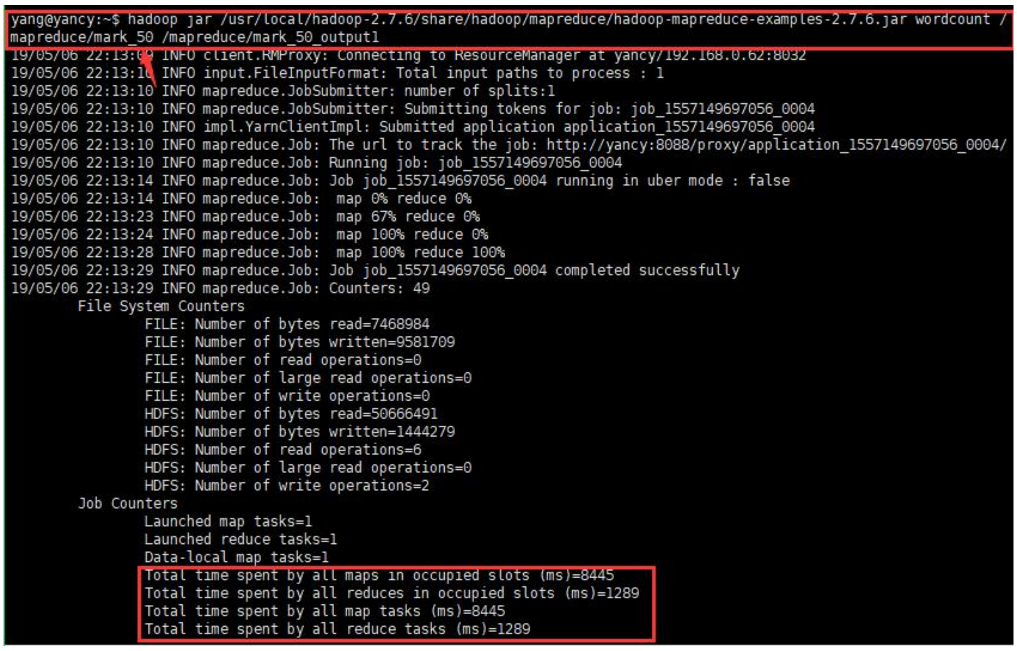


```
yang@yancy:~$ hadoop jar /usr/local/hadoop-2.7.6/share/hadoop/mapreduce/hadoop-mapreduce-examples
-2.7.6.jar wordcount /mapreduce/mark_10 /mapreduce/mark_10_output1
19/05/06 21:48:49 INFO client.RMProxy: Connecting to ResourceManager at yancy/192.168.0.62:8032
19/05/06 21:48:49 INFO input.FileInputFormat: Total input paths to process : 1
19/05/06 21:48:50 INFO mapreduce.JobSubmitter: number of splits:1
19/05/06 21:48:50 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1557149697056_0003
19/05/06 21:48:50 INFO impl.YarnClientImpl: Submitted application application_1557149697056_0003
19/05/06 21:48:50 INFO mapreduce.Job: The url to track the job: http://yancy:8088/proxy/applicati
on_1557149697056_0003/
19/05/06 21:48:50 INFO mapreduce.Job: Running job: job_1557149697056_0003
19/05/06 21:48:54 INFO mapreduce.Job: Job job_1557149697056_0003 running in uber mode : false
19/05/06 21:48:54 INFO mapreduce.Job:  map 0% reduce 0%
19/05/06 21:48:58 INFO mapreduce.Job:  map 100% reduce 0%
19/05/06 21:49:03 INFO mapreduce.Job:  map 100% reduce 100%
19/05/06 21:49:04 INFO mapreduce.Job: Job job_1557149697056_0003 completed successfully
19/05/06 21:49:04 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=1867246
                FILE: Number of bytes written=3979971
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=10133391
                HDFS: Number of bytes written=1400593
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=2665
                Total time spent by all reduces in occupied slots (ms)=1359
                Total time spent by all map tasks (ms)=2665
                Total time spent by all reduce tasks (ms)=1359
```

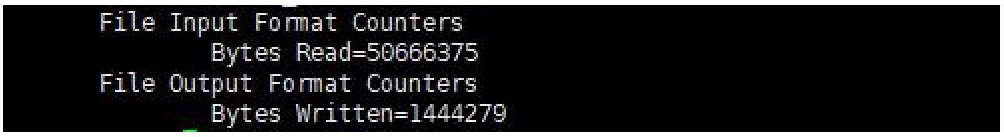


```
        File Input Format Counters
                Bytes Read=10133275
        File Output Format Counters
                Bytes Written=1400593
```

（3）50M 大小文本执行 Map、Reduce 时间及文件大小输入输出结果图：

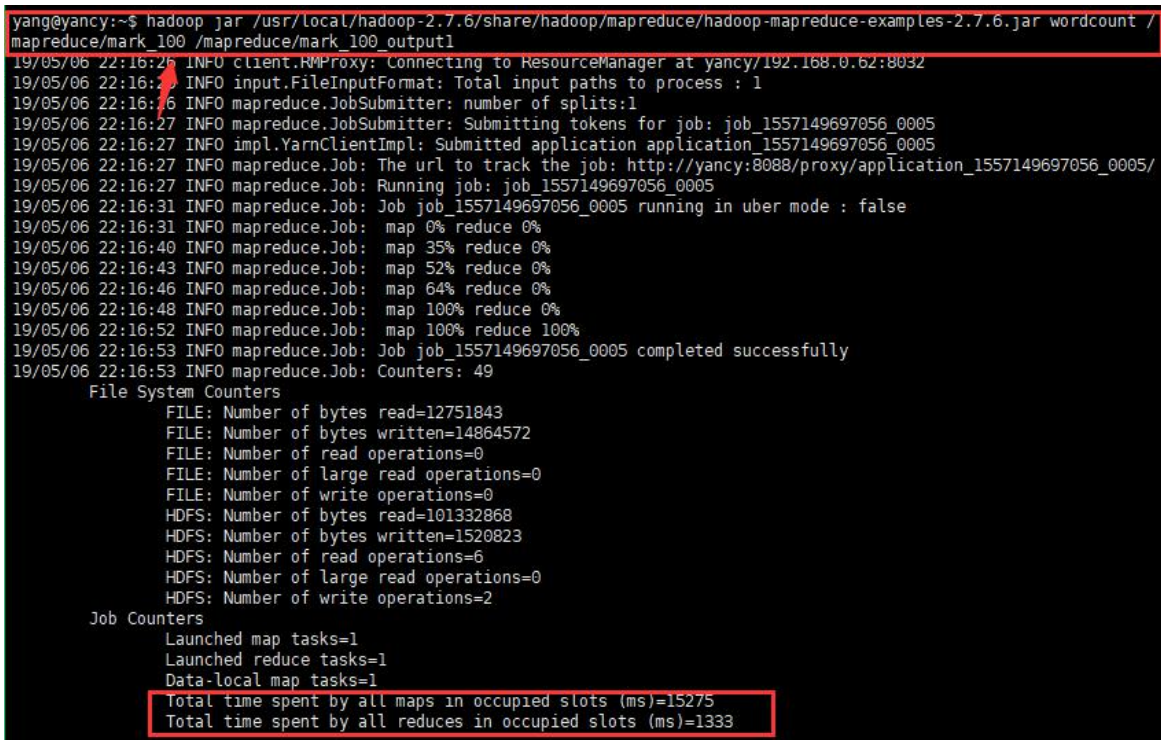


```
yang@yancy:~$ hadoop jar /usr/local/hadoop-2.7.6/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.6.jar wordcount /
mapreduce/mark_50 /mapreduce/mark_50_output1
19/05/06 22:13:09 INFO client.RMProxy: Connecting to ResourceManager at yancy/192.168.0.62:8032
19/05/06 22:13:10 INFO input.FileInputFormat: Total input paths to process : 1
19/05/06 22:13:10 INFO mapreduce.JobSubmitter: number of splits:1
19/05/06 22:13:10 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1557149697056_0004
19/05/06 22:13:10 INFO impl.YarnClientImpl: Submitted application application_1557149697056_0004
19/05/06 22:13:10 INFO mapreduce.Job: The url to track the job: http://yancy:8088/proxy/application_1557149697056_0004/
19/05/06 22:13:10 INFO mapreduce.Job: Running job: job_1557149697056_0004
19/05/06 22:13:14 INFO mapreduce.Job: Job job_1557149697056_0004 running in uber mode : false
19/05/06 22:13:14 INFO mapreduce.Job:  map 0% reduce 0%
19/05/06 22:13:23 INFO mapreduce.Job:  map 67% reduce 0%
19/05/06 22:13:24 INFO mapreduce.Job:  map 100% reduce 0%
19/05/06 22:13:28 INFO mapreduce.Job:  map 100% reduce 100%
19/05/06 22:13:29 INFO mapreduce.Job: Job job_1557149697056_0004 completed successfully
19/05/06 22:13:29 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=7468984
                FILE: Number of bytes written=9581709
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=50666491
                HDFS: Number of bytes written=1444279
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=8445
                Total time spent by all reduces in occupied slots (ms)=1289
                Total time spent by all map tasks (ms)=8445
                Total time spent by all reduce tasks (ms)=1289
```

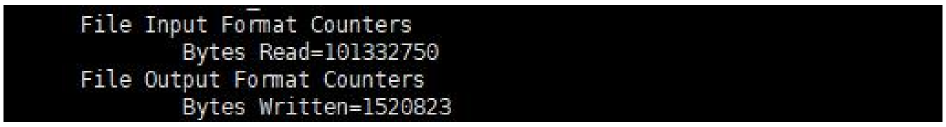


```
        File Input Format Counters
                Bytes Read=50666375
        File Output Format Counters
                Bytes Written=1444279
```

（4）100M 大小文本执行 Map、Reduce 时间及文件大小输入输出结果图：

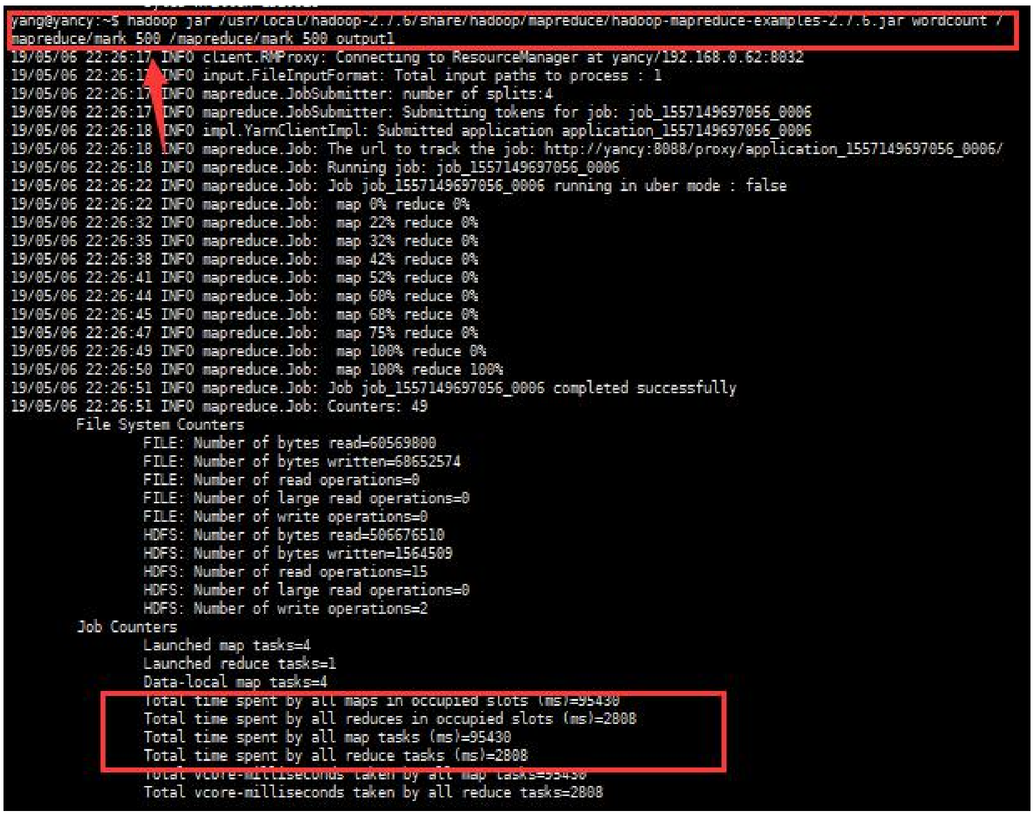


```
yang@yancy:~$ hadoop jar /usr/local/hadoop-2.7.6/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.6.jar wordcount /
mapreduce/mark_100 /mapreduce/mark_100_output1
19/05/06 22:16:26 INFO client.RMProxy: Connecting to ResourceManager at yancy/192.168.0.62:8032
19/05/06 22:16:26 INFO input.FileInputFormat: Total input paths to process : 1
19/05/06 22:16:26 INFO mapreduce.JobSubmitter: number of splits:1
19/05/06 22:16:27 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1557149697056_0005
19/05/06 22:16:27 INFO impl.YarnClientImpl: Submitted application application_1557149697056_0005
19/05/06 22:16:27 INFO mapreduce.Job: The url to track the job: http://yancy:8088/proxy/application_1557149697056_0005/
19/05/06 22:16:27 INFO mapreduce.Job: Running job: job_1557149697056_0005
19/05/06 22:16:31 INFO mapreduce.Job: Job job_1557149697056_0005 running in uber mode : false
19/05/06 22:16:31 INFO mapreduce.Job:  map 0% reduce 0%
19/05/06 22:16:40 INFO mapreduce.Job:  map 35% reduce 0%
19/05/06 22:16:43 INFO mapreduce.Job:  map 52% reduce 0%
19/05/06 22:16:46 INFO mapreduce.Job:  map 64% reduce 0%
19/05/06 22:16:48 INFO mapreduce.Job:  map 100% reduce 0%
19/05/06 22:16:52 INFO mapreduce.Job:  map 100% reduce 100%
19/05/06 22:16:53 INFO mapreduce.Job: Job job_1557149697056_0005 completed successfully
19/05/06 22:16:53 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=12751843
                FILE: Number of bytes written=14864572
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=101332868
                HDFS: Number of bytes written=1520823
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=15275
                Total time spent by all reduces in occupied slots (ms)=1333
```

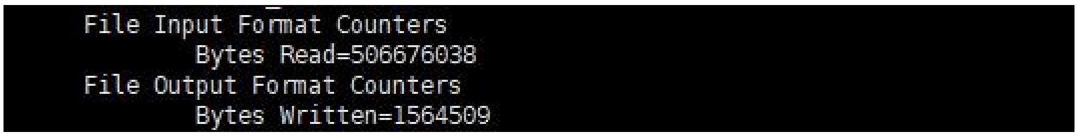


```
        File Input Format Counters
                Bytes Read=101332750
        File Output Format Counters
                Bytes Written=1520823
```

（5）500M 大小文本执行 Map、Reduce 时间及文件大小输入输出结果图：

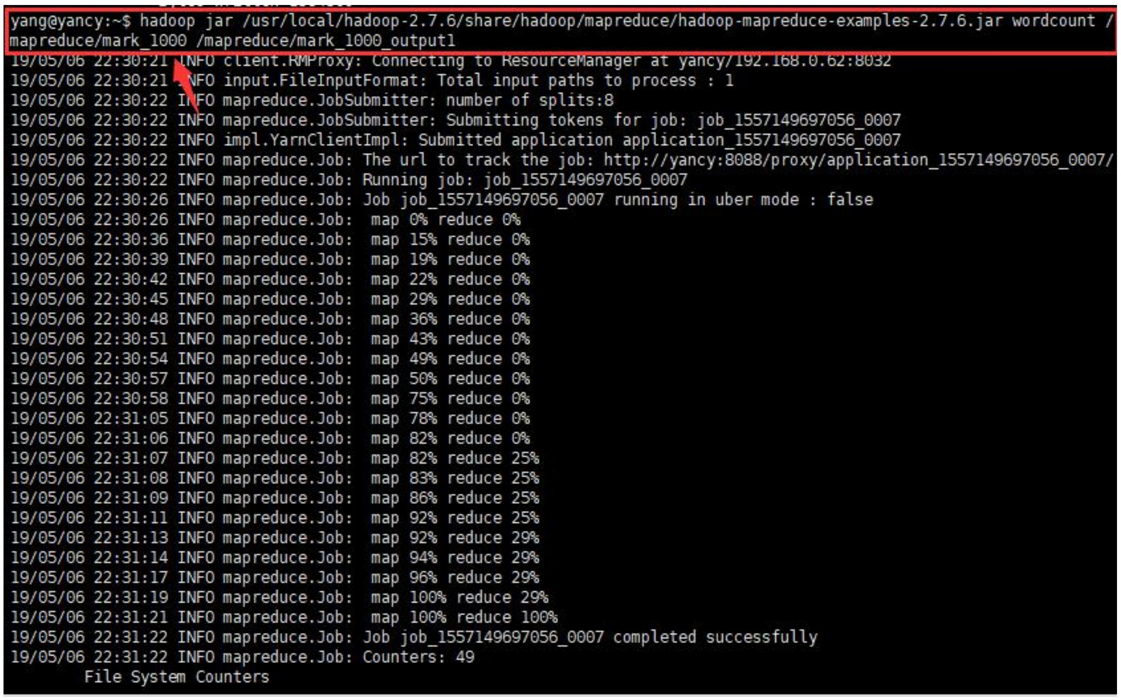


```
yang@yancy:~$ hadoop jar /usr/local/hadoop-2.7.6/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.6.jar wordcount /
mapreduce/mark_500 /mapreduce/mark_500_output1
19/05/06 22:26:17 INFO client.RMProxy: Connecting to ResourceManager at yancy/192.168.0.62:8032
19/05/06 22:26:17 INFO input.FileInputFormat: Total input paths to process : 1
19/05/06 22:26:17 INFO mapreduce.JobSubmitter: number of splits:4
19/05/06 22:26:17 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1557149697056_0006
19/05/06 22:26:18 INFO impl.YarnClientImpl: Submitted application application_1557149697056_0006
19/05/06 22:26:18 INFO mapreduce.Job: The url to track the job: http://yancy:8088/proxy/application_1557149697056_0006/
19/05/06 22:26:18 INFO mapreduce.Job: Running job: job_1557149697056_0006
19/05/06 22:26:22 INFO mapreduce.Job: Job job_1557149697056_0006 running in uber mode : false
19/05/06 22:26:22 INFO mapreduce.Job:  map 0% reduce 0%
19/05/06 22:26:32 INFO mapreduce.Job:  map 22% reduce 0%
19/05/06 22:26:35 INFO mapreduce.Job:  map 32% reduce 0%
19/05/06 22:26:38 INFO mapreduce.Job:  map 42% reduce 0%
19/05/06 22:26:41 INFO mapreduce.Job:  map 52% reduce 0%
19/05/06 22:26:44 INFO mapreduce.Job:  map 60% reduce 0%
19/05/06 22:26:45 INFO mapreduce.Job:  map 68% reduce 0%
19/05/06 22:26:47 INFO mapreduce.Job:  map 75% reduce 0%
19/05/06 22:26:49 INFO mapreduce.Job:  map 100% reduce 0%
19/05/06 22:26:50 INFO mapreduce.Job:  map 100% reduce 100%
19/05/06 22:26:51 INFO mapreduce.Job: Job job_1557149697056_0006 completed successfully
19/05/06 22:26:51 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=60569800
                FILE: Number of bytes written=68652574
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=506676510
                HDFS: Number of bytes written=1564509
                HDFS: Number of read operations=15
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=4
                Launched reduce tasks=1
                Data-local map tasks=4
                Total time spent by all maps in occupied slots (ms)=95430
                Total time spent by all reduces in occupied slots (ms)=2808
                Total time spent by all map tasks (ms)=95430
                Total time spent by all reduce tasks (ms)=2808
                Total vcore-milliseconds taken by all map tasks=95430
                Total vcore-milliseconds taken by all reduce tasks=2808
```

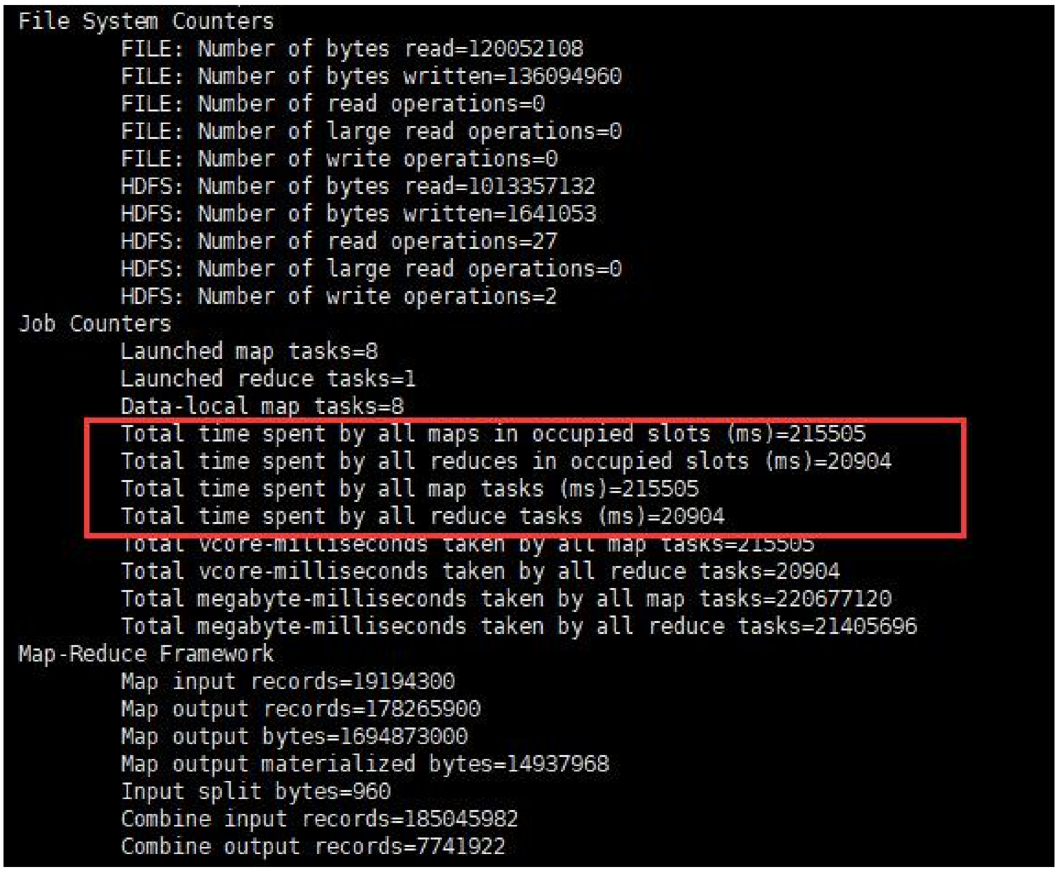


```
        File Input Format Counters
                Bytes Read=506676038
        File Output Format Counters
                Bytes Written=1564509
```

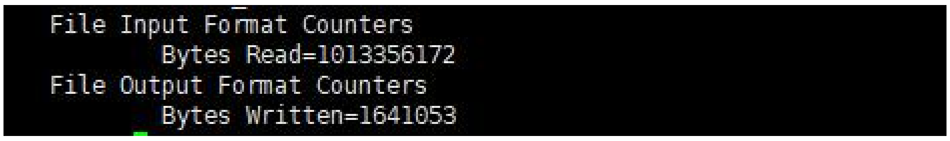
（6）1000M 大小文本执行 Map、Reduce 时间及文件大小输入输出结果图：



```
yang@yancy:~$ hadoop jar /usr/local/hadoop-2.7.6/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.6.jar wordcount /
mapreduce/mark_1000 /mapreduce/mark_1000_output1
19/05/06 22:30:21 INFO client.RMProxy: Connecting to ResourceManager at yancy/192.168.0.62:8032
19/05/06 22:30:21 INFO input.FileInputFormat: Total input paths to process : 1
19/05/06 22:30:22 INFO mapreduce.JobSubmitter: number of splits:8
19/05/06 22:30:22 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1557149697056_0007
19/05/06 22:30:22 INFO impl.YarnClientImpl: Submitted application application_1557149697056_0007
19/05/06 22:30:22 INFO mapreduce.Job: The url to track the job: http://yancy:8088/proxy/application_1557149697056_0007/
19/05/06 22:30:22 INFO mapreduce.Job: Running job: job_1557149697056_0007
19/05/06 22:30:26 INFO mapreduce.Job: Job job_1557149697056_0007 running in uber mode : false
19/05/06 22:30:26 INFO mapreduce.Job:  map 0% reduce 0%
19/05/06 22:30:36 INFO mapreduce.Job:  map 15% reduce 0%
19/05/06 22:30:39 INFO mapreduce.Job:  map 19% reduce 0%
19/05/06 22:30:42 INFO mapreduce.Job:  map 22% reduce 0%
19/05/06 22:30:45 INFO mapreduce.Job:  map 29% reduce 0%
19/05/06 22:30:48 INFO mapreduce.Job:  map 36% reduce 0%
19/05/06 22:30:51 INFO mapreduce.Job:  map 43% reduce 0%
19/05/06 22:30:54 INFO mapreduce.Job:  map 49% reduce 0%
19/05/06 22:30:57 INFO mapreduce.Job:  map 50% reduce 0%
19/05/06 22:30:58 INFO mapreduce.Job:  map 75% reduce 0%
19/05/06 22:31:05 INFO mapreduce.Job:  map 78% reduce 0%
19/05/06 22:31:06 INFO mapreduce.Job:  map 82% reduce 0%
19/05/06 22:31:07 INFO mapreduce.Job:  map 82% reduce 25%
19/05/06 22:31:08 INFO mapreduce.Job:  map 83% reduce 25%
19/05/06 22:31:09 INFO mapreduce.Job:  map 86% reduce 25%
19/05/06 22:31:11 INFO mapreduce.Job:  map 92% reduce 25%
19/05/06 22:31:13 INFO mapreduce.Job:  map 92% reduce 29%
19/05/06 22:31:14 INFO mapreduce.Job:  map 94% reduce 29%
19/05/06 22:31:17 INFO mapreduce.Job:  map 96% reduce 29%
19/05/06 22:31:19 INFO mapreduce.Job:  map 100% reduce 29%
19/05/06 22:31:21 INFO mapreduce.Job:  map 100% reduce 100%
19/05/06 22:31:22 INFO mapreduce.Job: Job job_1557149697056_0007 completed successfully
19/05/06 22:31:22 INFO mapreduce.Job: Counters: 49
        File System Counters
```

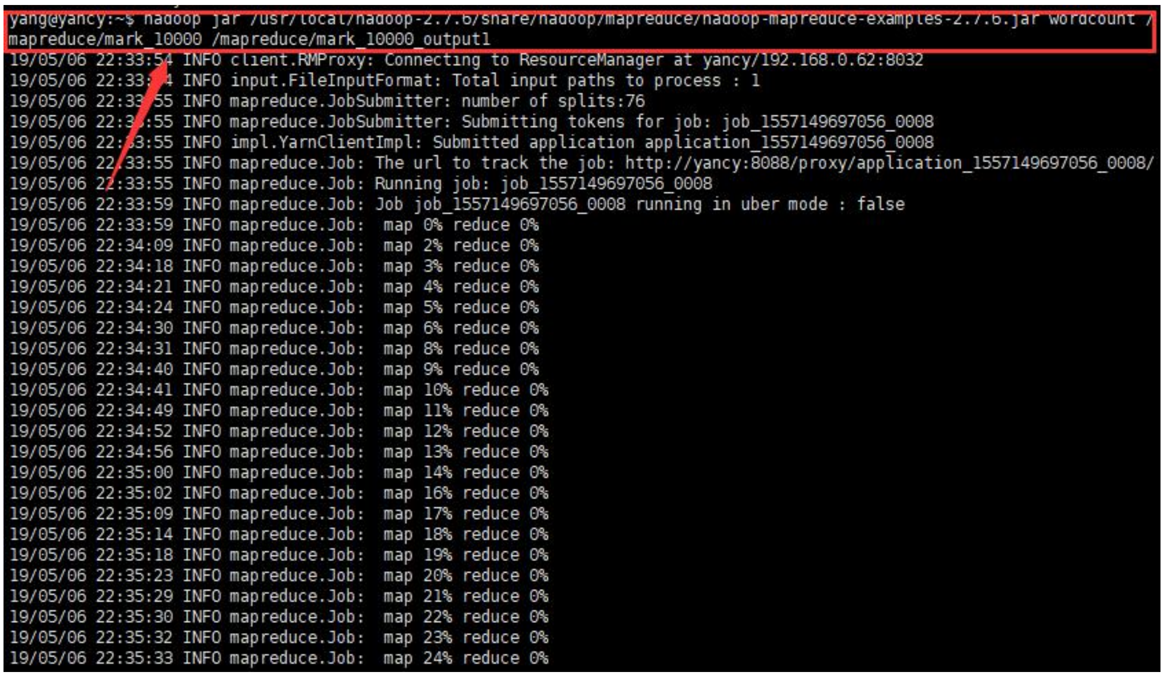


```
File System Counters
        FILE: Number of bytes read=120052108
        FILE: Number of bytes written=136094960
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=1013357132
        HDFS: Number of bytes written=1641053
        HDFS: Number of read operations=27
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=2
Job Counters
        Launched map tasks=8
        Launched reduce tasks=1
        Data-local map tasks=8
        Total time spent by all maps in occupied slots (ms)=215505
        Total time spent by all reduces in occupied slots (ms)=20904
        Total time spent by all map tasks (ms)=215505
        Total time spent by all reduce tasks (ms)=20904
        Total vcore-milliseconds taken by all map tasks=215505
        Total vcore-milliseconds taken by all reduce tasks=20904
        Total megabyte-milliseconds taken by all map tasks=220677120
        Total megabyte-milliseconds taken by all reduce tasks=21405696
Map-Reduce Framework
        Map input records=19194300
        Map output records=178265900
        Map output bytes=1694873000
        Map output materialized bytes=14937968
        Input split bytes=960
        Combine input records=185045982
        Combine output records=7741922
```

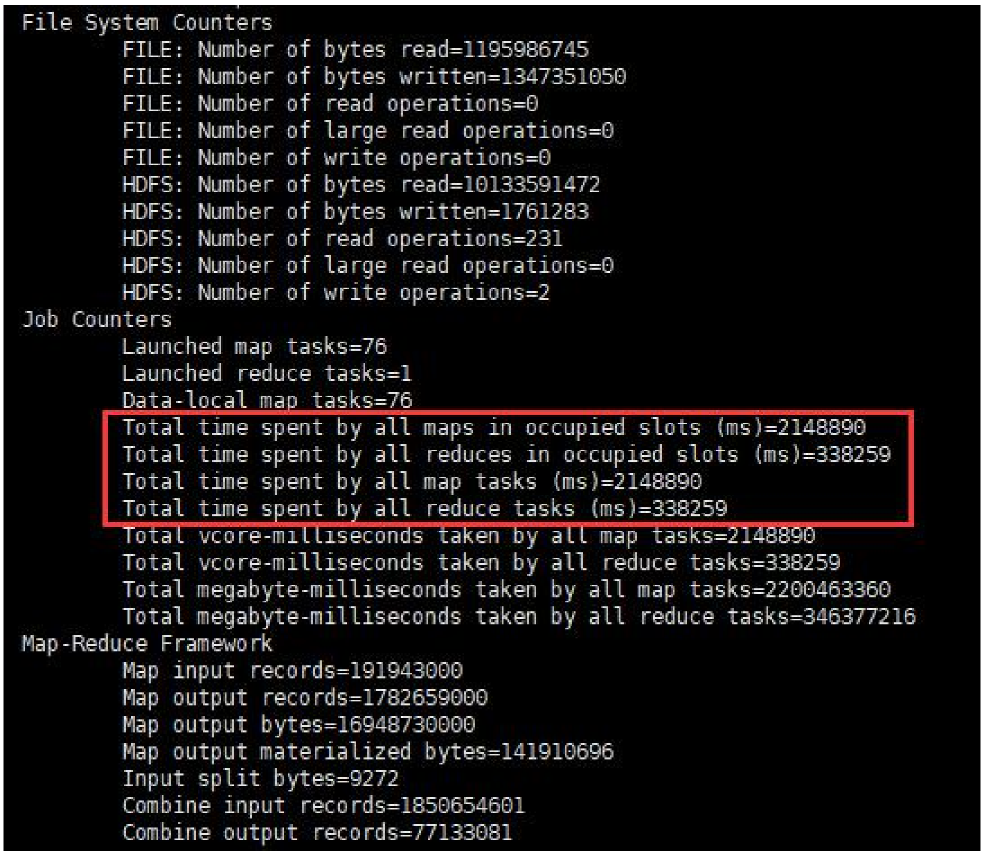


```
  File Input Format Counters
          Bytes Read=1013356172
  File Output Format Counters
          Bytes Written=1641053
```

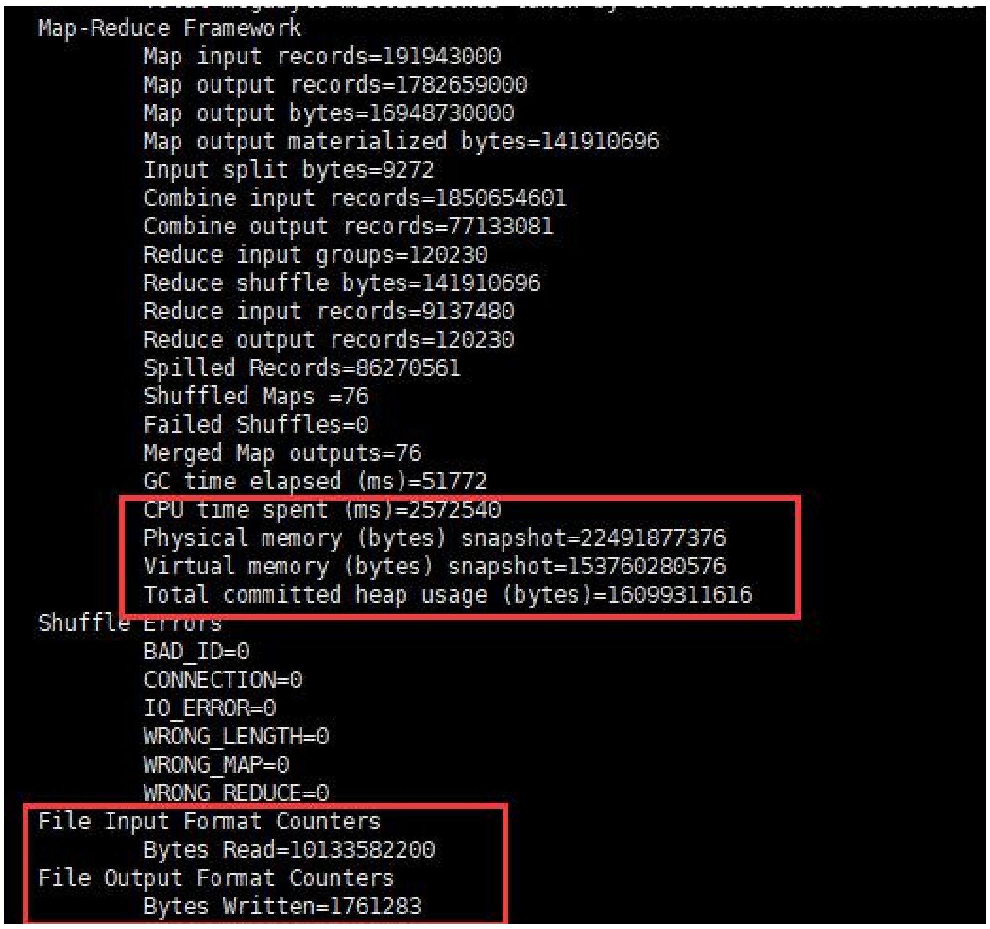
（7）10000M 大小文本执行 Map、Reduce 时间及文件大小输入输出结果图：



```
yang@yancy:~$ hadoop jar /usr/local/hadoop-2.7.6/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.6.jar wordcount /
mapreduce/mark_10000 /mapreduce/mark_10000_output1
19/05/06 22:33:54 INFO client.RMProxy: Connecting to ResourceManager at yancy/192.168.0.62:8032
19/05/06 22:33:54 INFO input.FileInputFormat: Total input paths to process : 1
19/05/06 22:33:55 INFO mapreduce.JobSubmitter: number of splits:76
19/05/06 22:33:55 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1557149697056_0008
19/05/06 22:33:55 INFO impl.YarnClientImpl: Submitted application application_1557149697056_0008
19/05/06 22:33:55 INFO mapreduce.Job: The url to track the job: http://yancy:8088/proxy/application_1557149697056_0008/
19/05/06 22:33:55 INFO mapreduce.Job: Running job: job_1557149697056_0008
19/05/06 22:33:59 INFO mapreduce.Job: Job job_1557149697056_0008 running in uber mode : false
19/05/06 22:33:59 INFO mapreduce.Job:  map 0% reduce 0%
19/05/06 22:34:09 INFO mapreduce.Job:  map 2% reduce 0%
19/05/06 22:34:18 INFO mapreduce.Job:  map 3% reduce 0%
19/05/06 22:34:21 INFO mapreduce.Job:  map 4% reduce 0%
19/05/06 22:34:24 INFO mapreduce.Job:  map 5% reduce 0%
19/05/06 22:34:30 INFO mapreduce.Job:  map 6% reduce 0%
19/05/06 22:34:31 INFO mapreduce.Job:  map 8% reduce 0%
19/05/06 22:34:40 INFO mapreduce.Job:  map 9% reduce 0%
19/05/06 22:34:41 INFO mapreduce.Job:  map 10% reduce 0%
19/05/06 22:34:49 INFO mapreduce.Job:  map 11% reduce 0%
19/05/06 22:34:52 INFO mapreduce.Job:  map 12% reduce 0%
19/05/06 22:34:56 INFO mapreduce.Job:  map 13% reduce 0%
19/05/06 22:35:00 INFO mapreduce.Job:  map 14% reduce 0%
19/05/06 22:35:02 INFO mapreduce.Job:  map 16% reduce 0%
19/05/06 22:35:09 INFO mapreduce.Job:  map 17% reduce 0%
19/05/06 22:35:14 INFO mapreduce.Job:  map 18% reduce 0%
19/05/06 22:35:18 INFO mapreduce.Job:  map 19% reduce 0%
19/05/06 22:35:23 INFO mapreduce.Job:  map 20% reduce 0%
19/05/06 22:35:29 INFO mapreduce.Job:  map 21% reduce 0%
19/05/06 22:35:30 INFO mapreduce.Job:  map 22% reduce 0%
19/05/06 22:35:32 INFO mapreduce.Job:  map 23% reduce 0%
19/05/06 22:35:33 INFO mapreduce.Job:  map 24% reduce 0%
```

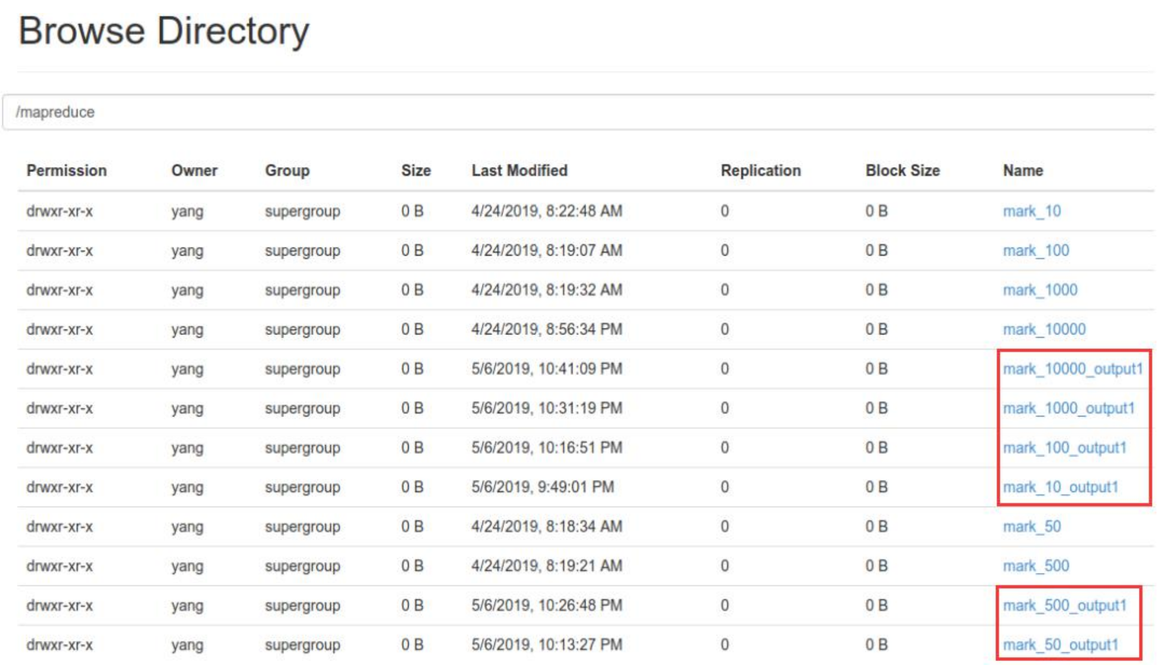


```
File System Counters
        FILE: Number of bytes read=1195986745
        FILE: Number of bytes written=1347351050
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=10133591472
        HDFS: Number of bytes written=1761283
        HDFS: Number of read operations=231
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=2
Job Counters
        Launched map tasks=76
        Launched reduce tasks=1
        Data-local map tasks=76
        Total time spent by all maps in occupied slots (ms)=2148890
        Total time spent by all reduces in occupied slots (ms)=338259
        Total time spent by all map tasks (ms)=2148890
        Total time spent by all reduce tasks (ms)=338259
        Total vcore-milliseconds taken by all map tasks=2148890
        Total vcore-milliseconds taken by all reduce tasks=338259
        Total megabyte-milliseconds taken by all map tasks=2200463360
        Total megabyte-milliseconds taken by all reduce tasks=346377216
Map-Reduce Framework
        Map input records=191943000
        Map output records=1782659000
        Map output bytes=16948730000
        Map output materialized bytes=141910696
        Input split bytes=9272
        Combine input records=1850654601
        Combine output records=77133081
```



```
Map-Reduce Framework
        Map input records=191943000
        Map output records=1782659000
        Map output bytes=16948730000
        Map output materialized bytes=141910696
        Input split bytes=9272
        Combine input records=1850654601
        Combine output records=77133081
        Reduce input groups=120230
        Reduce shuffle bytes=141910696
        Reduce input records=9137480
        Reduce output records=120230
        Spilled Records=86270561
        Shuffled Maps =76
        Failed Shuffles=0
        Merged Map outputs=76
        GC time elapsed (ms)=51772
        CPU time spent (ms)=2572540
        Physical memory (bytes) snapshot=22491877376
        Virtual memory (bytes) snapshot=153760280576
        Total committed heap usage (bytes)=16099311616
Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
File Input Format Counters
        Bytes Read=10133582200
File Output Format Counters
        Bytes Written=1761283
```

（8）10M、50M、100M、500M、100M、10000M 文本执行生成输出图：

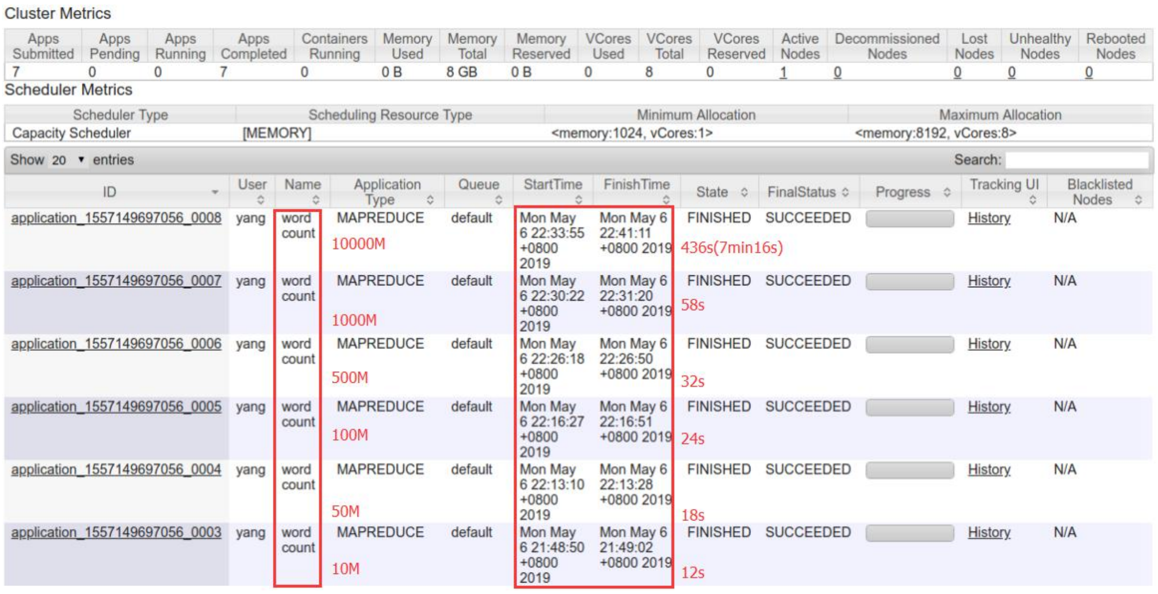


Browse Directory

/mapreduce

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	yang	supergroup	0 B	4/24/2019, 8:22:48 AM	0	0 B	mark_10
drwxr-xr-x	yang	supergroup	0 B	4/24/2019, 8:19:07 AM	0	0 B	mark_100
drwxr-xr-x	yang	supergroup	0 B	4/24/2019, 8:19:32 AM	0	0 B	mark_1000
drwxr-xr-x	yang	supergroup	0 B	4/24/2019, 8:56:34 PM	0	0 B	mark_10000
drwxr-xr-x	yang	supergroup	0 B	5/6/2019, 10:41:09 PM	0	0 B	mark_10000_output1
drwxr-xr-x	yang	supergroup	0 B	5/6/2019, 10:31:19 PM	0	0 B	mark_1000_output1
drwxr-xr-x	yang	supergroup	0 B	5/6/2019, 10:16:51 PM	0	0 B	mark_100_output1
drwxr-xr-x	yang	supergroup	0 B	5/6/2019, 9:49:01 PM	0	0 B	mark_10_output1
drwxr-xr-x	yang	supergroup	0 B	4/24/2019, 8:18:34 AM	0	0 B	mark_50
drwxr-xr-x	yang	supergroup	0 B	4/24/2019, 8:19:21 AM	0	0 B	mark_500
drwxr-xr-x	yang	supergroup	0 B	5/6/2019, 10:26:48 PM	0	0 B	mark_500_output1
drwxr-xr-x	yang	supergroup	0 B	5/6/2019, 10:13:27 PM	0	0 B	mark_50_output1

（9）10M、50M、100M、500M、100M、10000M 文本执行 MapReduce 各自时耗：



Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
7	0	0	7	0	0 B	8 GB	0 B	0	8	0	1	0	0	0	0

Scheduler Metrics

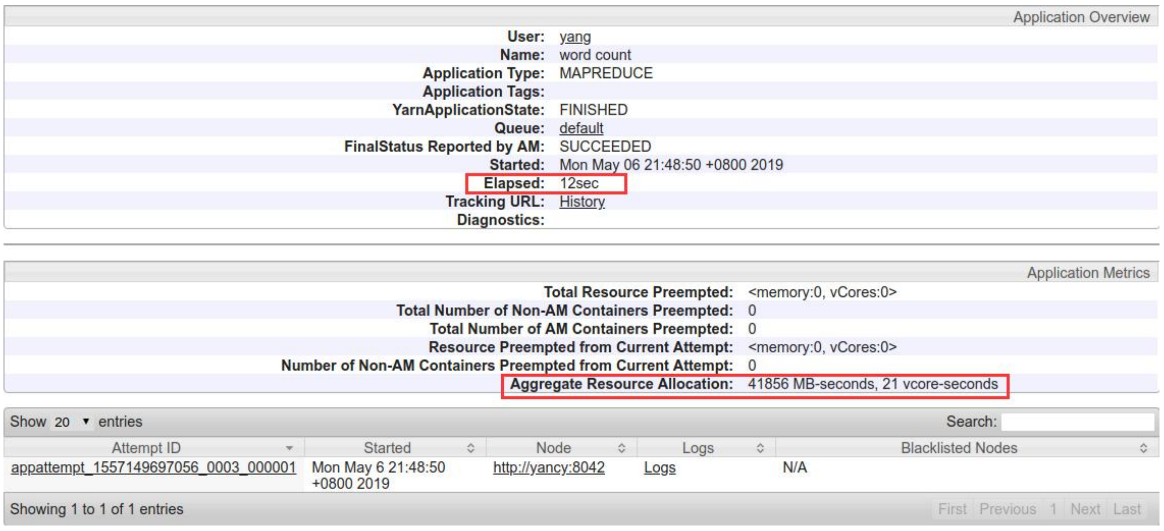
Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:8>

Show 20 ▾ entries　　Search:

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI	Blacklisted Nodes
application_1557149697056_0008	yang	word count	MAPREDUCE 10000M	default	Mon May 6 22:33:55 +0800 2019	Mon May 6 22:41:11 +0800 2019	FINISHED 436s(7min16s)	SUCCEEDED		History	N/A
application_1557149697056_0007	yang	word count	MAPREDUCE 1000M	default	Mon May 6 22:30:22 +0800 2019	Mon May 6 22:31:20 +0800 2019	FINISHED 58s	SUCCEEDED		History	N/A
application_1557149697056_0006	yang	word count	MAPREDUCE 500M	default	Mon May 6 22:26:18 +0800 2019	Mon May 6 22:26:50 +0800 2019	FINISHED 32s	SUCCEEDED		History	N/A
application_1557149697056_0005	yang	word count	MAPREDUCE 100M	default	Mon May 6 22:16:27 +0800 2019	Mon May 6 22:16:51 +0800 2019	FINISHED 24s	SUCCEEDED		History	N/A
application_1557149697056_0004	yang	word count	MAPREDUCE 50M	default	Mon May 6 22:13:10 +0800 2019	Mon May 6 22:13:28 +0800 2019	FINISHED 18s	SUCCEEDED		History	N/A
application_1557149697056_0003	yang	word count	MAPREDUCE 10M	default	Mon May 6 21:48:50 +0800 2019	Mon May 6 21:49:02 +0800 2019	FINISHED 12s	SUCCEEDED		History	N/A

6.1.2 资源分配量

（1）10M 文本执行 MapReduce 耗时及资源分配：



Application Overview

User:	yang
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Mon May 06 21:48:50 +0800 2019
Elapsed:	12sec
Tracking URL:	History
Diagnostics:	

Application Metrics

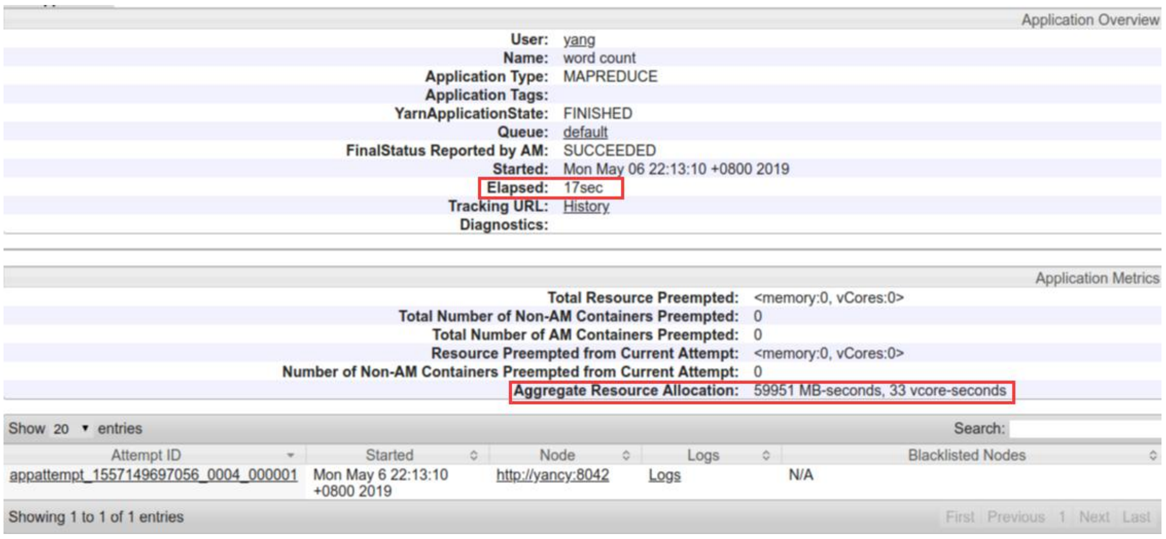
Total Resource Preempted:	<memory:0, vCores:0>
Total Number of Non-AM Containers Preempted:	0
Total Number of AM Containers Preempted:	0
Resource Preempted from Current Attempt:	<memory:0, vCores:0>
Number of Non-AM Containers Preempted from Current Attempt:	0
Aggregate Resource Allocation:	41856 MB-seconds, 21 vcore-seconds

Show 20 entries Search:

Attempt ID	Started	Node	Logs	Blacklisted Nodes
appattempt_1557149697056_0003_000001	Mon May 6 21:48:50 +0800 2019	http://yancy:8042	Logs	N/A

Showing 1 to 1 of 1 entries First Previous 1 Next Last

（2）50M 文本执行 MapReduce 耗时及资源分配：



Application Overview

User:	yang
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Mon May 06 22:13:10 +0800 2019
Elapsed:	17sec
Tracking URL:	History
Diagnostics:	

Application Metrics

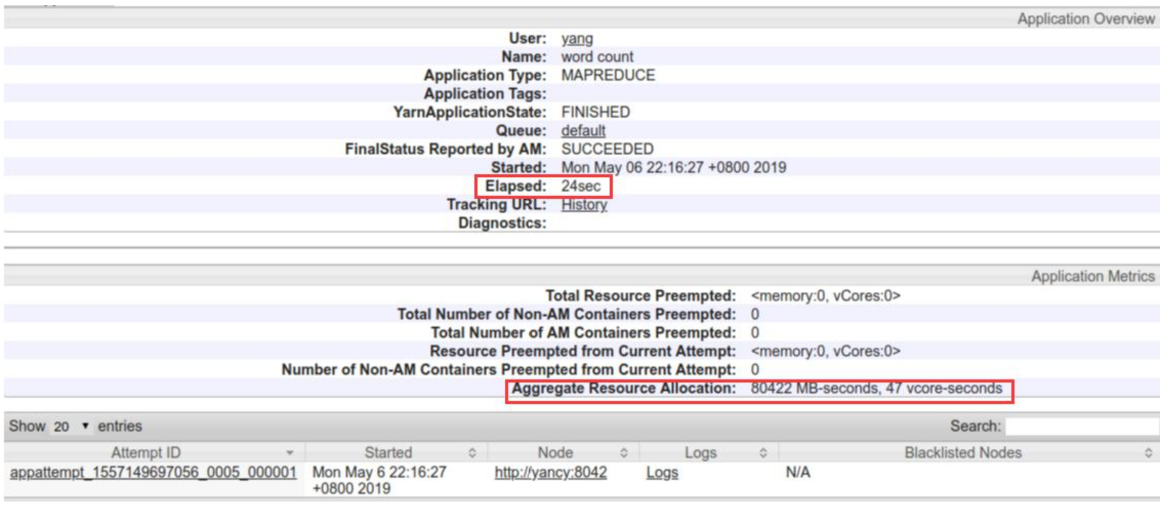
Total Resource Preempted:	<memory:0, vCores:0>
Total Number of Non-AM Containers Preempted:	0
Total Number of AM Containers Preempted:	0
Resource Preempted from Current Attempt:	<memory:0, vCores:0>
Number of Non-AM Containers Preempted from Current Attempt:	0
Aggregate Resource Allocation:	59951 MB-seconds, 33 vcore-seconds

Show 20 entries Search:

Attempt ID	Started	Node	Logs	Blacklisted Nodes
appattempt_1557149697056_0004_000001	Mon May 6 22:13:10 +0800 2019	http://yancy:8042	Logs	N/A

Showing 1 to 1 of 1 entries First Previous 1 Next Last

（3）100M 文本执行 MapReduce 耗时及资源分配：



Application Overview

User:	yang
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Mon May 06 22:16:27 +0800 2019
Elapsed:	24sec
Tracking URL:	History
Diagnostics:	

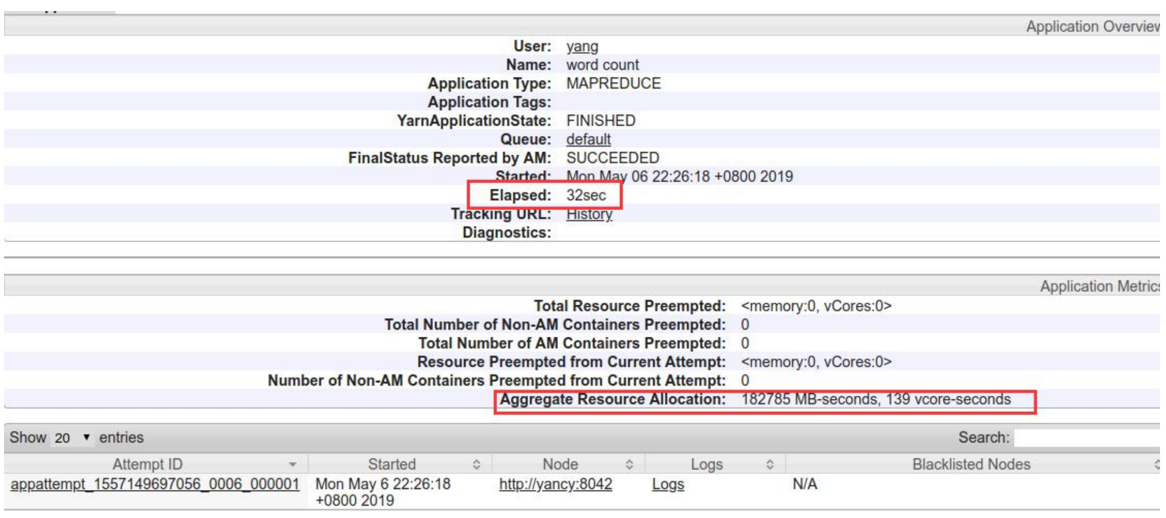
Application Metrics

Total Resource Preempted:	<memory:0, vCores:0>
Total Number of Non-AM Containers Preempted:	0
Total Number of AM Containers Preempted:	0
Resource Preempted from Current Attempt:	<memory:0, vCores:0>
Number of Non-AM Containers Preempted from Current Attempt:	0
Aggregate Resource Allocation:	80422 MB-seconds, 47 vcore-seconds

Show 20 entries Search:

Attempt ID	Started	Node	Logs	Blacklisted Nodes
appattempt_1557149697056_0005_000001	Mon May 6 22:16:27 +0800 2019	http://yancy:8042	Logs	N/A

（4）500M 文本执行 MapReduce 耗时及资源分配：



Application Overview

User:	yang
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Mon May 06 22:26:18 +0800 2019
Elapsed:	32sec
Tracking URL:	History
Diagnostics:	

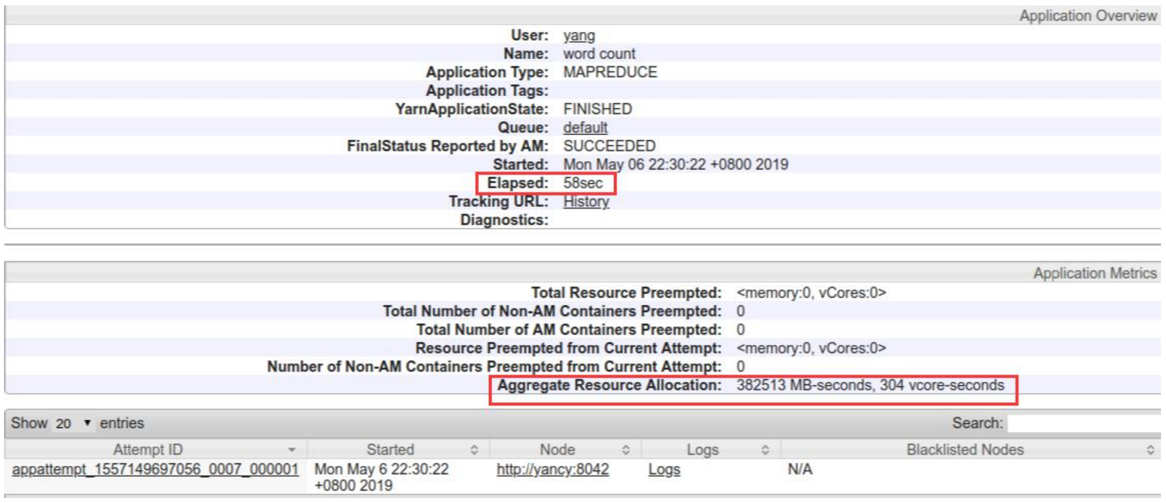
Application Metrics

Total Resource Preempted:	<memory:0, vCores:0>
Total Number of Non-AM Containers Preempted:	0
Total Number of AM Containers Preempted:	0
Resource Preempted from Current Attempt:	<memory:0, vCores:0>
Number of Non-AM Containers Preempted from Current Attempt:	0
Aggregate Resource Allocation:	182785 MB-seconds, 139 vcore-seconds

Show 20 entries Search:

Attempt ID	Started	Node	Logs	Blacklisted Nodes
appattempt_1557149697056_0006_000001	Mon May 6 22:26:18 +0800 2019	http://yancy:8042	Logs	N/A

（5）1000M 文本执行 MapReduce 耗时及资源分配：



Application Overview

User:	yang
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Mon May 06 22:30:22 +0800 2019
Elapsed:	58sec
Tracking URL:	History
Diagnostics:	

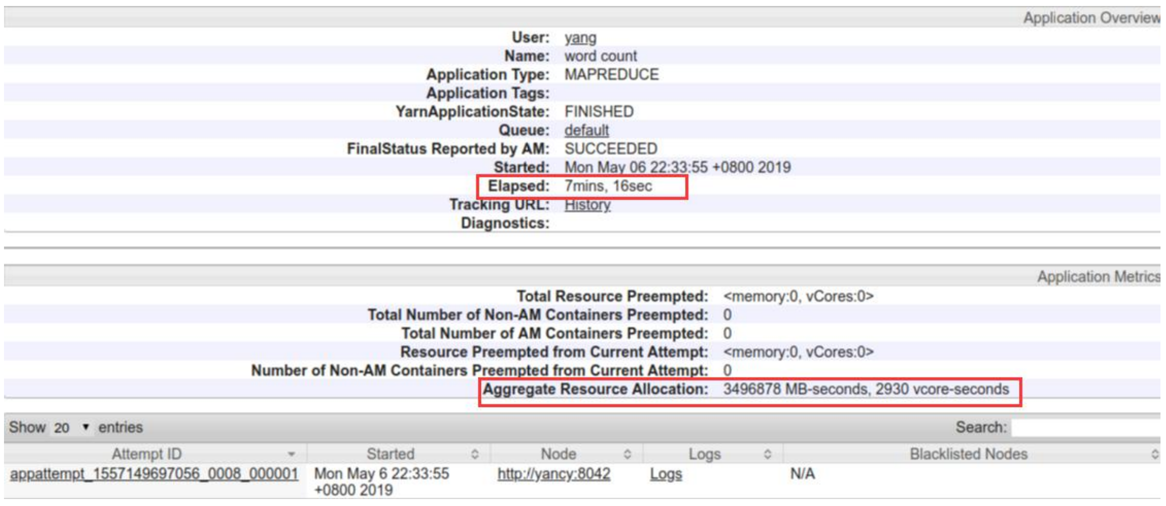
Application Metrics

Total Resource Preempted:	<memory:0, vCores:0>
Total Number of Non-AM Containers Preempted:	0
Total Number of AM Containers Preempted:	0
Resource Preempted from Current Attempt:	<memory:0, vCores:0>
Number of Non-AM Containers Preempted from Current Attempt:	0
Aggregate Resource Allocation:	382513 MB-seconds, 304 vcore-seconds

Show 20 entries Search:

Attempt ID	Started	Node	Logs	Blacklisted Nodes
appattempt_1557149697056_0007_000001	Mon May 6 22:30:22 +0800 2019	http://yancy:8042	Logs	N/A

（6）10000M 文本执行 MapReduce 耗时及资源分配：



Application Overview

User:	yang
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Mon May 06 22:33:55 +0800 2019
Elapsed:	7mins, 16sec
Tracking URL:	History
Diagnostics:	

Application Metrics

Total Resource Preempted:	<memory:0, vCores:0>
Total Number of Non-AM Containers Preempted:	0
Total Number of AM Containers Preempted:	0
Resource Preempted from Current Attempt:	<memory:0, vCores:0>
Number of Non-AM Containers Preempted from Current Attempt:	0
Aggregate Resource Allocation:	3496878 MB-seconds, 2930 vcore-seconds

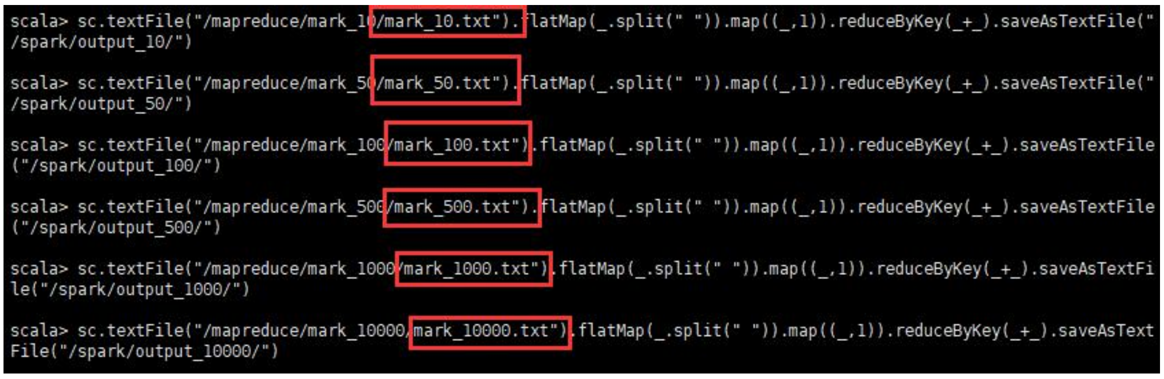
Show 20 entries Search:

Attempt ID	Started	Node	Logs	Blacklisted Nodes
appattempt_1557149697056_0008_000001	Mon May 6 22:33:55 +0800 2019	http://yancy:8042	Logs	N/A

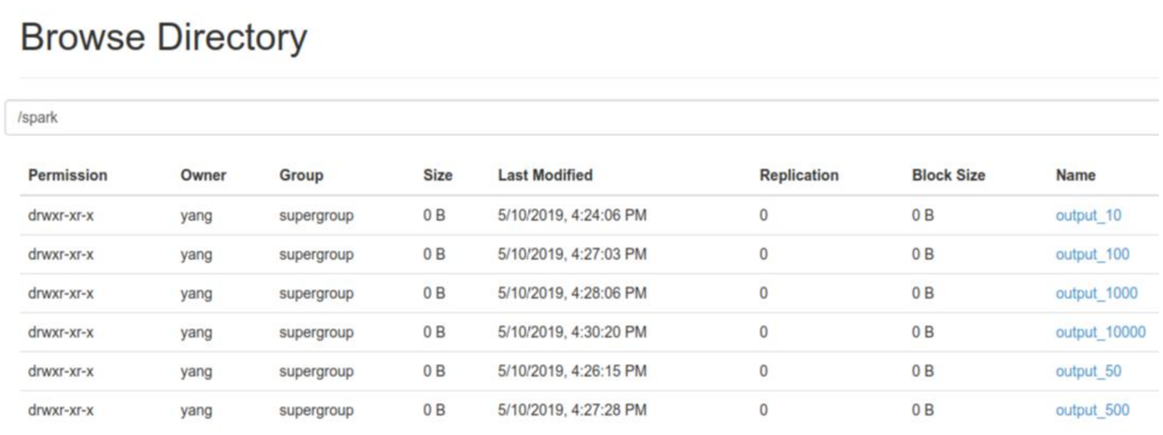
6.2 Spark 执行效率

6.2.1 运行时间

（1）10M、50M、100M、500M、100M、10000M 大小文本执行各自输出结果图：



```
scala> sc.textFile("/mapreduce/mark_10/mark_10.txt").flatMap(_.split(" ")).map((_,1)).reduceByKey(_+_).saveAsTextFile("
/spark/output_10/")

scala> sc.textFile("/mapreduce/mark_50/mark_50.txt").flatMap(_.split(" ")).map((_,1)).reduceByKey(_+_).saveAsTextFile("
/spark/output_50/")

scala> sc.textFile("/mapreduce/mark_100/mark_100.txt").flatMap(_.split(" ")).map((_,1)).reduceByKey(_+_).saveAsTextFile
("/spark/output_100/")

scala> sc.textFile("/mapreduce/mark_500/mark_500.txt").flatMap(_.split(" ")).map((_,1)).reduceByKey(_+_).saveAsTextFile
("/spark/output_500/")

scala> sc.textFile("/mapreduce/mark_1000/mark_1000.txt").flatMap(_.split(" ")).map((_,1)).reduceByKey(_+_).saveAsTextFi
le("/spark/output_1000/")

scala> sc.textFile("/mapreduce/mark_10000/mark_10000.txt").flatMap(_.split(" ")).map((_,1)).reduceByKey(_+_).saveAsText
File("/spark/output_10000/")
```

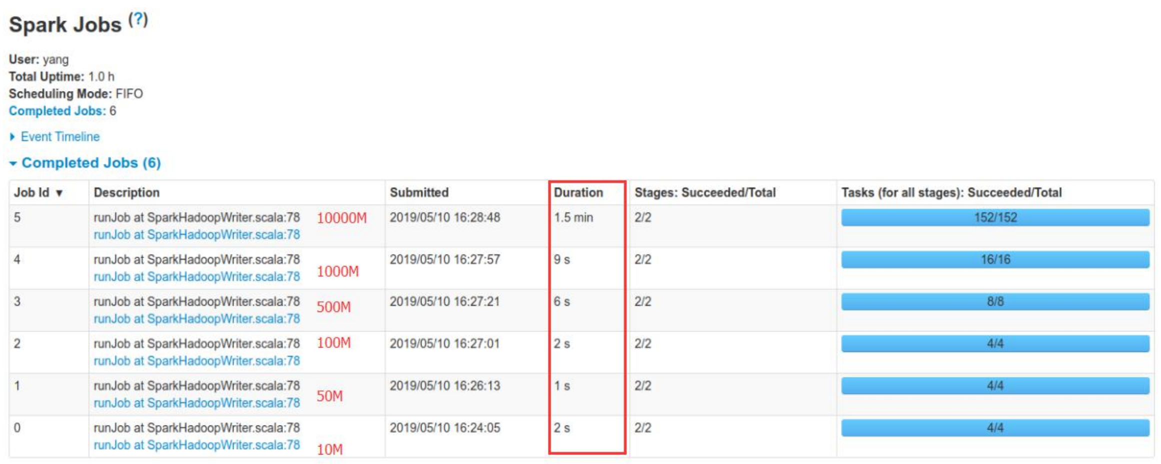


Browse Directory

/spark

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	yang	supergroup	0 B	5/10/2019, 4:24:06 PM	0	0 B	output_10
drwxr-xr-x	yang	supergroup	0 B	5/10/2019, 4:27:03 PM	0	0 B	output_100
drwxr-xr-x	yang	supergroup	0 B	5/10/2019, 4:28:06 PM	0	0 B	output_1000
drwxr-xr-x	yang	supergroup	0 B	5/10/2019, 4:30:20 PM	0	0 B	output_10000
drwxr-xr-x	yang	supergroup	0 B	5/10/2019, 4:26:15 PM	0	0 B	output_50
drwxr-xr-x	yang	supergroup	0 B	5/10/2019, 4:27:28 PM	0	0 B	output_500

（2）10M、50M、100M、500M、100M、10000M 大小文本运行各自总时耗：



Spark Jobs (?)

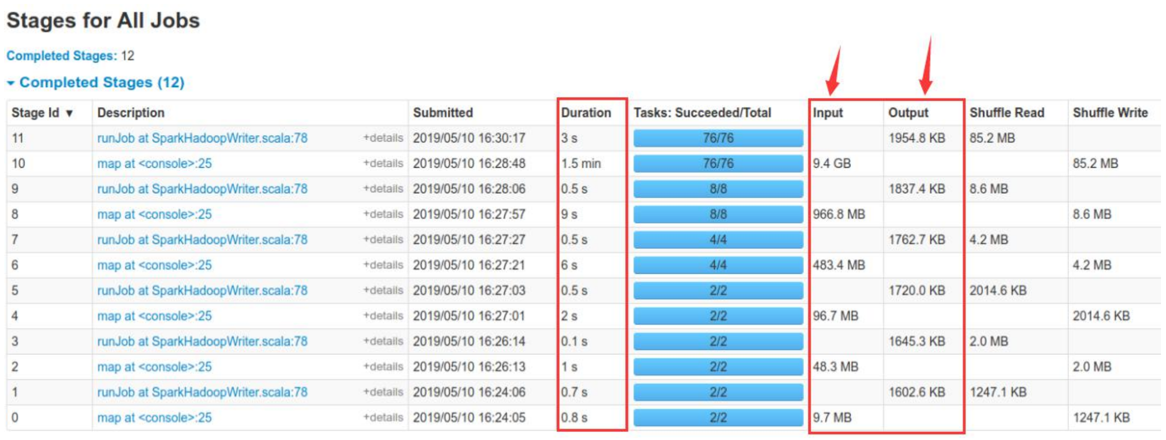
User: yang
Total Uptime: 1.0 h
Scheduling Mode: FIFO
Completed Jobs: 6

▸ Event Timeline

▾ Completed Jobs (6)

Job Id ▾	Description		Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
5	runJob at SparkHadoopWriter.scala:78 runJob at SparkHadoopWriter.scala:78	10000M	2019/05/10 16:28:48	1.5 min	2/2	152/152
4	runJob at SparkHadoopWriter.scala:78 runJob at SparkHadoopWriter.scala:78	1000M	2019/05/10 16:27:57	9 s	2/2	16/16
3	runJob at SparkHadoopWriter.scala:78 runJob at SparkHadoopWriter.scala:78	500M	2019/05/10 16:27:21	6 s	2/2	8/8
2	runJob at SparkHadoopWriter.scala:78 runJob at SparkHadoopWriter.scala:78	100M	2019/05/10 16:27:01	2 s	2/2	4/4
1	runJob at SparkHadoopWriter.scala:78 runJob at SparkHadoopWriter.scala:78	50M	2019/05/10 16:26:13	1 s	2/2	4/4
0	runJob at SparkHadoopWriter.scala:78 runJob at SparkHadoopWriter.scala:78	10M	2019/05/10 16:24:05	2 s	2/2	4/4

6.2.2 输出文本大小



Stages for All Jobs

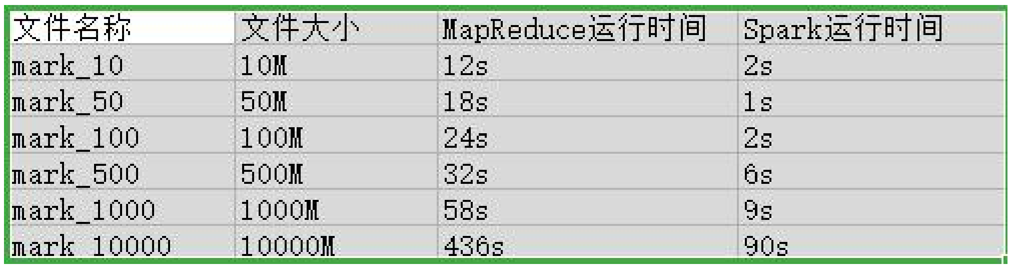
Completed Stages: 12

▾ Completed Stages (12)

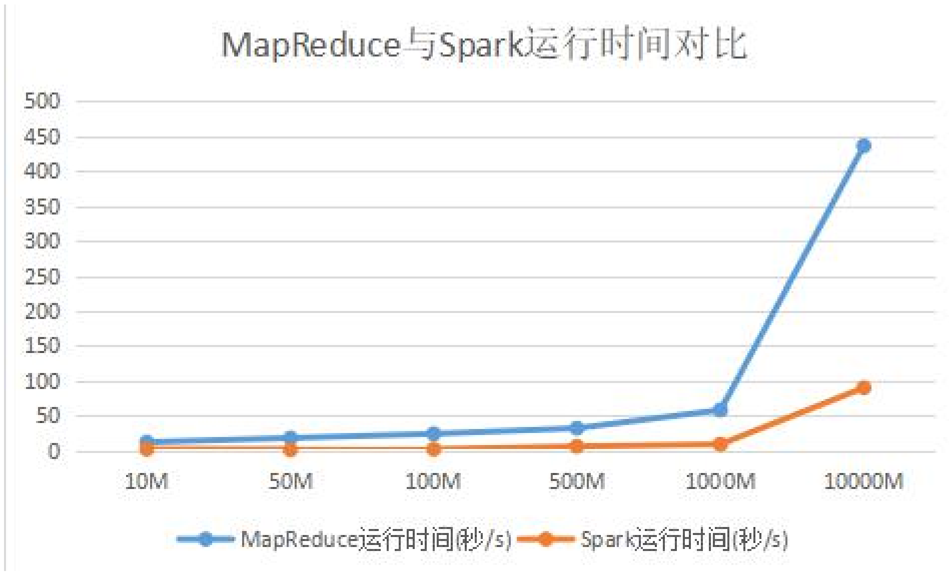
Stage Id ▾	Description		Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
11	runJob at SparkHadoopWriter.scala:78	+details	2019/05/10 16:30:17	3 s	76/76		1954.8 KB	85.2 MB	
10	map at <console>:25	+details	2019/05/10 16:28:48	1.5 min	76/76	9.4 GB			85.2 MB
9	runJob at SparkHadoopWriter.scala:78	+details	2019/05/10 16:28:06	0.5 s	8/8		1837.4 KB	8.6 MB	
8	map at <console>:25	+details	2019/05/10 16:27:57	9 s	8/8	966.8 MB			8.6 MB
7	runJob at SparkHadoopWriter.scala:78	+details	2019/05/10 16:27:27	0.5 s	4/4		1762.7 KB	4.2 MB	
6	map at <console>:25	+details	2019/05/10 16:27:21	6 s	4/4	483.4 MB			4.2 MB
5	runJob at SparkHadoopWriter.scala:78	+details	2019/05/10 16:27:03	0.5 s	2/2		1720.0 KB	2014.6 KB	
4	map at <console>:25	+details	2019/05/10 16:27:01	2 s	2/2	96.7 MB			2014.6 KB
3	runJob at SparkHadoopWriter.scala:78	+details	2019/05/10 16:26:14	0.1 s	2/2		1645.3 KB	2.0 MB	
2	map at <console>:25	+details	2019/05/10 16:26:13	1 s	2/2	48.3 MB			2.0 MB
1	runJob at SparkHadoopWriter.scala:78	+details	2019/05/10 16:24:06	0.7 s	2/2		1602.6 KB	1247.1 KB	
0	map at <console>:25	+details	2019/05/10 16:24:05	0.8 s	2/2	9.7 MB			1247.1 KB

6.3 两者结果对比

MapReduce 与 Spark 在 10M、50M、100M、500M、100M、10000M 大小文本执行各自运行消耗时间对比图：



文件名称	文件大小	MapReduce运行时间	Spark运行时间
mark_10	10M	12s	2s
mark_50	50M	18s	1s
mark_100	100M	24s	2s
mark_500	500M	32s	6s
mark_1000	1000M	58s	9s
mark_10000	10000M	436s	90s



7　从软件体系架构角度解释分析实验结果

7.1　运行时间差异

由上小节实验可得出，MapReduce 整体的运行时间要大于 Spark，当文本大小较小时（小于 500M），MapReduce 与 Spark 上的运行时间差异不明显，而随着文本大小的增大（大于 500M），MapReduce 与 Spark 上的运行时间差距越来越明显。

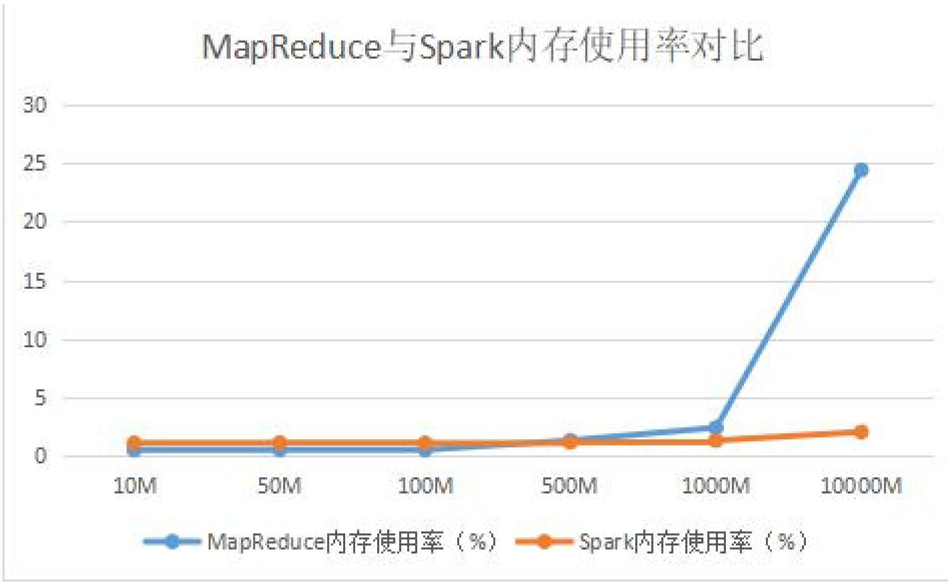
这是由于 Hadoop 生态圈包含内容较多，每次任务启动时首先需要加载所需的资源包，然后缓慢地发起任务，所以导致整体计算时间长、性能差。而 Spark 的计算速度快是因为 Spark 把中间数据放到内存中，迭代运算效率高。MapReduce 在 shuffle 的时候，计算结果需要保存到磁盘上，这样会影响整体速度，而 Spark 利用 RDD 技术，计算在内存中进行。且 Spark 支持 DAG 图的分布式并行计算的编程框架，减少了迭代过程中数据的落地，提高了处理效率。

7.2　CPU 使用率

由上小节实验得出，当文本大小在 500M 左右时，MapReduce 占用的 CPU 资源基本处于饱和。而 Spark 的 CPU 使用率并没有伴随文本文件增大而大幅度上涨。

7.3　内存使用率

由上小节实验得出，随着文本数据的增大，MapReduce 慢慢需要占用较多的内存，而 Spark 随着数据量增大而内存需求不明显。



7.4　容错性

Spark 容错性比 MapReduce 高。Spark 引进了弹性分布式数据集 RDD，它是分布在一组节点中的只读对象集合，这些集合是弹性的，如果数据集一部分丢失，则可以对它们进行重建。另外在 RDD 计算时可以通过 CheckPoint 来实现容错。

7.5　通用性

MapReduce 只提供了 Map 和 Reduce 两种操作，而 Spark 提供的数据集操作类型有很多，大致分为转换操作（Transformations）和行动操作（Actions）两大类，Spark 更加通用。

由于 Hadoop 生态圈庞大，对大数据支持性很好，如果需要处理的数据量大，并且用户希望计算出的数据可以被存储和二次计算或数据挖掘，那么可以考虑 MapReduce。Spark 由于架构层面设计不同，所以对于 CPU、内存的使用率一直保持较低状态，它可以用于海量数据分析等用途。

8　实验总结

MapReduce 采用了多进程模型，虽然在 server 端二者都采用了相同的并发模型，但是在任务级别上，MapReduce 的多进程模型会消耗过多的启动时间，在运行低延迟类型的任务上表现不如 Spark。

而 Spark 采用了多线程模型，Spark 可以用 Scala 编程，Scala 适合并行计算，与 Java 相比，极大的减少代码量。Spark 计算速度胜于 MapReduce 还在于中间结果是缓存在内存而不是直接写入到磁盘。MapReduce 每次计算先写磁盘，下次计算先从磁盘读，计算结果再写磁盘，如此往复。Spark 为每个应用程序在 worker 上开启一个进程，而一个 Job 中的 Task 会在同一个线程池中运行，而 MapReduce 的计算模型是每个 Task(Mapper 或者 Reducer）都是一个单独的进程，启动停止进程非常昂贵，同时，进程间的数据共享也不能基于内存，只能是 HDFS。Spark 克服了 MapReduce 的这些不足，不过其带来的缺点是难以细粒度地控制每个任务的占用资源。

在数据的读取上，如果任务复杂的话，MapReduce 在读写磁盘文件时会在 IO 上花费大量的时间，而 Spark 在这一点上消除了冗余的 HDFS 读写，速度更快。MapReduce 每一次 shuffle 必将连接着一次完成 Map 和 Reduce 操作，而 Spark

基于 RDD 提供了丰富的算子操作，避免了冗余的 Map 和 Reduce 阶段。MapReduce 每次启动一个任务就要启动一次基于进程的 JVM，而 Spark 基于线程的机制，只会在启动执行器时启动一次 JVM，执行任务时复用执行器中的线程。由于启动 JVM 时要耗费大量的时间，如果任务数一多，MapReduce 频繁地启动 JVM 会耗费大量的时间。在上节中的实验环节，我们也可以看到这一点。

总的来说，现在的大数据应用中对更复杂的多重处理和低延迟的交互式查询需求日益增长，MapReduce 计算模型在上述两类任务上因为其架构的原因，执行效率并不快，Spark 则在某种程度上克服了这些困难，其基于内存的计算提升了计算速度，基于 RDD 结构提升了容错性能，提供了各种计算框架。不过其也有一些局限和缺点，主要体现在耗费的内存资源过大、稳定性不足、无法细粒度地进行资源分配等。

9 实验遇到的问题

（1）问题一：Namenode，Datanode 未正常启动。

解决：由于多次初始化 Damenode 导致 ID 不对应。关闭 hadoop，删除 data,name,logs,tmp 目录下所有文件，然后 hadoop namenode -format 重启初始化，最后开启 hadoop。

（2）问题二：在 hadoop 安装完后，依次执行 hadoop namenode -format，start-dfs.sh，start-yarn.sh 之后，打开浏览器输入 http://localhost:50070/无反应，查了网上很多说法都没有解决，最后发现是因为电脑连接了 VPN。

解决：直到关掉 VPN 之后就可以了。

（3）问题三：显示错误信息，hedulerImpl: Initial job has not accepted any resources; check your cluster UI to ensure that workers are registered and have sufficient resources 19/05/07 12:56:49 WARN scheduler.TaskSchedulerImpl: Initial job has not accepted any resources; check your cluster UI to ensure that workers are registered and have sufficient resources。

解决：执行语句少了一个斜杠，即为 saveAsTextFile("/spark/output/")。