深入理解大数据-大数据处理与编程实践

Ch.4. Hadoop系统安装运行与程序开发

南京大学计算机科学与技术系

主讲人：黄宜华、顾荣

鸣谢：本课程得到Google（北京）与Intel公司
中国大学合作部精品课程计划资助

Ch.4. Hadoop系统安装运行与程序开发

1. 单机Hadoop系统安装基本步骤
2. 集群Hadoop系统安装基本步骤
3. Hadoop集群远程作业提交与执行
4. Hadoop MapReduce程序开发
5. 实验2：安装单机Hadoop系统与WordCount词频统计

Hadoop系统运行的软件环境

- **Linux操作系统**

  如RHELS 7.0 (Red Hat Enterprise Linux Server 7.0)

  - 直接安装Linux
  - Window下安装Linux虚拟机

- **SSH（Secure Shell）**

  主要用于远程管理Hadoop节点以及Hadoop节点间的安全共享访问

- **Java**

  如Java1.7

Hadoop系统的安装方式

- **单机方式**

  在一台运行Linux或Windows下虚拟Linux的单机上安装运行Hadoop系统

- **单机伪分布方式**

  在一台运行Linux或Window下虚拟Linux的单机上，用伪分布方式，用不同的java进程模拟分布运行中的NameNode、DataNode、JobTracker、TaskTracker等各类节点

- **集群分布模式**

  在一个真实的集群环境下安装运行Hadoop系统，集群的每个节点可以运行Linux或Window下的虚拟Linux。

  单机和伪分布模式下编写调试完成的程序不需修改即可在真实的分布式Hadoop集群下运行（但通常需要修改配置）

1. 单机Hadoop系统安装基本步骤

基本安装步骤

- 安装JDK
- 下载安装Hadoop
- 配置SSH
- 配置Hadoop的环境
- 格式化HDFS文件系统
- 启动Hadoop环境
- 运行程序测试
- 查看集群状态

单机和单机伪分布方式安装过程

1. 单机操作系统安装

在单机上安装Linux或Window下虚拟Linux，假设安装后机器名为Siler

2. 安装SSH

如果安装RHELS 7.0 , 确保软件安装时把SSH选上；如果安装Linux时没有安装SSH，则需要另行安装SSH

3. 安装Java

下载和安装Java，例如，将java安装在/usr/java目录下

单机和单机伪分布方式安装过程

4. 创建用户

为Hadoop创建一个专门的用户组如hadoop-user，然后在该用户组下创建Hadoop用户。可在安装系统的时候创建，也可以在安装好之后用如下命令创建：

```
[root@Siler ~]# groupadd  hadoop-user
[root@Siler ~]# useradd -g hadoop-user -d /home/hadoop  -s /bin/bash  hadoop
```

“hadoop”是所创建的用户名, -d指明“ hadoop”用户的home目录是/home/hadoop，
-s指定用户登入后所使用的shell版本。

```
[root@Siler ~]# passwd hadoop [给用户hadoop设置口令]
```

单机和单机伪分布方式安装过程

5. 解压安装Hadoop

- 到Hadoop官网下载hadoop
- 建立安装目录

  [hadoop@Siler ~] mkdir ~/hadoop_installs
- 把hadoop-2.7.1.tar.gz放在这里，然后解压：

  [hadoop@Siler hadoop_installs]$ tar –zxvf hadoop-2.7.1.tar.gz

单机和单机伪分布方式安装过程

Hadoop版本信息

根据Apache Hadoop官方提供的release，目前有以下版本可下载：

2.6.x：hadoop-2.6.0/，hadoop-2.6.1/，hadoop-2.6.2/，hadoop-2.6.3/

2.7.x：hadoop-2.7.1/，hadoop-2.7.2/, hadoop-2.7.3/

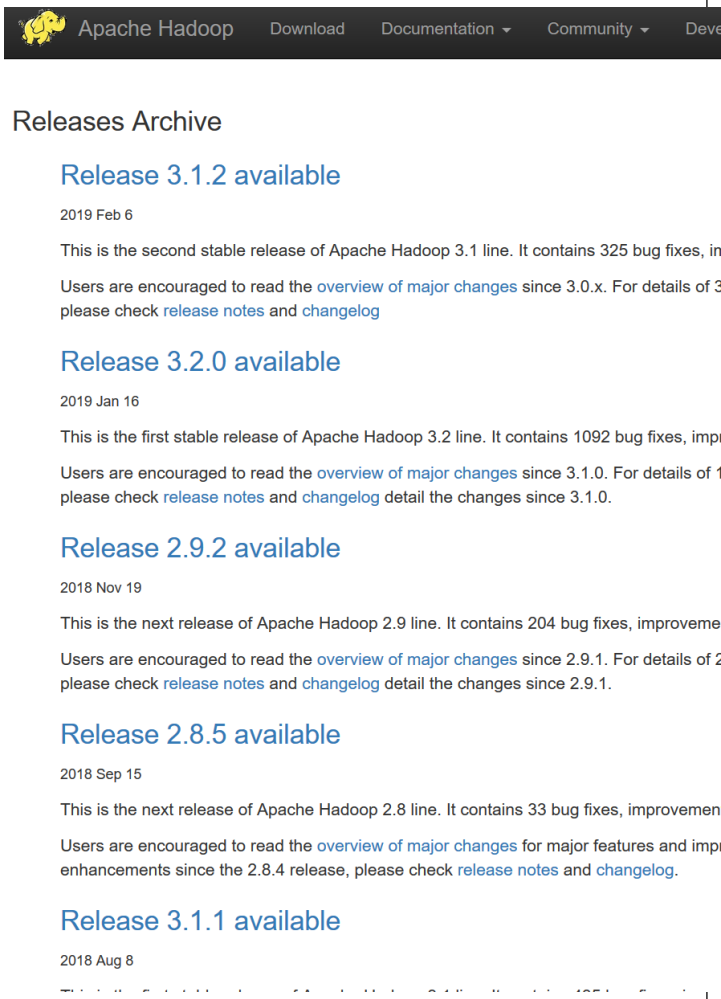
其中2.7.1 是稳定版本，发布于2015年7月6日，官方的说明是：“It is stable and includes new developer and user-facing incompatibilities, features, and major improvements.”。

到2016年9月3日，在2.7.3 的基础上发布了hadoop-3.0.0-alpha1

3.x：hadoop-3.1.1/，hadoop-3.1.2/

https://hadoop.apache.org/release.html

目前我们开课及对外提供服务的Hadoop 集群是2.7.1版本的



单机和单机伪分布方式安装过程

6. 配置环境变量

- 进入到“*hadoop*”用户下

  [root@Siler ~]# su - *hadoop* [注意中间的”-”不要丢]

  [*hadoop*@Siler ~]$

- 编辑~/.bash_profile文件(用vi 或gedit)

  vi ~/.bash_profile

- 设置如下环境变量：

```
PATH=$PATH:$HOME/bin
export  JAVA_HOME=/usr/java/java-1.7
export  HADOOP_HOME=/home/hadoop/hadoop_installs/hadoop-2.7.1
export  PATH=$JAVA_HOME/bin:$HADOOP_HOME/bin:$PATH
export  CLASSPATH=$JAVA_HOME/lib:.
```

单机和单机伪分布方式安装过程

7. 免密码SSH访问配置

在伪分布模式下(集群分布模式更需要这个设置过程)，为了实现免密码SSH登陆连接，需要进行相应的配置。方式是创建一个认证文件，然后用public key实现免密码的登录连接。过程如下：

- 执行命令产生认证文件(请先进入到 ~/.ssh 目录)

```
[hadoop@Siler ~]$ ssh-keygen -t rsa -P ""
```

一直敲回车，然后将在/home/hadoop/.ssh目录下生成id_rsa认证文件

- 将该文件复制到名为authorized_keys的文件

```
[hadoop@Siler ~] cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

- 然后测试一下看看能不能登录:

```
[hadoop@Siler ~] ssh localhost
```

单机和单机伪分布方式安装过程

什么是SSH?

- SSH(Secure Shell)，是建立在应用层和传输层基础上的安全协议。
- 传统的网络服务程序，如FTP、POP和Telnet本质上都是不安全的；它们在网络上用明文传送数据、用户帐号和用户口令，很容易受到中间人（man-in-the-middle）攻击方式的攻击。
- 而SSH是目前较为可靠的、专为远程登录会话和其他网络服务提供安全性的协议。利用SSH协议可以有效防止远程管理过程中的信息泄露问题。通过**SSH**可以对所有传输的数据进行加密，也能够防止DNS欺骗和IP欺骗。
- SSH另一项优点是其传输的数据是经过压缩的，所以可以加快传输的速度。SSH有很多功能，它既可以代替Telnet，又可以为FTP、POP、PPP提供一个安全的登陆会话“通道”。
- Hadoop使用SSH保证在远程管理Hadoop节点和节点间用户共享访问时的安全性。

单机和单机伪分布方式安装过程

8. 修改hadoop配置文件

Hadoop的配置文件存放在hadoop安装目录下的etc/hadoop目录中，主要有以下几个配置文件要修改：

- hadoop-env.sh：Hadoop环境变量设置
- core-site.xml：主要完成NameNode的IP和端口设置
- hdfs-site.xml：主要完成HDFS的数据块副本等参数设置
- yarn-site.xml ：完成ResouceManager的主机名和服务等配置
- mapred-site.xml：主要完成mapreduce framework设置和jobhistory server的设置

单机和单机伪分布方式安装过程

9.格式化NameNode

- 执行Hadoop的bin文件夹中的格式化命令：

```
[hadoop@Siler ~]$ bin/hdfs namenode -format
```

如果格式化成功，会返回一堆有关NameNode的启动信息，其中会有一句“…. has been successfully formatted.”

单机和单机伪分布方式安装过程

10. 启动HDFS和MapReduce

- 执行以下命令启动HDFS和MapReduce

  [hadoop@Siler ~]$ start-all.sh

- 用JPS命令检查一下是否正常启动：

  [hadoop@Siler ~]$ jps

  显示以下各进程信息则说明HDFS和MapReduce都已正常启动：

  4706 ResourceManager
  4582 SecondaryNameNode
  4278 NameNode
  4413 DataNode
  4853 NodeManager
  4889 Jps

11. 停止HDFS和MapReduce

- 执行以下命令启动HDFS和MapReduce

  [hadoop@Siler~]$ stop-all.sh

单机和单机伪分布方式安装过程

12. 运行测试

- 在Linux文件系统下（如/root/test)创建两个文本数据文件:

  file1.txt：hello hadoop hello world

  file2.txt： goodbye hadoop

- 将文件复制到HDFS文件系统中：

  [hadoop@Siler ~]$ bin/hdfs dfs -copyFromLocal /root/test test-in

  test-in是在HDFS中建立的一个数据数据目录

- 运行hadoop安装包中自带的WordCount程序进行测试：

  [hadoop@Siler ~]$ hadoop jar hadoop-mapreduce-examples-2.7.1.jar wordcount test-in test-out

  其中test-out只能由程序创建，不能事先存在,hadoop-mapreduce-examples-2.7.1.jar在${HADDOP_HOME}/share/hadoop/mapreduce下

2. 集群Hadoop系统基本安装步骤

集群分布方式安装过程

1. 操作系统安装

在每个节点上安装Linux或Window下虚拟Linux，假设安装后机器名为Master。

2. 安装SSH

如果安装RHELS 7.0 , 确保软件安装时把SSH选上；如果安装Linux时没有安装SSH，则需要另行安装SSH

3. 安装Java

下载和安装Java，将java安装在/usr/java目录下

集群分布方式安装过程

4. 创建用户

为Hadoop创建一个专门的用户组如hadoop-user，然后在该用户组下创建Hadoop用户。可在安装系统的时候创建，也可以在安装好之后用如下命令创建：

```
[root@ Master ~]# groupadd  hadoop-user
[root@ Master ~]# useradd -g hadoop-user -d /home/hadoop  -s /bin/bash  hadoop
```

“hadoop”是所创建的用户名, -d指明“ hadoop”用户的home目录是/home/hadoop，-s指定用户登入后所使用的shell版本。

```
[root@ Master ~]# passwd hadoop [给用户hadoop设置口令]
```

1). 在真实集群分布模式下，要求每个节点使用相同的用户名，比如，可以使用“hadoop”作为所有节点上统一的用户名。

2). 并且要求在所有节点上安装的hadoop系统具有完全一致的目录结构。

集群分布方式安装过程

5. 在主节点上解压安装Hadoop

- 到Hadoop官网下载hadoop
- 建立安装目录

  [hadoop@ Master ~] mkdir ~/hadoop_installs
- 把hadoop-2.7.1.tar.gz放在这里，然后解压：

  [hadoop@ Master hadoop_installs]$ tar –zxvf hadoop-2.7.1.tar.gz

注：这个过程仅需在主节点上完成，然后安装好的Hadoop系统可被复制到所有从节点

集群分布方式安装过程

6. 配置环境变量(每个节点都必须做）

- 进入到“*hadoop*”用户下

  [root@Siler ~]# su - *hadoop* [注意中间的”-”不要丢]

  [*hadoop*@Siler ~]$

- 编辑~/.bash_profile文件(用vi 或gedit)

  vi ~/.bash_profile

- 设置如下环境变量：

```
PATH=$PATH:$HOME/bin
export  JAVA_HOME=/usr/java/java-1.7
export  HADOOP_HOME=/home/hadoop/hadoop_installs/hadoop-2.7.1
export  PATH=$JAVA_HOME/bin:$HADOOP_HOME/bin:$PATH
export  CLASSPATH=$JAVA_HOME/lib:.
```

集群分布方式安装过程

7. 免密码SSH访问配置

在真实集群分布模式下更需要这个设置过程，为了实现节点间相互的免密码SSH访问，需要进行相应的配置。方式是创建一个认证文件，然后用public key实现免密码的登录连接。过程如下：

- 执行命令产生认证文件

[hadoop@ Master ~]$ ssh-keygen -t rsa -P '"

敲回车,然后将在/home/hadoop/.ssh目录下生成id_rsa认证文件

- 将该文件复制为名为authorized_keys的文件

[hadoop@ Master ~]$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys

- 将authorized_keys文件复制到所有节点上

[hadoop@Master ~]$ scp authorized_keys

[从节点主机名或IP]:/home/hadoop/.ssh

集群分布方式安装过程

8.在主节点上修改hadoop配置文件

Hadoop的配置文件存放在hadoop安装目录下的etc/hadoop目录中，主要有以下几个配置文件要修改：

- hadoop-env.sh：Hadoop环境变量设置
- core-site.xml：主要完成NameNode的IP和端口设置
- hdfs-site.xml：主要完成HDFS的数据块副本等参数设置
- yarn-site.xml ：完成ResouceManager的主机名和服务等配置
- mapred-site.xml：主要完成mapreduce framework设置和jobhistory server的设置
- slaves：完成Slaves节点IP设置

注：这个过程仅需在主节点上进行，然后将随着主机上安装好的Hadoop目录一起复制到所有从节点

集群分布方式安装过程

8.在主节点上修改hadoop配置文件

slaves：列出所有从节点的主机名

在NameNode和DataNode节点上进行主机名和IP解析配置

修改每台机器的/etc/hosts设置：

- 若为NameNode，则需要在hosts文件中添加集群中所有节点的IP地址及其对应的主机名
- 若为DataNode，则只需要在文件中添加本机和NameNode的IP地址及对应的主机名

集群分布方式安装过程

9.复制Hadoop系统

- 将在主节点安装好的Hadoop系统目录复制到每一个从节点上：

```
[hadoop@ Master ~]$ scp -r /home/hadoop/hadoop-installs
                                [从节点主机名或IP]:/home/hadoop/
```

这样可以避免对每一个从节点重复进行Hadoop系统安装。

集群分布方式安装过程

10.格式化NameNode

- 执行Hadoop的bin文件夹中的格式化命令：

  [hadoop@ Master ~]$ bin/hdfs namenode -format

  如果格式化成功，会返回一堆有关NameNode的启动信息，其中会有一句“…. has been successfully formatted.”

集群分布方式安装过程

11. 启动HDFS和MapReduce

- 执行以下命令启动HDFS和MapReduce

  [hadoop@ Master ~]$ start-all.sh

- 用JPS命令检查一下是否正常启动：

  [hadoop@ Master ~]$ jps

  显示以下各进程信息则说明HDFS和MapReduce都已正常启动：

  4706 ResourceManager
  4582 SecondaryNameNode
  4278 NameNode
  4413 DataNode
  4853 NodeManager
  4889 Jps

12. 停止HDFS和MapReduce

- 执行以下命令启动HDFS和MapReduce

  [hadoop@ Master ~]$ stop-all.sh

集群分布方式安装过程

13. 运行测试

- 在Linux文件系统下（如/root/test)创建两个文本数据文件:

  file1.txt：hello hadoop hello world

  file2.txt：goodbye hadoop

- 将文件复制到HDFS文件系统中：

  [hadoop@ Master ~]$ bin/hdfs dfs -copyFromLocal /root/test test-in

  test-in是在HDFS中建立的一个数据数据目录

- 运行hadoop安装包中自带的WorldCount程序进行测试：

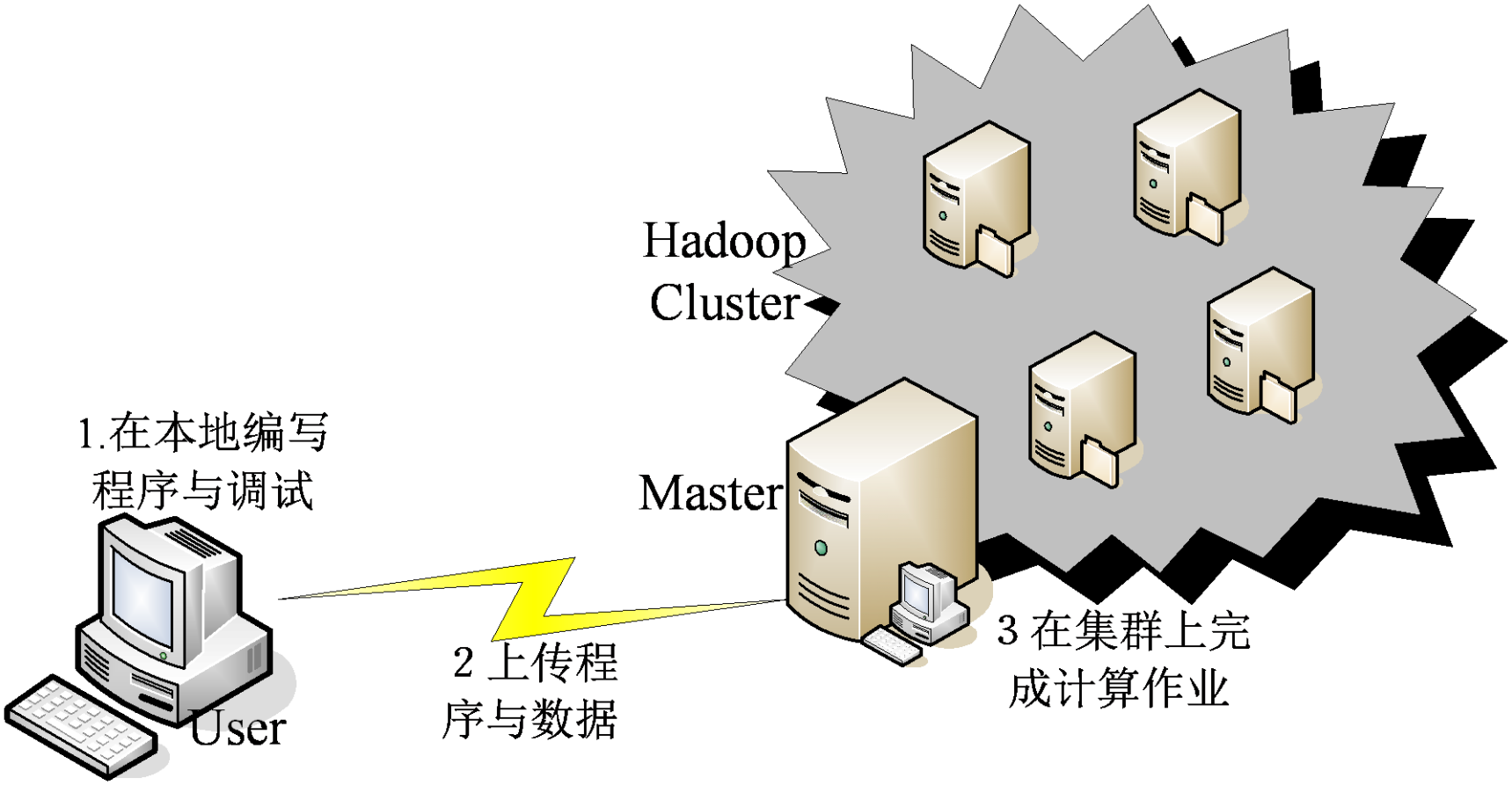
  [hadoop@ Master ~]$ hadoop jar hadoop-2.7.1-examples.jar

  wordcount test-in test-out

  其中test-out只能由程序创建，不能事先存在, hadoop-mapreduce-examples-2.7.1.jar在${HADDOP_HOME}/share/hadoop/mapreduce下

3. Hadoop集群远程作业提交与执行

程序开发与提交作业基本过程



集群分布方式下远程提交作业

1. 本地完成程序编写和调试

在自己本地安装了单机或伪分布Hadoop系统的机器上，完成程序编写和调试

2.创建用户账户

为了能访问Hadoop集群提交作业，需要为每个程序用户创建一个账户，获取用户名、密码等信息。

集群分布方式下远程提交作业

3.将数据和程序传送到Hadoop集群

- 准备好数据和程序目录

  例如：

  me@local:~/workspace$ ls -R wordcount

  wordcount:

  wordcount.jar

  wordcount/files: file01.txt file02.txt

- 用scp命令传送至Hadoop平台主机上：

  me@local:~/workspace$ scp -r wordcount

  username@Master :workspace/wordcount

  username@ Master's password: [在此输入您的密码]

集群分布方式下远程提交作业

4.用SSH命令远程登录到Hadoop集群

```
me@local:~/workspace$ ssh username@Master
username@Master's password: [在此输入您的密码]
```

5.将数据复制到HDFS中

- 进入到程序包所在目录：

```
username@Master:~$ cd workspace/wordcount
username@Master:~/workspace/wordcount$ ls
files  wordcount.jar
```

- 用hadoop dfs –put命令将数据从Linux文件系统中复制到HDFS：

```
username@Master:~/workspace/wordcount$ hdfs dfs -put files test-in
```

集群分布方式下远程提交作业

6.用hadoop jar命令向Hadoop提交计算作业

```
username@Master:~/workspace/wordcount$
                    hadoop jar wordcount.jar  test-in  test-out
```

这里的test-in为被统计的文本文件的目录，test-out为指定的输出结果的目录，注意test-out目录事先不能存在，若存在需要先删除。

集群分布方式下远程提交作业

7. 查看运行结果

- 查看test-out目录，统计结果被输出到文件test-out/part-r-00000中

```
username@Master:~/workspace/wordcount$ hdfs dfs -ls test-out
Found 2 items
drwxr-xr-x   - hadoopusr supergroup    0 2010-05-23 20:29 /user/hadoopusr/test-out/_SUCCESS
-rw-r--r--   1 hadoopusr supergroup   35 2010-05-23 20:30 /user/hadoopusr/test-out/part-r-00000
```

- 查看计算结果

```
username@Master:~/workspace/wordcount$ hdfs dfs -cat test-out/part-r-00000
GoodBye         1
Hadoop          2
Hello           2
World           1
```

- 也可以把文件从HDFS中复制到Linux文件系统中查看

```
username@Master:~/workspace/wordcount$ hdfs dfs -get test-out/part-r-00000  test-out.txt
username@Master:~/workspace/wordcount$ vi test-out.txt
GoodBye         1
Hadoop          2
Hello           2
World           1
```

集群分布方式下远程提交作业

8. 用Hadoop的Web界面查看Hadoop集群和作业状态

- 在浏览器中打开http:// NameNode节点IP:50070/. 可看到HDFS的基本信息

Summary

Security is off.

Safemode is off.

54 files and directories, 35 blocks = 89 total filesystem object(s).

Heap Memory used 68.97 MB of 478.5 MB Heap Memory. Max Heap Memory is 889 MB.

Non Heap Memory used 38.92 MB of 39.94 MB Commited Non Heap Memory. Max Non Heap Memory is 130 MB.

Configured Capacity:	492.03 GB
DFS Used:	2.23 MB (0%)
Non DFS Used:	102.31 GB
DFS Remaining:	389.71 GB (79.21%)
Block Pool Used:	2.23 MB (0%)
DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 0.00% / 0.00%
Live Nodes	1 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Total Datanode Volume Failures	0 (0 B)

集群分布方式下远程提交作业

8. 用Hadoop的Web界面查看Hadoop集群和作业状态

- 打开页面中的Namenode Logs链接，可以查看到大量的日志文件，每个都可以打开查看其内容

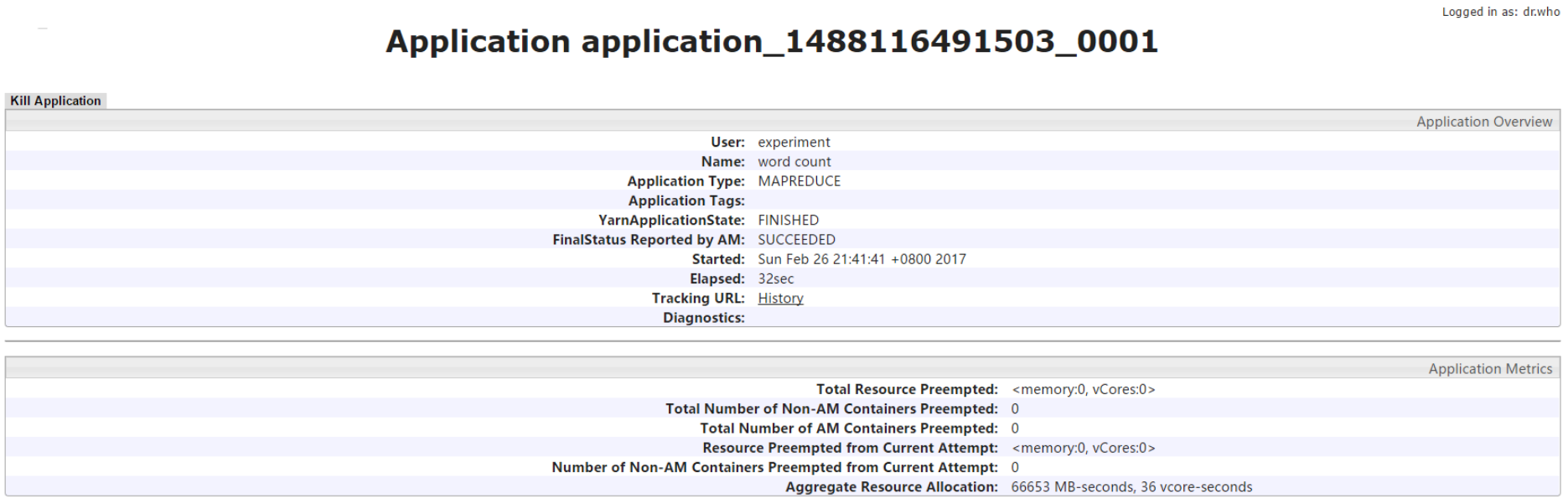
Directory: /logs/

README.txt	5903 bytes	Feb 26, 2017 8:35:24 AM
SecurityAuth-experiment.audit	0 bytes	Dec 29, 2016 9:26:07 AM
hadoop-experiment-all-simple36.out	767 bytes	Jan 13, 2017 11:14:12 PM
hadoop-experiment-balancer-simple36.log	773 bytes	Feb 26, 2017 9:23:32 AM
hadoop-experiment-balancer-simple36.out	1273 bytes	Feb 26, 2017 9:23:32 AM
hadoop-experiment-datanode-simple36.log	1776561 bytes	Feb 26, 2017 9:42:16 PM
hadoop-experiment-datanode-simple36.out	1273 bytes	Feb 26, 2017 8:48:02 PM
hadoop-experiment-datanode-simple36.out.1	1273 bytes	Feb 26, 2017 8:46:13 PM
hadoop-experiment-datanode-simple36.out.2	1273 bytes	Feb 26, 2017 8:42:41 PM
hadoop-experiment-datanode-simple36.out.3	1464 bytes	Feb 25, 2017 5:41:11 PM
hadoop-experiment-datanode-simple36.out.4	1464 bytes	Feb 25, 2017 5:35:26 PM
hadoop-experiment-datanode-simple36.out.5	1464 bytes	Jan 10, 2017 4:25:10 PM
hadoop-experiment-jobtracker-simple36.out	1636 bytes	Jan 13, 2017 11:14:23 PM
hadoop-experiment-namenode-simple36.log	3561951 bytes	Feb 26, 2017 10:03:22 PM
hadoop-experiment-namenode-simple36.out	5511 bytes	Feb 26, 2017 9:25:59 PM
hadoop-experiment-namenode-simple36.out.1	1273 bytes	Feb 26, 2017 8:46:09 PM
hadoop-experiment-namenode-simple36.out.2	1273 bytes	Feb 26, 2017 8:42:37 PM
hadoop-experiment-namenode-simple36.out.3	1464 bytes	Feb 25, 2017 5:41:07 PM
hadoop-experiment-namenode-simple36.out.4	1464 bytes	Feb 25, 2017 5:35:22 PM
hadoop-experiment-namenode-simple36.out.5	5702 bytes	Jan 10, 2017 4:29:07 PM

集群分布方式下远程提交作业

8.用Hadoop的Web界面查看Hadoop集群和作业状态

- 点击一个作业可以查看作业的详细信息



Logged in as: dr.who

Application application_1488116491503_0001

Kill Application

Application Overview

User:	experiment
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
YarnApplicationState:	FINISHED
FinalStatus Reported by AM:	SUCCEEDED
Started:	Sun Feb 26 21:41:41 +0800 2017
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:	66653 MB-seconds, 36 vcore-seconds

4. Hadoop MapReduce程序开发

开发环境与工具

在程序员本地的单机Hadoop系统上进行程序设计与调试，然后上载到Hadoop集群上运行。

IntelliJ IDEA开发环境

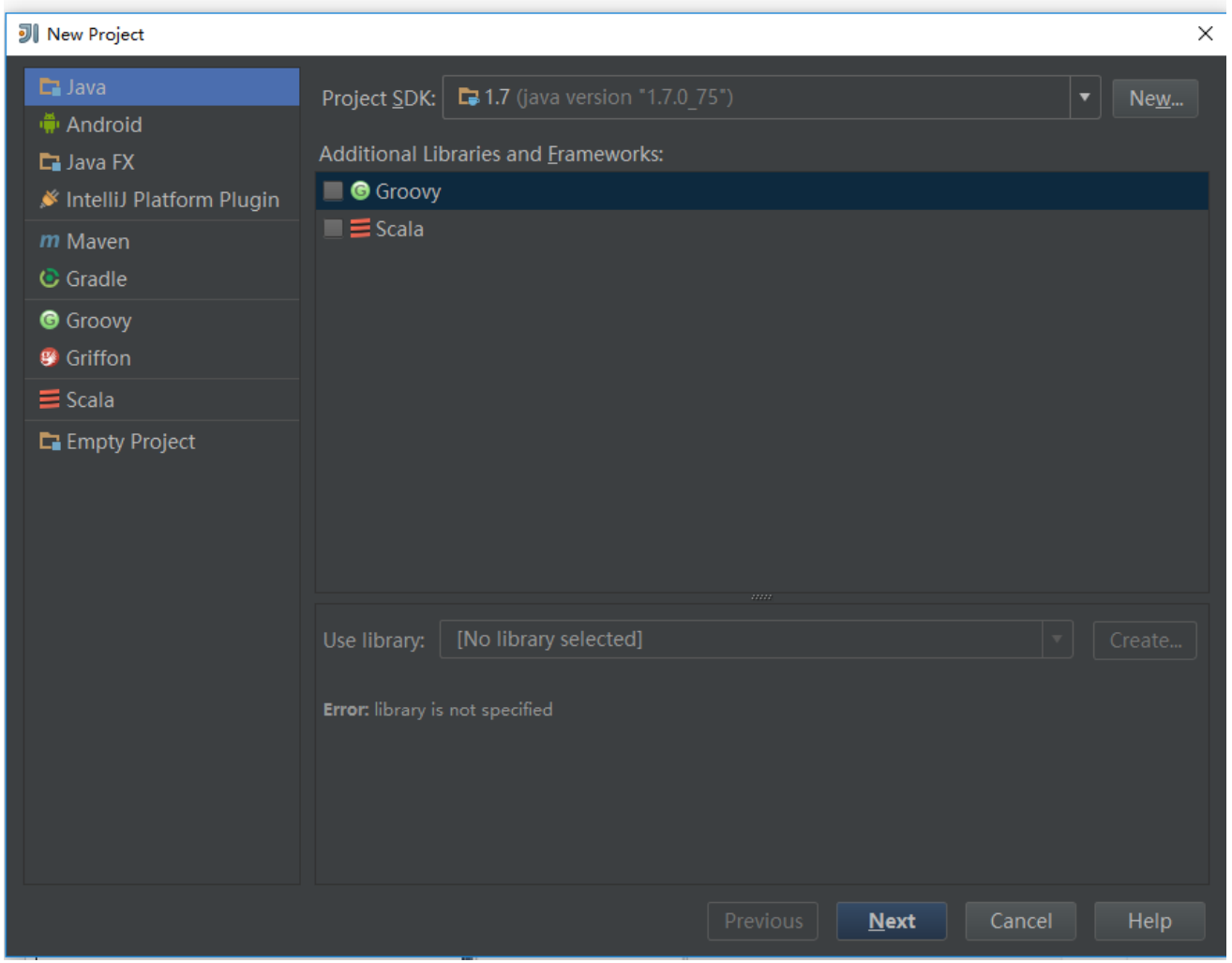
IntelliJ IDEA是一款非常流行的软件集成开发环境（IDE），可以提供对Java应用的编程开发所需要的完整工具平台。

IntelliJ IDEA官方网站： https://www.jetbrains.com/idea/

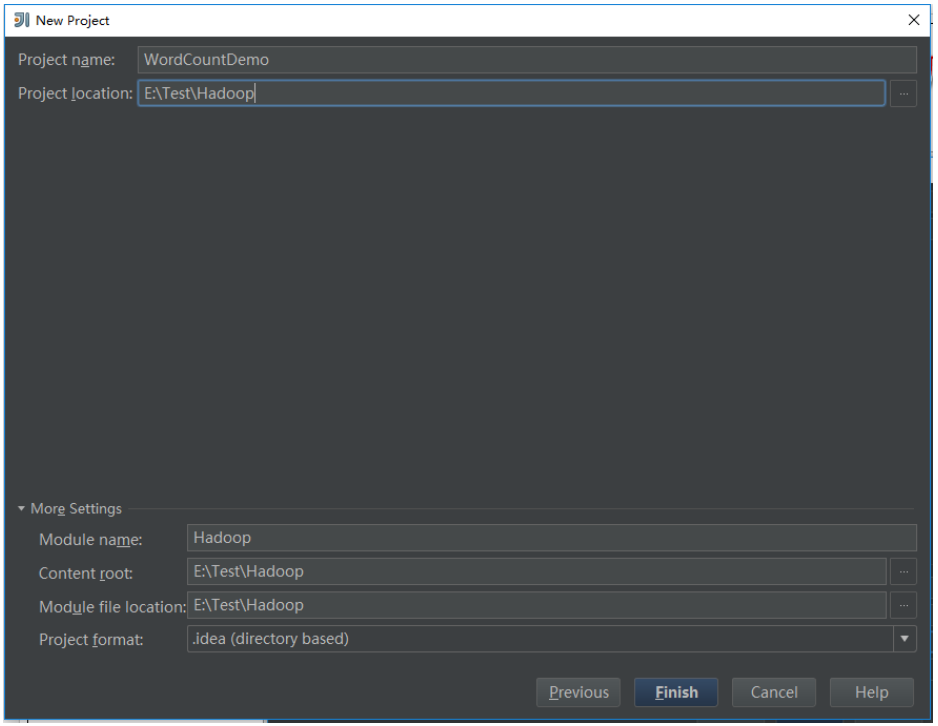
可以下载Linux版本的IntelliJ IDEA Community版开发包，并安装在本地的Linux系统中

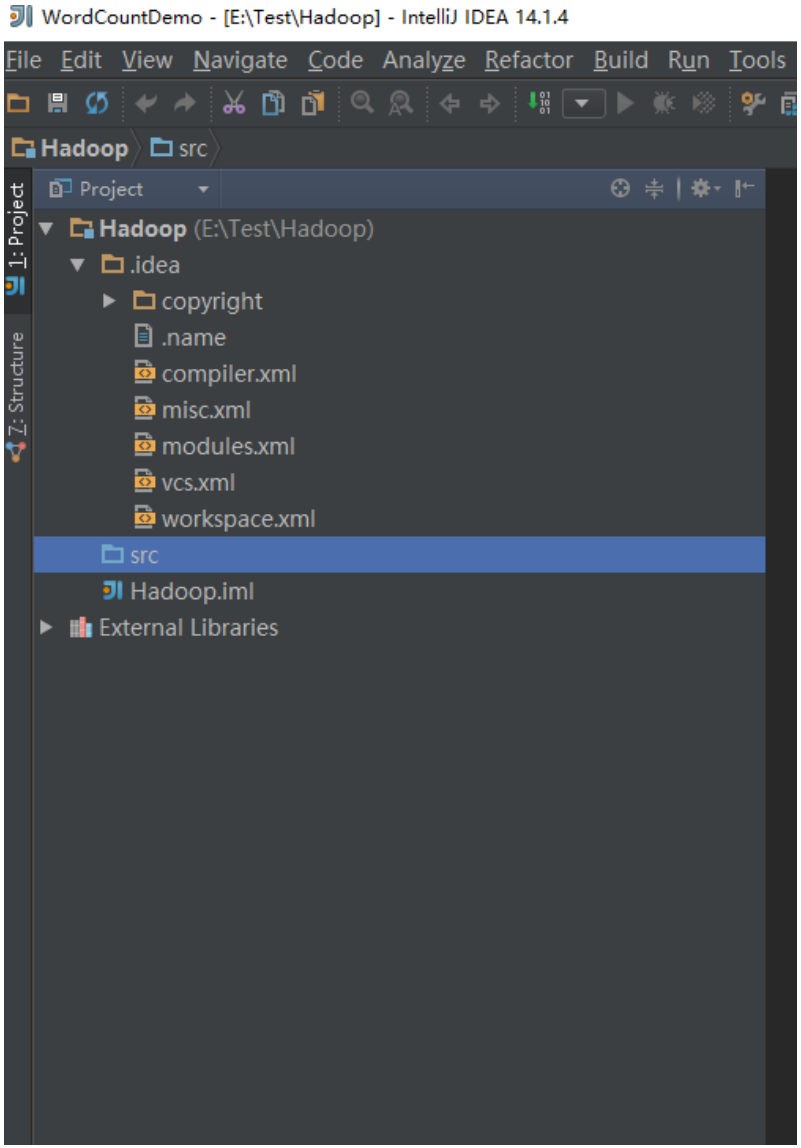
启动IntelliJ IDEA

创建Java Project



创建Java Project



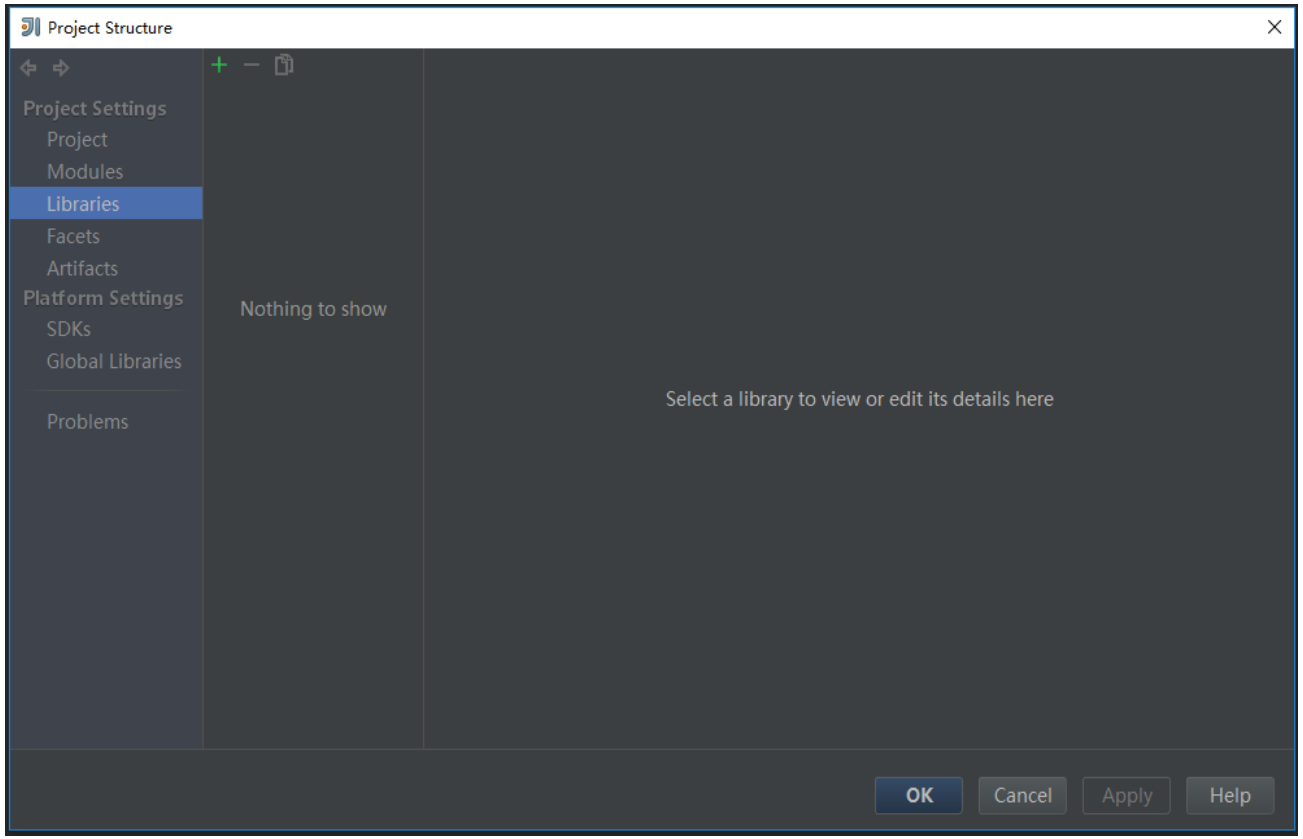


配置Java Project

加入外部jar文件

进入工具栏菜单 File-》Project Stucture

加入hadoop-mapreduce-client-core-2.7.1.jar
以及lib下所有的jar,
在${HADOOP_HOME}/share/hadoop/mapreduce目录下

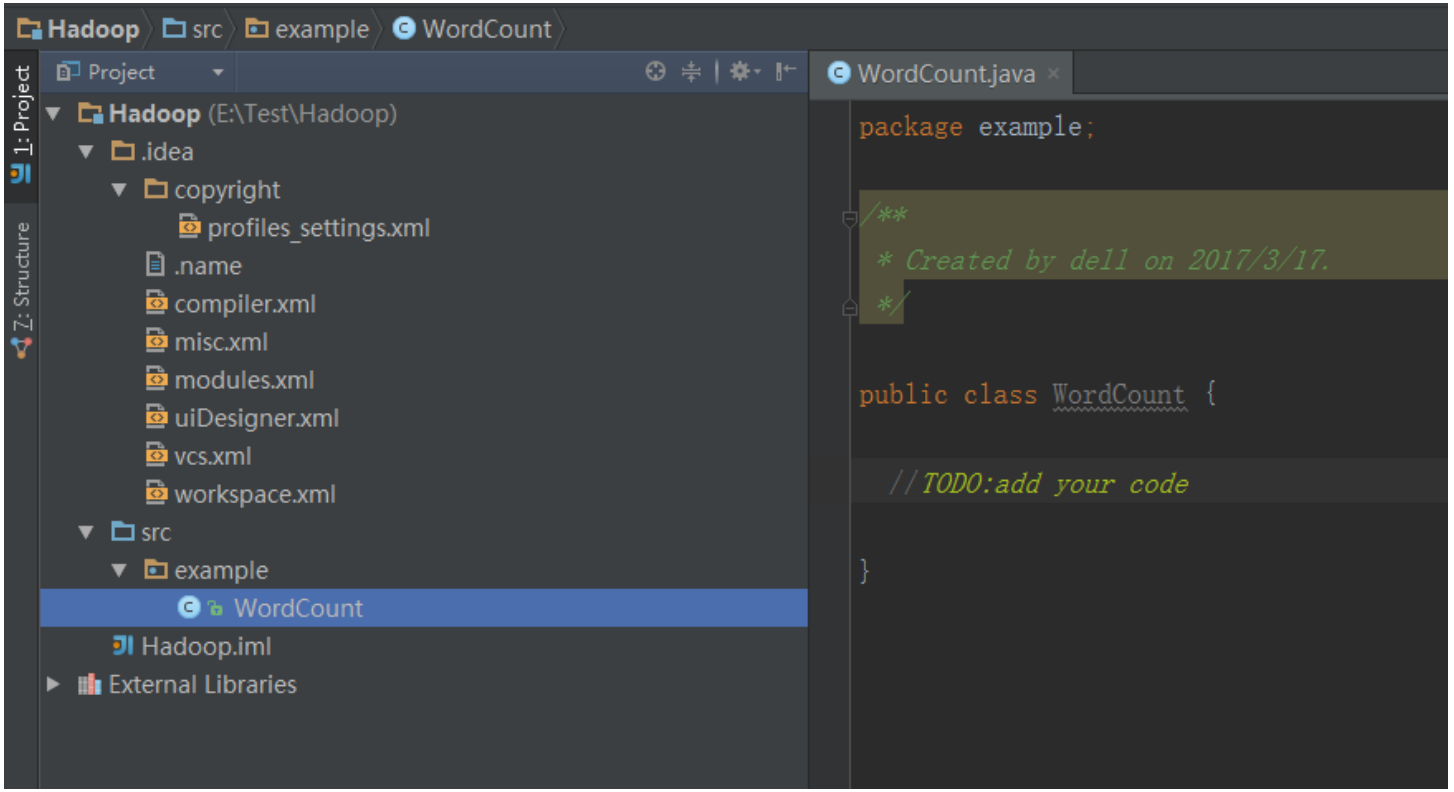


编写程序代码

编写程序类代码

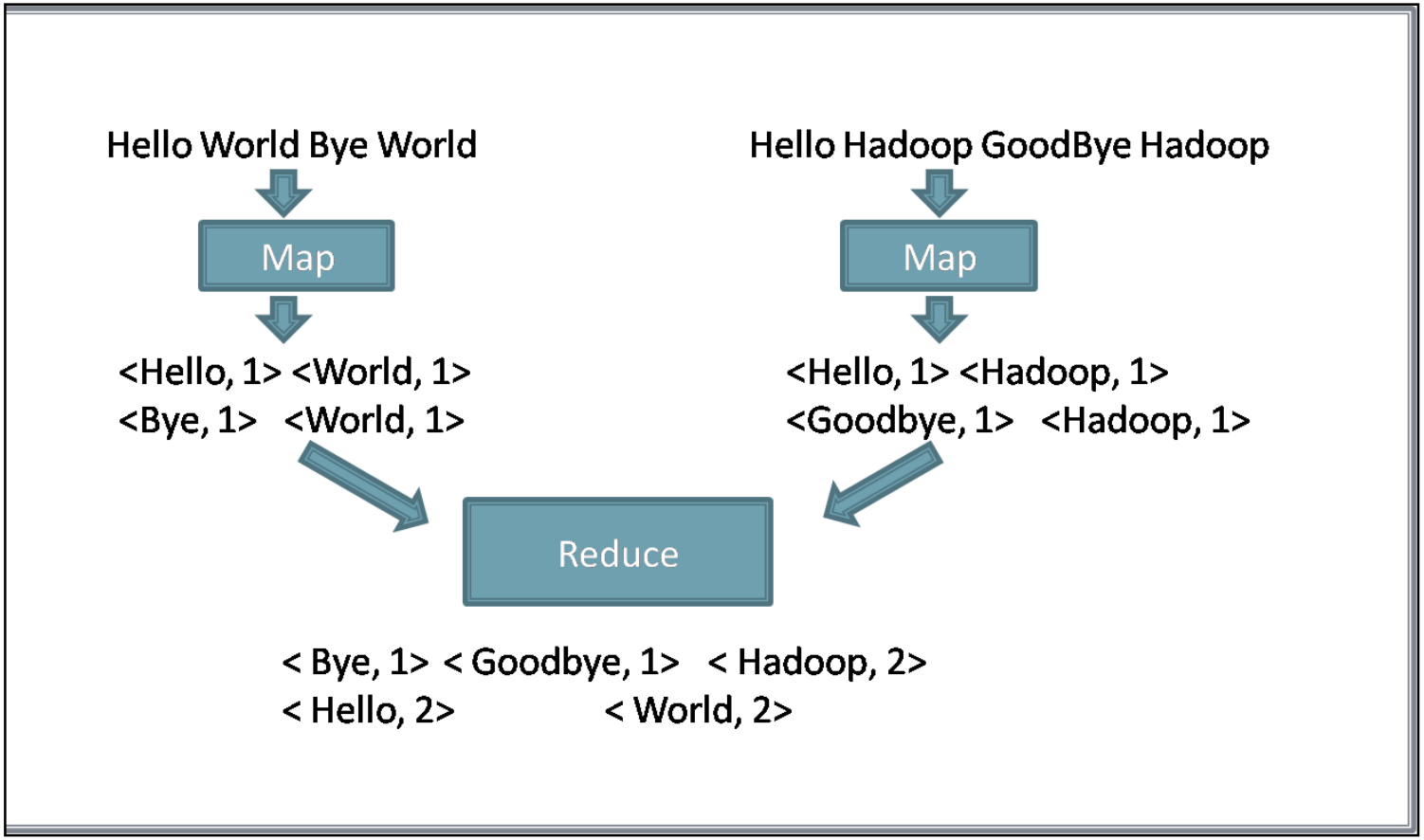
生成名为example的package

生成example.WordCount.java类



WordCount编程示例

基本数据计算流程



WordCount编程示例

编程实现

程序员主要的编码工作如下：

- 实现Map类
- 实现Reduce类
- 实现main函数（运行Job）

WordCount编程示例

实现Map类

- 这个类实现 Mapper 接口中的 map 方法，输入参数中的 value 是文本文件中的一行，利用 StringTokenizer 将这个字符串拆成单词，然后通过**context.write**收集<key, value>对。
- 代码中 LongWritable, IntWritable, Text 均是 Hadoop 中实现的用于封装 Java 数据类型的类，这些类都能够被串行化从而便于在分布式环境中进行数据交换，可以将它们分别视为 long, int, String 的替代。

WordCount编程示例

Map类代码

```
public static class TokenizerMapper  //定义Map类实现字符串分解
        extends Mapper<Object, Text, Text, IntWritable>
{
   private final static IntWritable one = new IntWritable(1);
   private Text word = new Text();
   //实现map()函数
   public void map(Object key, Text value, Context context)
            throws IOException, InterruptedException
   { //将字符串拆解成单词
      StringTokenizer itr = new StringTokenizer(value.toString());
      while (itr.hasMoreTokens())
      {   word.set(itr.nextToken());    //将分解后的一个单词写入word类
          context.write(word, one);     //收集<key, value>
      }
   }
}
```

详细WordCount程序代码

实现Reduce类

- 这个类实现 Reducer 接口中的 reduce 方法, 输入参数中的 (key, values) 是由 Map 任务输出的中间结果，values 是一个 Iterator, 遍历这个 Iterator, 就可以得到属于同一个 key 的所有 value. 此处，key 是一个单词，value 是词频。只需要将所有的 value 相加，就可以得到这个单词的总的出现次数。

详细WordCount程序代码

Reduce类代码

```
//定义Reduce类规约同一key的value
public static class IntSumReducer  extends Reducer<Text,IntWritable,Text,IntWritable>
{
   private IntWritable result = new IntWritable();
   //实现reduce()函数
   public void reduce(Text key, Iterable<IntWritable> values, Context context )
                    throws IOException, InterruptedException
   {
     int sum = 0;
     //遍历迭代values，得到同一key的所有value
     for (IntWritable val : values) {  sum += val.get();      }
     result.set(sum);
     //产生输出对<key, value>
     context.write(key, result);
   }
}
```

详细WordCount程序代码

实现main函数（运行Job）

- 在 Hadoop 中一次计算任务称之为一个 job, main函数主要负责新建一个Job对象并为之设定相应的Mapper和Reducer类，以及输入、输出路径等。

详细WordCount程序代码

main函数代码

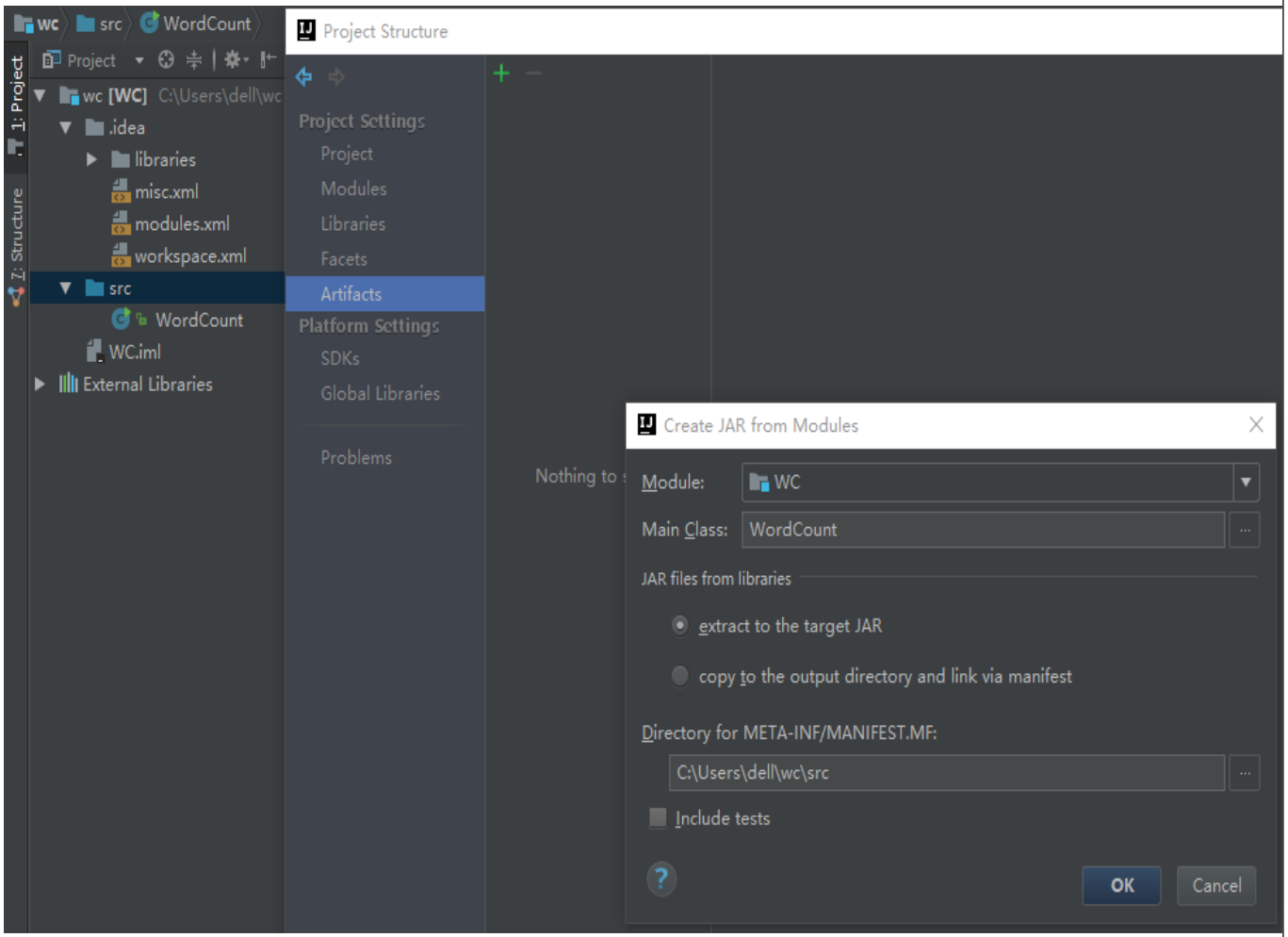
```
public static void main(String[] args) throws Exception
{   //为任务设定配置文件
    Configuration conf = new Configuration();
    //命令行参数
   String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
    if (otherArgs.length != 2)
    {    System.err.println("Usage: wordcount <in> <out>");
         System.exit(2);
    }
    Job job = Job.getInstance(conf, "word count");//新建一个用户定义的Job
    job.setJarByClass(WordCount.class);              //设置执行任务的jar
    job.setMapperClass(TokenizerMapper.class);   //设置Mapper类
    job.setCombinerClass(IntSumReducer.class);  //设置Combine类
    job.setReducerClass(IntSumReducer.class);    //设置Reducer类
    job.setOutputKeyClass(Text.class);               //设置job输出的key
    //设置job输出的value
    job.setOutputValueClass(IntWritable.class);
    //设置输入文件的路径
    FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
    //设置输出文件的路径
    FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
    //提交任务并等待任务完成
    System.exit(job.waitForCompletion(true) ? 0 : 1);
}
```

编译源代码

完成编译

导出jar文件

导出WordCount
程序的jar打包文件



本地运行调试

- 导出jar文件的时候可以指定一个主类MainClass，作为默认执行的一个类
- 将程序复制到本地Hadoop系统的执行目录，并准备一个小的测试数据，即可通过hadoop的安装包进行运行调试

```
bin/hdfs dfs –mkdir input
bin/hdfs dfs –put docs/*.html input
bin/hadoop jar example.jar wordcount input output
```

远程作业提交

- 当需要用集群进行海量数据处理时，在本地程序调试正确运行后，可按照前述的远程作业提交步骤，将作业提交到远程hadoop集群上运行。

实验2：安装单机Hadoop系统与WordCount程序实验

实验内容与要求

1. 每人在自己本地电脑上正确安装和运行伪分布式Hadoop系统

   安装操作手册和本课程课件请从MapReduce课程目录下载。

2. 安装完成后,自己寻找一组英文网页数据,在本机上运行Hadoop系统自带的WordCount可执行程序文件,并产生输出结果

3. 实验结果提交：要求书写一个实验报告，其中包括：

   1. 系统安装运行的情况
   2. 实验数据说明（下载的什么网页数据，多少个HTML或text文件)
   3. 程序运行后在Hadoop Web作业状态查看界面上的作业运行状态屏幕拷贝
   4. 实验输出结果开头部分的屏幕拷贝
   5. 实验体会
   6. 实验报告文件命名规则：MPLab1-学号-姓名.doc

      实验完成时间：4月11日前完成并提交报告

后续实验安排

实验分组

- 自第三次实验开始，以小组形式完成后续实验，以及最后的课程设计。
- 可自由组合实验小组，每组2-4人
- 请各小组成立后，为在Hadoop集群上创建小组的访问账户，请将小组成员信息（姓名，学号，邮箱）发送给课程邮箱

Thanks !