在Max OSX上安装Hadoop

这学期刚好选修了大数据课程，需要使用Hadoop。由于自己使用的电脑刚好是Mac OSX,这里给出如何在Mac OSX上面安装和使用Hadoop

**安装homebrew**

习惯使用Ubuntu的同学，一定很喜欢Ubuntu系统apt-get的软件安装方式。Mac上也有类似这样的包管理器，利用Homebrew即可。  
[Homebrew的官方网站](http://brew.sh/)  
安装Homebrew的方法：

|  |
| --- |
| /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)" |

需要注意的是：  
Homebrew安装的软件都集中在/usr/local/Cellar里面

**ssh登录本地**

|  |
| --- |
| ssh-keygen -t rsa -P "" cat $HOME/.ssh/id\_rsa.pub >> $HOME/.ssh/authorized\_keys |

这样就可以生成ssh公钥，接下来进行测试登录本地是否成功

|  |
| --- |
| ssh localhost |

登录成功显示结果如下：

|  |
| --- |
| Last login: Mon Feb 29 18:29:55 2016 from ::1 |

**安装Hadoop**

输入以下代码，自动安装hadoop

|  |
| --- |
| brew install hadoop |

安装过程会提示重要的信息，如下：

|  |
| --- |
| $JAVA\_HOME has been set to be the output of:  /usr/libexec/java\_home |

在Mac中，我们可以终端输入：/usr/libexec/java\_home来获取JAVA\_HOME的路径  
Hadoop的安装需要配置JAVA\_HOME，用 brew安装，就已经帮我们配置好了。

**测试Hadoop是否安装成功**

Hadoop有三种安装模式：单机模式，伪分布式模式，分布式模式  
分布式模式需要在多台电脑上面测试，这里只测试前面两种，即单机模式和伪分布式模式

**测试单机模式**

这里使用Hadoop附带的示例来检验单机模式是否运行正常。  
通过Homebrew安装的Hadoop，附带的示例在路径/usr/local/Cellar/hadoop/2.7.1/libexec/share/hadoop/mapreduce  
单词计数wordcount是最简单也是最能体现MapReduce思想的程序之一，可以称为MapReduce版”Hello World”，单词计数主要完成功能是：统计一系列文本文件中每个单词出现的次数.

* 创建input目录和output目录  
  input作为输入目录，output目录作为输出目录

|  |
| --- |
| cd /usr/local/Cellar/hadoop/2.7.1/ mkdir input  mkdir output |

* 在input文件夹中创建两个测试文件file1.txt和file2.txt

|  |
| --- |
| cd input echo 'hello world' > file1.txt echo 'hello hadoop' > file2.txt |

* 运行示例检测单机模式

|  |
| --- |
| hadoop jar ./libexec/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar wordcount ./input ./output |

* 查看结果

|  |
| --- |
| more output/part-r-00000 |

显示结果如下：

* hadoop 1
* hello 2
* world 1

**测试伪分布式模式**

测试为分布模式前，需要修改相关的配置文件，把之前的单机模式修改成伪分布式模式

* 修改hadoop-env.sh  
  文件地址：/usr/local/Cellar/hadoop/2.7.1/libexec/etc/hadoop/hadoop-env.sh  
  注释原来的HADOOP\_OPOTS

|  |
| --- |
| #export HADOOP\_OPTS="$HADOOP\_OPTS -Djava.net.preferIPv4Stack=true" |

* 修改为

|  |
| --- |
| export HADOOP\_OPTS="$HADOOP\_OPTS -Djava.net.preferIPv4Stack=true -Djava.security.krb5.realm= -Djava.security.krb5.kdc=" |

* 修改Core-site.xml  
  文件地址：/usr/local/Cellar/hadoop/2.7.1/libexec/etc/hadoop/core-site.xml  
  把原来的

|  |
| --- |
| <configuration> </configuration> |

* 修改为

|  |
| --- |
| <configuration>  <property>  <name>hadoop.tmp.dir</name>  <value>/usr/local/Cellar/hadoop/hdfs/tmp</value>  <description>A base for other temporary directories</description>  </property>  <property>  <name>fs.default.name</name>  <value>hdfs://localhost:9000</value>  </property> </configuration> |

* fs.default.name 保存了NameNode的位置，HDFS和MapReduce组件都需要用到它，这就是它出现在core-site.xml 文件中而不是 hdfs-site.xml文件中的原因
* 修改mapred-site.xml.template  
  文件地址：/usr/local/Cellar/hadoop/2.7.1/libexec/etc/hadoop/mapred-site.xml.template  
  把原来的

|  |
| --- |
| <configuration> </configuration> |

* 修改为

|  |
| --- |
| <configuration>  <property>  <name>mapred.job.tracker</name>  <value>localhost:9010</value>  </property> </configuration> |

* 变量mapred.job.tracker 保存了JobTracker的位置，因为只有MapReduce组件需要知道这个位置，所以它出现在mapred-site.xml文件中。
* 修改hdfs-site.xml  
  文件地址：/usr/local/Cellar/hadoop/2.7.1/libexec/etc/hadoop/hdfs-site.xml  
  把原来的

|  |
| --- |
| <configuration> </configuration> |

* 修改为

|  |
| --- |
| <configuration>  <property>  <name>dfs.replication</name>  <value>1</value>  </property> </configuration> |

* 变量dfs.replication指定了每个HDFS数据库的复制次数。 通常为3, 由于我们只有一台主机和一个伪分布式模式的DataNode，将此值修改为1。
* 运行

|  |
| --- |
| hadoop namenode -format ./sbin/start-all.sh |

* 运行成功后，用浏览器访问：[Hdfs的Web界面](localhost:50070),可以查看相关的信息
* 用示例测试
  + 估计圆周率PI的值

|  |
| --- |
| hadoop jar ./libexec/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar pi 2 5 |

* + 运行结果如下：

|  |
| --- |
| Estimated value of Pi is 3.60000000000000000000 |

* + wordcount统计数据  
    把原来用于单机模式的测试input文件上传到hdfs中

|  |
| --- |
| hadoop fs -put ./input input |

* + 运行wordcount

|  |
| --- |
| hadoop jar ./libexec/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar wordcount input output |

* + 查看output的结果文件

|  |
| --- |
| hadoop fs -tail output/part-r-00000 |

* + 得到的结果是：
    - hadoop 1
    - hello 2
    - world 1
* 关闭伪分布式

|  |
| --- |
| ./sbin/stop-all.sh |

以下程序在hadoop1.2.1上测试成功。

本例先将源代码呈现，然后详细说明执行步骤，最后对源代码及执行过程进行分析。

一、源代码

**[java]** [view plain](http://blog.csdn.net/jediael_lu/article/details/38705371) [copy](http://blog.csdn.net/jediael_lu/article/details/38705371)

[CODE上查看代码片](https://code.csdn.net/snippets/454639)

1. **package** org.jediael.hadoopdemo.wordcount;
3. **import** java.io.IOException;
4. **import** java.util.StringTokenizer;
6. **import** org.apache.hadoop.conf.Configuration;
7. **import** org.apache.hadoop.fs.Path;
8. **import** org.apache.hadoop.io.IntWritable;
9. **import** org.apache.hadoop.io.LongWritable;
10. **import** org.apache.hadoop.io.Text;
11. **import** org.apache.hadoop.mapreduce.Job;
12. **import** org.apache.hadoop.mapreduce.Mapper;
13. **import** org.apache.hadoop.mapreduce.Reducer;
14. **import** org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
15. **import** org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
16. **import** org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
17. **import** org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
19. **public** **class** WordCount {
21. **public** **static** **class** WordCountMap **extends**
22. Mapper<LongWritable, Text, Text, IntWritable> {
24. **private** **final** IntWritable one = **new** IntWritable(1);
25. **private** Text word = **new** Text();
27. **public** **void** map(LongWritable key, Text value, Context context)
28. **throws** IOException, InterruptedException {
29. String line = value.toString();
30. StringTokenizer token = **new** StringTokenizer(line);
31. **while** (token.hasMoreTokens()) {
32. word.set(token.nextToken());
33. context.write(word, one);
34. }
35. }
36. }
38. **public** **static** **class** WordCountReduce **extends**
39. Reducer<Text, IntWritable, Text, IntWritable> {
41. **public** **void** reduce(Text key, Iterable<IntWritable> values,
42. Context context) **throws** IOException, InterruptedException {
43. **int** sum = 0;
44. **for** (IntWritable val : values) {
45. sum += val.get();
46. }
47. context.write(key, **new** IntWritable(sum));
48. }
49. }
51. **public** **static** **void** main(String[] args) **throws** Exception {
52. Configuration conf = **new** Configuration();
53. Job job = **new** Job(conf);
54. job.setJarByClass(WordCount.**class**);
55. job.setJobName("wordcount");
57. job.setOutputKeyClass(Text.**class**);
58. job.setOutputValueClass(IntWritable.**class**);
60. job.setMapperClass(WordCountMap.**class**);
61. job.setReducerClass(WordCountReduce.**class**);
63. job.setInputFormatClass(TextInputFormat.**class**);
64. job.setOutputFormatClass(TextOutputFormat.**class**);
66. FileInputFormat.addInputPath(job, **new** Path(args[0]));
67. FileOutputFormat.setOutputPath(job, **new** Path(args[1]));
69. job.waitForCompletion(**true**);
70. }
71. }

二、执行程序

1、从eclipse从导出至wordcount.jar，并上传至[**Hadoop**](http://lib.csdn.net/base/20)服务器，本例中，将程序上传至/home/jediael/project。

2、安装hadoop伪分布模式，可参考[Hadoop1.2.1伪分布模式安装指南](http://blog.csdn.net/jediael_lu/article/details/38637277)，本实例将运行在hadoop的伪公布环境中。

3、在HDFS中创建目录wcinput，用作输入目录，并将需要分析的文件复制到目录下。

**[plain]** [view plain](http://blog.csdn.net/jediael_lu/article/details/38705371) [copy](http://blog.csdn.net/jediael_lu/article/details/38705371)

[CODE上查看代码片](https://code.csdn.net/snippets/454639)

1. [root@jediael conf]# hadoop fs -mkdir wcinput
2. [root@jediael conf]# hadoop fs -copyFromLocal \* wcinput
3. [root@jediael conf]# hadoop fs -ls wcinput
4. Found 26 items
5. -rw-r--r-- 1 root supergroup 1524 2014-08-20 12:29 /user/root/wcinput/automaton-urlfilter.txt
6. -rw-r--r-- 1 root supergroup 1311 2014-08-20 12:29 /user/root/wcinput/configuration.xsl
7. -rw-r--r-- 1 root supergroup 131090 2014-08-20 12:29 /user/root/wcinput/domain-suffixes.xml
8. -rw-r--r-- 1 root supergroup 4649 2014-08-20 12:29 /user/root/wcinput/domain-suffixes.xsd
9. -rw-r--r-- 1 root supergroup 824 2014-08-20 12:29 /user/root/wcinput/domain-urlfilter.txt
10. -rw-r--r-- 1 root supergroup 3368 2014-08-20 12:29 /user/root/wcinput/gora-accumulo-mapping.xml
11. -rw-r--r-- 1 root supergroup 3279 2014-08-20 12:29 /user/root/wcinput/gora-cassandra-mapping.xml
12. -rw-r--r-- 1 root supergroup 3447 2014-08-20 12:29 /user/root/wcinput/gora-hbase-mapping.xml
13. -rw-r--r-- 1 root supergroup 2677 2014-08-20 12:29 /user/root/wcinput/gora-sql-mapping.xml
14. -rw-r--r-- 1 root supergroup 2993 2014-08-20 12:29 /user/root/wcinput/gora.properties
15. -rw-r--r-- 1 root supergroup 983 2014-08-20 12:29 /user/root/wcinput/hbase-site.xml
16. -rw-r--r-- 1 root supergroup 3096 2014-08-20 12:29 /user/root/wcinput/httpclient-auth.xml
17. -rw-r--r-- 1 root supergroup 3948 2014-08-20 12:29 /user/root/wcinput/log4j.properties
18. -rw-r--r-- 1 root supergroup 511 2014-08-20 12:29 /user/root/wcinput/nutch-conf.xsl
19. -rw-r--r-- 1 root supergroup 42610 2014-08-20 12:29 /user/root/wcinput/nutch-default.xml
20. -rw-r--r-- 1 root supergroup 753 2014-08-20 12:29 /user/root/wcinput/nutch-site.xml
21. -rw-r--r-- 1 root supergroup 347 2014-08-20 12:29 /user/root/wcinput/parse-plugins.dtd
22. -rw-r--r-- 1 root supergroup 3016 2014-08-20 12:29 /user/root/wcinput/parse-plugins.xml
23. -rw-r--r-- 1 root supergroup 857 2014-08-20 12:29 /user/root/wcinput/prefix-urlfilter.txt
24. -rw-r--r-- 1 root supergroup 2484 2014-08-20 12:29 /user/root/wcinput/regex-normalize.xml
25. -rw-r--r-- 1 root supergroup 1736 2014-08-20 12:29 /user/root/wcinput/regex-urlfilter.txt
26. -rw-r--r-- 1 root supergroup 18969 2014-08-20 12:29 /user/root/wcinput/schema-solr4.xml
27. -rw-r--r-- 1 root supergroup 6020 2014-08-20 12:29 /user/root/wcinput/schema.xml
28. -rw-r--r-- 1 root supergroup 1766 2014-08-20 12:29 /user/root/wcinput/solrindex-mapping.xml
29. -rw-r--r-- 1 root supergroup 1044 2014-08-20 12:29 /user/root/wcinput/subcollections.xml
30. -rw-r--r-- 1 root supergroup 1411 2014-08-20 12:29 /user/root/wcinput/suffix-urlfilter.txt

4、运行程序

**[plain]** [view plain](http://blog.csdn.net/jediael_lu/article/details/38705371) [copy](http://blog.csdn.net/jediael_lu/article/details/38705371)

[CODE上查看代码片](https://code.csdn.net/snippets/454639)

1. [root@jediael project]# hadoop org.jediael.hadoopdemo.wordcount.WordCount wcinput wcoutput3
2. 14/08/20 12:50:25 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. Applications should implement Tool for the same.
3. 14/08/20 12:50:26 INFO input.FileInputFormat: Total input paths to process : 26
4. 14/08/20 12:50:26 INFO util.NativeCodeLoader: Loaded the native-hadoop library
5. 14/08/20 12:50:26 WARN snappy.LoadSnappy: Snappy native library not loaded
6. 14/08/20 12:50:26 INFO mapred.JobClient: Running job: job\_201408191134\_0005
7. 14/08/20 12:50:27 INFO mapred.JobClient: map 0% reduce 0%
8. 14/08/20 12:50:38 INFO mapred.JobClient: map 3% reduce 0%
9. 14/08/20 12:50:39 INFO mapred.JobClient: map 7% reduce 0%
10. 14/08/20 12:50:50 INFO mapred.JobClient: map 15% reduce 0%
11. 14/08/20 12:50:57 INFO mapred.JobClient: map 19% reduce 0%
12. 14/08/20 12:50:58 INFO mapred.JobClient: map 23% reduce 0%
13. 14/08/20 12:51:00 INFO mapred.JobClient: map 23% reduce 5%
14. 14/08/20 12:51:04 INFO mapred.JobClient: map 30% reduce 5%
15. 14/08/20 12:51:06 INFO mapred.JobClient: map 30% reduce 10%
16. 14/08/20 12:51:11 INFO mapred.JobClient: map 38% reduce 10%
17. 14/08/20 12:51:16 INFO mapred.JobClient: map 38% reduce 11%
18. 14/08/20 12:51:18 INFO mapred.JobClient: map 46% reduce 11%
19. 14/08/20 12:51:19 INFO mapred.JobClient: map 46% reduce 12%
20. 14/08/20 12:51:22 INFO mapred.JobClient: map 46% reduce 15%
21. 14/08/20 12:51:25 INFO mapred.JobClient: map 53% reduce 15%
22. 14/08/20 12:51:31 INFO mapred.JobClient: map 53% reduce 17%
23. 14/08/20 12:51:32 INFO mapred.JobClient: map 61% reduce 17%
24. 14/08/20 12:51:39 INFO mapred.JobClient: map 69% reduce 17%
25. 14/08/20 12:51:40 INFO mapred.JobClient: map 69% reduce 20%
26. 14/08/20 12:51:45 INFO mapred.JobClient: map 73% reduce 20%
27. 14/08/20 12:51:46 INFO mapred.JobClient: map 76% reduce 23%
28. 14/08/20 12:51:52 INFO mapred.JobClient: map 80% reduce 23%
29. 14/08/20 12:51:53 INFO mapred.JobClient: map 84% reduce 23%
30. 14/08/20 12:51:55 INFO mapred.JobClient: map 84% reduce 25%
31. 14/08/20 12:51:59 INFO mapred.JobClient: map 88% reduce 25%
32. 14/08/20 12:52:00 INFO mapred.JobClient: map 92% reduce 25%
33. 14/08/20 12:52:02 INFO mapred.JobClient: map 92% reduce 29%
34. 14/08/20 12:52:06 INFO mapred.JobClient: map 96% reduce 29%
35. 14/08/20 12:52:07 INFO mapred.JobClient: map 100% reduce 29%
36. 14/08/20 12:52:11 INFO mapred.JobClient: map 100% reduce 30%
37. 14/08/20 12:52:15 INFO mapred.JobClient: map 100% reduce 100%
38. 14/08/20 12:52:17 INFO mapred.JobClient: Job complete: job\_201408191134\_0005
39. 14/08/20 12:52:18 INFO mapred.JobClient: Counters: 29
40. 14/08/20 12:52:18 INFO mapred.JobClient: Job Counters
41. 14/08/20 12:52:18 INFO mapred.JobClient: Launched reduce tasks=1
42. 14/08/20 12:52:18 INFO mapred.JobClient: SLOTS\_MILLIS\_MAPS=192038
43. 14/08/20 12:52:18 INFO mapred.JobClient: Total time spent by all reduces waiting after reserving slots (ms)=0
44. 14/08/20 12:52:18 INFO mapred.JobClient: Total time spent by all maps waiting after reserving slots (ms)=0
45. 14/08/20 12:52:18 INFO mapred.JobClient: Launched map tasks=26
46. 14/08/20 12:52:18 INFO mapred.JobClient: Data-local map tasks=26
47. 14/08/20 12:52:18 INFO mapred.JobClient: SLOTS\_MILLIS\_REDUCES=95814
48. 14/08/20 12:52:18 INFO mapred.JobClient: File Output Format Counters
49. 14/08/20 12:52:18 INFO mapred.JobClient: Bytes Written=123950
50. 14/08/20 12:52:18 INFO mapred.JobClient: FileSystemCounters
51. 14/08/20 12:52:18 INFO mapred.JobClient: FILE\_BYTES\_READ=352500
52. 14/08/20 12:52:18 INFO mapred.JobClient: HDFS\_BYTES\_READ=247920
53. 14/08/20 12:52:18 INFO mapred.JobClient: FILE\_BYTES\_WRITTEN=2177502
54. 14/08/20 12:52:18 INFO mapred.JobClient: HDFS\_BYTES\_WRITTEN=123950
55. 14/08/20 12:52:18 INFO mapred.JobClient: File Input Format Counters
56. 14/08/20 12:52:18 INFO mapred.JobClient: Bytes Read=244713
57. 14/08/20 12:52:18 INFO mapred.JobClient: Map-Reduce Framework
58. 14/08/20 12:52:18 INFO mapred.JobClient: Map output materialized bytes=352650
59. 14/08/20 12:52:18 INFO mapred.JobClient: Map input records=7403
60. 14/08/20 12:52:18 INFO mapred.JobClient: Reduce shuffle bytes=352650
61. 14/08/20 12:52:18 INFO mapred.JobClient: Spilled Records=45210
62. 14/08/20 12:52:18 INFO mapred.JobClient: Map output bytes=307281
63. 14/08/20 12:52:18 INFO mapred.JobClient: Total committed heap usage (bytes)=3398606848
64. 14/08/20 12:52:18 INFO mapred.JobClient: CPU time spent (ms)=14400
65. 14/08/20 12:52:18 INFO mapred.JobClient: Combine input records=0
66. 14/08/20 12:52:18 INFO mapred.JobClient: SPLIT\_RAW\_BYTES=3207
67. 14/08/20 12:52:18 INFO mapred.JobClient: Reduce input records=22605
68. 14/08/20 12:52:18 INFO mapred.JobClient: Reduce input groups=6749
69. 14/08/20 12:52:18 INFO mapred.JobClient: Combine output records=0
70. 14/08/20 12:52:18 INFO mapred.JobClient: Physical memory (bytes) snapshot=4799041536
71. 14/08/20 12:52:18 INFO mapred.JobClient: Reduce output records=6749
72. 14/08/20 12:52:18 INFO mapred.JobClient: Virtual memory (bytes) snapshot=19545337856
73. 14/08/20 12:52:18 INFO mapred.JobClient: Map output records=22605

5、查看结果

**[plain]** [view plain](http://blog.csdn.net/jediael_lu/article/details/38705371) [copy](http://blog.csdn.net/jediael_lu/article/details/38705371)

[CODE上查看代码片](https://code.csdn.net/snippets/454639)

1. root@jediael project]# hadoop fs -ls wcoutput3
2. Found 3 items
3. -rw-r--r-- 1 root supergroup 0 2014-08-20 12:52 /user/root/wcoutput3/\_SUCCESS
4. drwxr-xr-x - root supergroup 0 2014-08-20 12:50 /user/root/wcoutput3/\_logs
5. -rw-r--r-- 1 root supergroup 123950 2014-08-20 12:52 /user/root/wcoutput3/part-r-00000
6. [root@jediael project]# hadoop fs -cat wcoutput3/part-r-00000
7. !!      2
8. !ci.\*.\*.us      1
9. !co.\*.\*.us      1
10. !town.\*.\*.us    1
11. "AS     22
12. "Accept"        1
13. "Accept-Language"       1
14. "License");     22
15. "NOW"   1
16. "WiFi"  1
17. "Z"     1
18. "all"   1
19. "content"       1
20. "delete 1
21. "delimiter"     1

………………  
  
三、程序分析

1、WordCountMap类继承了org.apache.hadoop.mapreduce.Mapper，4个泛型类型分别是map函数输入key的类型，输入value的类型，输出key的类型，输出value的类型。

2、WordCountReduce类继承了org.apache.hadoop.mapreduce.Reducer，4个泛型类型含义与map类相同。

3、map的输出类型与reduce的输入类型相同，而一般情况下，map的输出类型与reduce的输出类型相同，因此，reduce的输入类型与输出类型相同。

4、hadoop根据以下代码确定输入内容的格式：

job.setInputFormatClass(TextInputFormat.class);

TextInputFormat是hadoop默认的输入方法，它继承自FileInputFormat。在TextInputFormat中，它将数据集切割成小数据集InputSplit，每一个InputSplit由一个mapper处理。此外，InputFormat还提供了一个RecordReader的实现，将一个InputSplit解析成<key,value>的形式，并提供给map函数：

key：这个数据相对于数据分片中的字节偏移量，数据类型是LongWritable。

value：每行数据的内容，类型是Text。

因此，在本例中，map函数的key/value类型是LongWritable与Text。

5、Hadoop根据以下代码确定输出内容的格式：

job.setOutputFormatClass(TextOutputFormat.class);

TextOutputFormat是hadoop默认的输出格式，它会将每条记录一行的形式存入文本文件，如

the 30

happy 23

……