大规模分布式系统第五次实验报告
员司雨 17307110448

用 Spark 实现两个矩阵的乘法运算

一. 实验介绍

输入文件的每一行为<矩阵名 行号 列号 值>例如某行为 A 1 2 4，则表示矩阵 A 第一行第二列的值为 4。输出的结果每一行为<行号 列号 值>。

二. 实验环境

1. Ubuntu18.04
2. jdk 1.8.0_131
3. hadoop 2.7.7
4. Python 3.6.2
5. spark 3.0.0
6. scala 2.11.8

三. 实验过程

PartI: scala 安装与配置

1. 下载并安装

```
ics@ubuntu:/tmp/mozilla_ics0$ sudo tar zxvf scala-2.11.8.tgz -C /usr/local/
scala-2.11.8/
scala-2.11.8/man/
scala-2.11.8/man/man1/
scala-2.11.8/man/man1/scala.1
scala-2.11.8/man/man1/scalap.1
scala-2.11.8/man/man1/fsc.1
scala-2.11.8/man/man1/scaladoc.1
scala-2.11.8/man/man1/scalac.1
scala-2.11.8/bin/
scala-2.11.8/bin/scalac
scala-2.11.8/bin/fsc
scala-2.11.8/bin/fsc.bat
```

安装 scala 2.11.8

2. 移动文件夹并更改文件名为 scala

```
ics@ubuntu:/usr/local$ ls
bin  games   include  man      sbin          share  src
etc  hadoop  lib      python3  scala-2.11.8  spark
ics@ubuntu:/usr/local$ sudo mv scala-2.11.8 scala
ics@ubuntu:/usr/local$ ls
bin  games   include  man      sbin   share  src
etc  hadoop  lib      python3  scala  spark
ics@ubuntu:/usr/local$
```

安装 scala 2.11.8

3. 配置环境

```
ics@ubuntu:/usr/local$ sudo gedit ~/.bashrc
```

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64 # JDK安装目录
export HADOOP_HOME=/usr/local/hadoop
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"
export JAVA_LIBRARY_PATH=$HADOOP_HOME/lib/native:$JAVA_LIBRARY_PATH
# scala env
export SCALA_HOME=/usr/local/scala
export PATH=$PATH:$SCALA_HOME/bin
```

配置环境

```
ics@ubuntu:/usr/local$ source ~/.bashrc
```

保存环境修改

4. 安装成功结果

```
ics@ubuntu:/usr/local$ scala -version
Scala code runner version 2.11.8 -- Copyright 2002-2016, LAMP/EPFL
```

安装 scala 2.11.8

PartII: spark 安装与配置

1. 下载并安装

```
ics@ubuntu: /tmp/mozilla_ics0
ics@ubuntu:/tmp/mozilla_ics0$ ls
spark-3.0.0-preview2-bin-hadoop2.7.tgz
ics@ubuntu:/tmp/mozilla_ics0$ sudo tar zxvf spark-3.0.0-preview2-bin-hadoop2.7.t
gz -C /usr/local/
spark-3.0.0-preview2-bin-hadoop2.7/
spark-3.0.0-preview2-bin-hadoop2.7/data/
spark-3.0.0-preview2-bin-hadoop2.7/data/streaming/
spark-3.0.0-preview2-bin-hadoop2.7/data/streaming/AFINN-111.txt
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/sample_binary_classification_data.
txt
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/sample_kmeans_data.txt
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/sample_multiclass_classification_d
ata.txt
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/sample_lda_libsvm_data.txt
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/iris_libsvm.txt
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/pagerank_data.txt
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/sample_linear_regression_data.txt
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/pic_data.txt
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/als/
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/als/test.data
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/als/sample_movielens_ratings.txt
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/ridge-data/
spark-3.0.0-preview2-bin-hadoop2.7/data/mllib/ridge-data/lpsa.data
```

安装 spark 3.0.0

2. 移动文件夹并更改文件名为 spark

```
ics@ubuntu:/usr/local$ ls
bin  games   include  man      sbin   spark-3.0.0-preview2-bin-hadoop2.7
etc  hadoop  lib      python3  share  src
ics@ubuntu:/usr/local$ sudo mv spark-3.0.0-preview2-bin-hadoop2.7 spark
ics@ubuntu:/usr/local$ ls
bin  etc  games  hadoop  include  lib  man  python3  sbin  share  spark  src
ics@ubuntu:/usr/local$
```

3. 配置环境

```
ics@ubuntu:/usr/local$ sudo gedit ~/.bashrc
```

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64 # JDK安装目录
export HADOOP_HOME=/usr/local/hadoop
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"
export JAVA_LIBRARY_PATH=$HADOOP_HOME/lib/native:$JAVA_LIBRARY_PATH
# scala env
export SCALA_HOME=/usr/local/scala
export PATH=$PATH:$SCALA_HOME/bin
# spark env
export SPARK_HOME=/usr/local/spark
export PATH=$PATH:$SPARK_HOME/bin
export PATH=$PATH:$SPARK_HOME/sbin
```

配置环境

```
ics@ubuntu:/usr/local$ source ~/.bashrc
```

保存环境修改

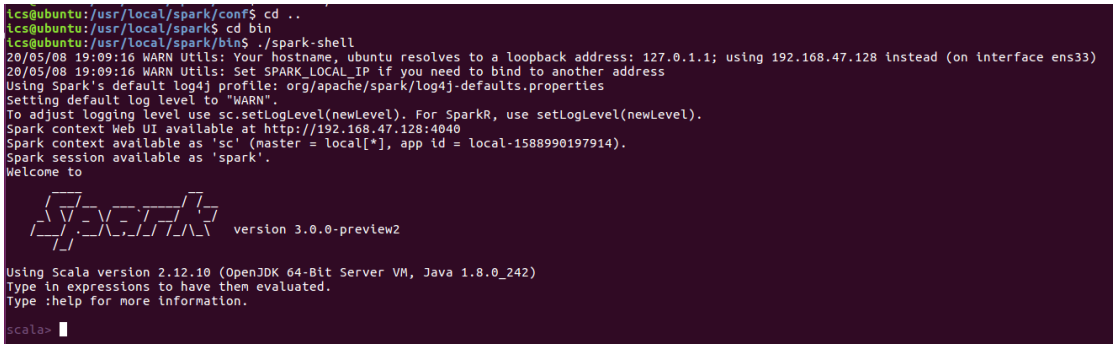
4. 配置 spark-env.sh

```
ics@ubuntu:/usr/local/spark/conf$ sudo gedit spark-env.sh
```

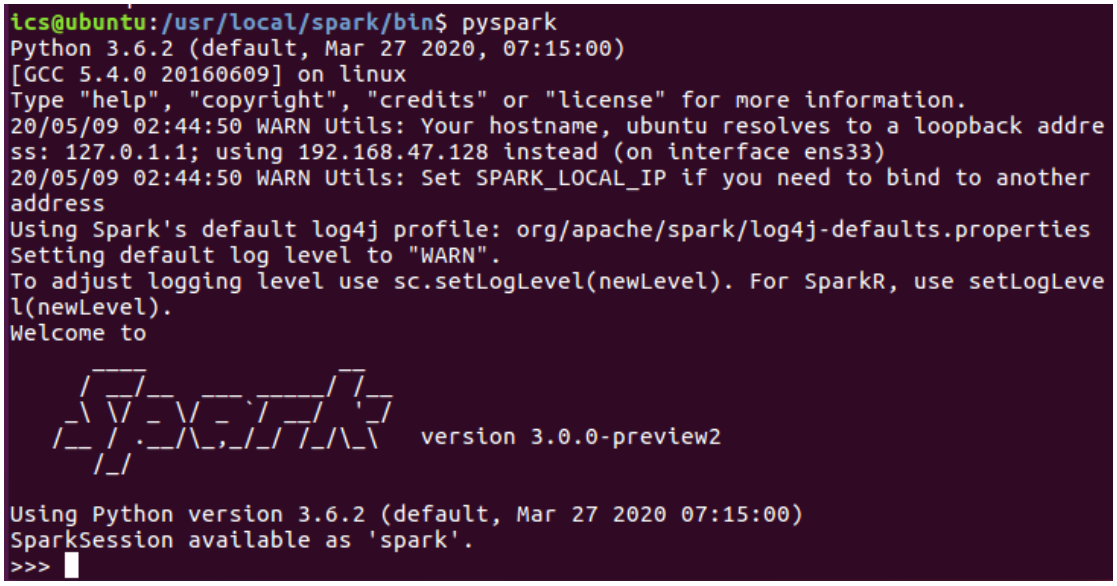
```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64 # JDK安装目录
export HADOOP_HOME=/usr/local/hadoop
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"
export JAVA_LIBRARY_PATH=$HADOOP_HOME/lib/native:$JAVA_LIBRARY_PATH
# scala env
export SCALA_HOME=/usr/local/scala
export PATH=$PATH:$SCALA_HOME/bin
# spark env
export SPARK_HOME=/usr/local/spark
export PATH=$PATH:$SPARK_HOME/bin
export PATH=$PATH:$SPARK_HOME/sbin
export SPARK_MASTER_IP=127.0.0.1
export SPARK_MASTER_PORT=7077
export SPARK_MASTER_WEBUI_PORT=8099
export SPARK_WORKER_CORES=3
export SPARK_WORKER_INSTANCES=1
export SPARK_WORKER_MEMORY=5G
export SPARK_WORKER_WEBUI_PORT=8081
export SPARK_EXECUTOR_CORES=1
export SPARK_EXECUTOR_MEMORY=1G
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:$HADOOP_HOME/lib/native
```

配置并保存

5. 安装成功结果



scala 编译环境



spark 编译环境

PartIII：矩阵乘法函数

1. 代码如下

```
# -*- coding: utf-8 -*-
"""
Created on Sat May  9 19:51:08 2020

@author: yuansiyu
"""

from pyspark import SparkContext

def read_matrix(address):
    f = open(address, encoding='UTF-8')
    line = f.readline()
    A = []
    B = []
```

```
    while line:
        line_ = line.replace('\n','')
        line_ = line_.split(' ')
        if line_[0] == 'A':
            A.append((int(line_[1]), int(line_[2]), int(line_[3])))
        else:
            B.append((int(line_[1]), int(line_[2]), int(line_[3])))
        line = f.readline()
    f.close()
    return A, B

def write_matrix(r):
    f = open('result.txt', 'w', encoding='UTF-8')
    for ele in r:
        f.write('C' + ' ' + str(ele[0][0]) + ' ' + str(ele[0][1]) + ' ' + str(ele[1]) + '\n')
    f.close()

A,B = read_matrix('matrix.txt')

print('read successful')
sc = SparkContext("local")
A_matrix = sc.parallelize(A)
B_matrix = sc.parallelize(B)
temp_A = A_matrix.map(lambda x: (x[1],(x[0],x[2])))
temp_B = B_matrix.map(lambda x: (x[0],(x[1],x[2])))

#temp1:((A(j,(i,v)), B(j,(k,w)))) temp1[0] = A(j,(i,v)) temp1[0][0] = j temp1[0][1][0] = i
temp1= temp_A.cartesian(temp_B).filter(lambda x: x[0][0] == x[1][0])
temp2= temp1.map(lambda x: ((x[0][1][0],x[1][1][0]),x[0][1][1]*x[1][1][1]))
result = temp2.reduceByKey(lambda x, y: x + y)
result = result.sortByKey()
r = result.collect()

write_matrix(r)
show = result.take(4)
print(show)
```

参见：try.py

51. 运行程序

```
ics@ubuntu:/tmp/mozilla_ics0$ spark-submit try.py
20/05/09 05:44:08 WARN Utils: Your hostname, ubuntu resolves to a loopback addre
ss: 127.0.1.1; using 192.168.47.128 instead (on interface ens33)
20/05/09 05:44:08 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another
address
read successful
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
20/05/09 05:44:10 INFO SparkContext: Running Spark version 3.0.0-preview2
20/05/09 05:44:10 INFO ResourceUtils: ==========================================
====================
20/05/09 05:44:10 INFO ResourceUtils: Resources for spark.driver:

20/05/09 05:44:10 INFO ResourceUtils: ==========================================
====================
20/05/09 05:44:10 INFO SparkContext: Submitted application: try.py
20/05/09 05:44:10 INFO SecurityManager: Changing view acls to: ics
20/05/09 05:44:10 INFO SecurityManager: Changing modify acls to: ics
20/05/09 05:44:10 INFO SecurityManager: Changing view acls groups to:
20/05/09 05:44:10 INFO SecurityManager: Changing modify acls groups to:
20/05/09 05:44:10 INFO SecurityManager: SecurityManager: authentication disabled
; ui acls disabled; users  with view permissions: Set(ics); groups with view per
missions: Set(); users  with modify permissions: Set(ics); groups with modify pe
rmissions: Set()
```

运行过程展示

52. 实验结果展示

```
20/05/09 05:44:47 INFO TaskSchedulerImpl: Killing all running tasks in stage 3:
Stage finished
20/05/09 05:44:47 INFO DAGScheduler: Job 1 finished: runJob at PythonRDD.scala:1
54, took 0.269242 s
[((1, 6801), 1330), ((1, 9430), 76), ((3, 6178), 2890), ((4, 7542), 1482)]
20/05/09 05:44:47 INFO SparkContext: Invoking stop() from shutdown hook
20/05/09 05:44:47 INFO SparkUI: Stopped Spark web UI at http://192.168.47.128:40
40
20/05/09 05:44:47 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEnd
point stopped!
20/05/09 05:44:47 INFO MemoryStore: MemoryStore cleared
```

部分结果

```
C 1 6801 1330
C 1 9430 76
C 3 6178 2890
C 4 7542 1482
C 5 9022 1540
C 5 9684 4130
C 6 160 1679
C 7 2062 285
C 7 4337 1408
C 8 5108 1500
C 14 9740 6786
```

输出文件展示