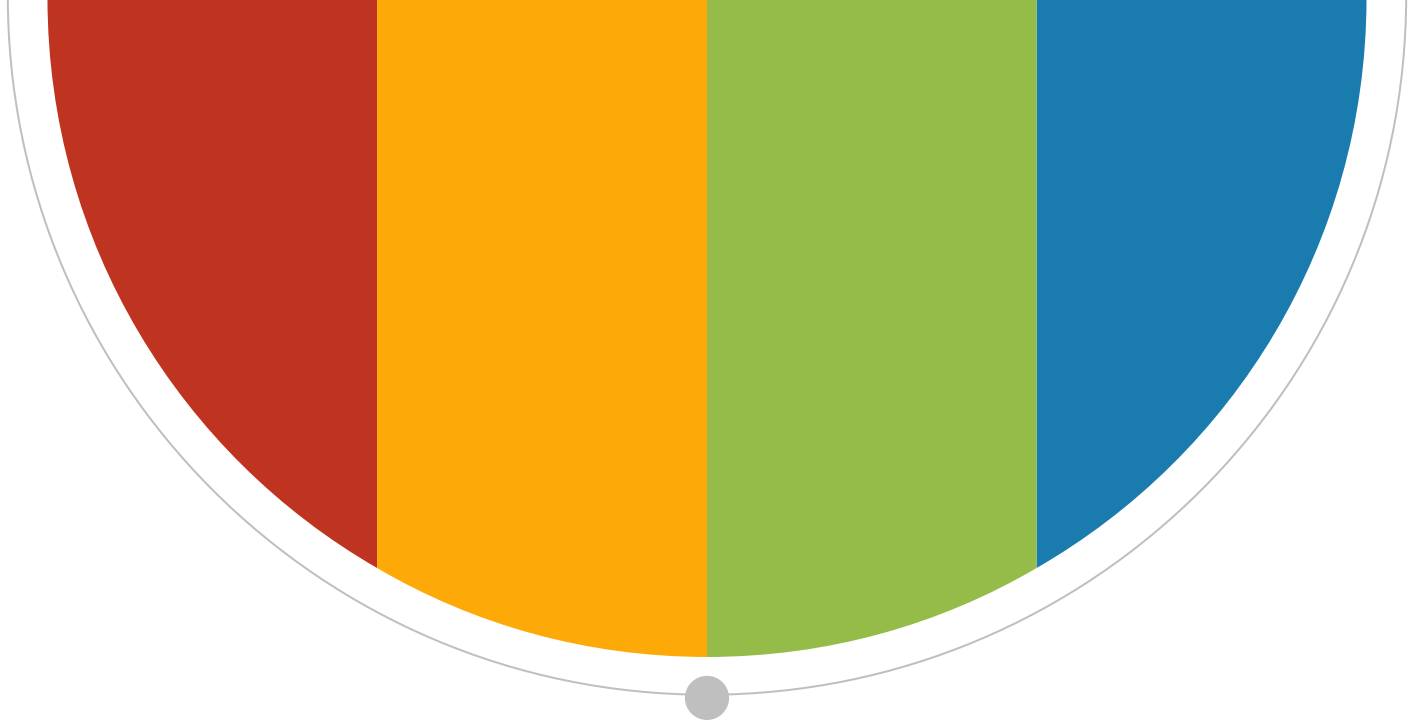


大数据处理与分析：Spark编程实践

王桂玲

北方工业大学信息学院数据工程研究院

大规模流数据集成与分析技术北京市重点实验室

wangguiling@ncut.edu.cn

1 Spark安装

2 PySpark及Jupyter Notebook环境

3 开发Spark应用程序

4 Spark集群环境搭建及使用

第四章 Spark环境搭建和使用方法

PART ONE

Spark的安装

Knowledge isn't free. You have to pay attention.

4.1 Spark 的安装

- 4.1.1 基础环境
- 4.1.2 下载安装文件
- 4.1.3 配置相关文件
- 4.1.4 Spark和Hadoop的交互

There's no shame in not knowing things! The only shame is to pretend that we know everything.

Richard Feynman

基础环境

- 安装Spark 3.0.0 之前需要安装Linux系统、Java环境、 Python环境
- Java版本必须在8u92版本之后
- Python要在3.6版本之后

JDK安装

安装JDK

- **第一步、下载JDK FOR LINUX**
- 下载地址：http://www.oracle.com/technetwork/java/javase/downloads/index.html
- 本实验下载的版本是jdk-8u261-linux-x64.tar.gz
- **第二步、指定目录，解压安装。**
- 当前目录下解压　　sudo tar zxvf
- 以最高权限拷贝到安装目录（例如，复制到/usr/lib/jvm下，可按个人习惯命名）

第三步.配置Java环境变量

- 1. 在/etc/profile.d下新建文件java_var.sh
- vi /etc/profile.d/java_var.sh
- 2. 加入JAVA_HOME、JRE_HOME、PATH和CLASSPATH到java_var.sh

```
export JAVA_HOME=/usr/lib/jvm/jdk1.8.0_261
export JRE_HOME=$JAVA_HOME/jre
export CLASSPATH=$CLASSPATH:.:$JAVA_HOME/lib:$JRE_HOME/lib
export PATH=$PATH:$JAVA_HOME/bin
~
```

- 3. 使配置生效
- source /etc/profile
- 4. 查看是否设置成功 （使用echo或printevn命令）
- echo $JAVA_HOME
- printenv JAVA_HOME

第四步. 配置默认JDK版本

改变系统bin默认java的指向

- 1.sudo update-alternatives --display java 查看安装了几个java及其优先级
- 2.sudo update-alternatives --config java 设置java的版本
- 例如，alternatives --config javac 设置javac的版本
- 输入上述命令，显示如下图，输入数字选择就可以啦。

```
[root@localhost bigdata]# update-alternatives --config java

共有 3 个提供“java”的程序。

  选项    命令
-----------------------------------------------
*  1           java-1.8.0-openjdk.x86_64 (/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.102-4.b
14.el7.x86_64/jre/bin/java)
   2           java-1.7.0-openjdk.x86_64 (/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.111-2.6
.7.8.el7.x86_64/jre/bin/java)
 + 3           /usr/lib/jvm/jdk1.8.0_261/bin/java

按 Enter 保留当前选项[+]，或者键入选项编号：
```

- 3.如果已手动安装的不在列表中，可以用下面的命令去添加到列表
- Update-alternatives --install /usr/bin/java java /usr/java/jdk1.8/bin/java 13000 （13000是优先级设置，一定要高于display查看到的其他版本java优先级设置）

第四步. 配置默认JDK版本

- 还需把/usr/bin/java下的所有java命令（javac, jar, javah, javap, jps） 指向/usr/lib/jvm/安装目录/bin/java并设置优先级为最高(优先级最高的为默认的程序使用路径).
- update-alternatives –install /usr/bin/java java /usr/lib/jvm/jdk1.8.0_261/bin/java 13000
- update-alternatives –install /usr/bin/jar jar /usr/lib/jvm/jdk1.8.0_261/bin/jar 13000
- update-alternatives –install /usr/bin/javac javac /usr/lib/jvm/jdk1.8.0_261/bin/javac 13000
- update-alternatives –install /usr/bin/javap javap /usr/lib/jvm/jdk1.8.0_261/bin/javap 13000
- update-alternatives –install /usr/bin/javah javah /usr/lib/jvm/jdk1.8.0_261/bin/javah 13000
- update-alternatives –install /usr/bin/jps jps /usr/lib/jvm/jdk1.8.0_261/bin/jps 13000

如果需要切换的时候使用如下命令：

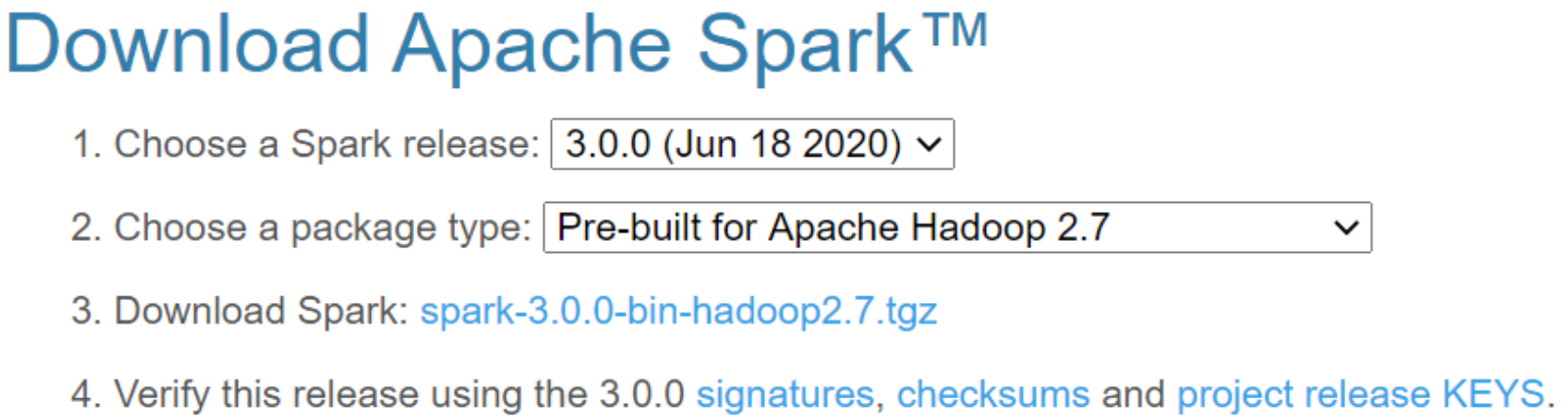
```
bruce@ubuntu:~$ sudo update-alternatives --config java
```

第五步.测试JDK安装配置结果

```
[root@localhost bigdata]# java -version
java version "1.8.0_261"
Java(TM) SE Runtime Environment (build 1.8.0_261-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.261-b12, mixed mode)
```

下载Spark安装文件

- Spark安装包下载地址： http://spark.apache.org
- 进入下载页面后，点击主页右侧的“Download Spark”按钮进入下载页面，下载页面中提供了几个下载选项，主要是Spark release及Package type的选择，如下图所示。第1项Spark release一般默认选择最新的发行版本，截至2020年8月份的最新版本为3.0.0（本教程采用3.0.0）。第2项package type则选择“Pre-built for Apache Hadoop 2.7”。选择好之后，再点击第3项给出的链接就可以下载Spark了。



下载安装文件

- 解压安装包spark-3.0.0….tgz至路径 /usr/local

配置相关文件

- 就可以启动、运行Spark了
- ./bin/run-example SparkPi 10

```
20/08/29 01:04:23 INFO TaskSchedulerImpl: Removed TaskSet 0.0, whose tasks have all co
pleted, from pool
20/08/29 01:04:23 INFO DAGScheduler: ResultStage 0 (reduce at SparkPi.scala:38) finish
d in 2.855 s
20/08/29 01:04:23 INFO DAGScheduler: Job 0 is finished. Cancelling potential speculati
e or zombie tasks for this job
20/08/29 01:04:23 INFO TaskSchedulerImpl: Killing all running tasks in stage 0: Stage
inished
20/08/29 01:04:23 INFO DAGScheduler: Job 0 finished: reduce at SparkPi.scala:38, took
.005043 s
Pi is roughly 3.137691137691138
20/08/29 01:04:24 INFO SparkUI: Stopped Spark web UI at http://192.168.48.128:4040
20/08/29 01:04:24 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint
topped!
20/08/29 01:04:24 INFO MemoryStore: MemoryStore cleared
20/08/29 01:04:24 INFO BlockManager: BlockManager stopped
20/08/29 01:04:24 INFO BlockManagerMaster: BlockManagerMaster stopped
20/08/29 01:04:24 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: Output
ommitCoordinator stopped!
20/08/29 01:04:24 INFO SparkContext: Successfully stopped SparkContext
20/08/29 01:04:24 INFO ShutdownHookManager: Shutdown hook called
20/08/29 01:04:24 INFO ShutdownHookManager: Deleting directory /tmp/spark-72e70b9d-238
-4fb3-816b-852c0e74b7b5
20/08/29 01:04:24 INFO ShutdownHookManager: Deleting directory /tmp/spark-de516c43-188
-4e55-b4d8-f82959927227
```

Spark部署模式

- Spark部署模式包括：
  - Local模式：单机模式
  - Standalone模式：使用Spark自带的简单集群管理器
  - YARN模式：使用YARN作为集群管理器
  - Mesos模式：使用Mesos作为集群管理器

在PySpark中运行代码

- PySpark提供了简单的方式来学习Spark API
- PySpark可以以实时、交互的方式来分析数据
- PySpark提供了Python交互式执行环境

- PySpark命令及其常用的参数如下：

```
pyspark --master <master-url>
```

- Spark的运行模式取决于传递给SparkContext的Master URL的值。Master URL可以是以下任一种形式：

  * local 使用一个Worker线程本地化运行SPARK(完全不并行)

  * local[*] 使用逻辑CPU个数数量的线程来本地化运行Spark

  * local[K] 使用K个Worker线程本地化运行Spark（理想情况下，K应该根据运行机器的CPU核数设定）

  * spark://HOST:PORT 连接到指定的Spark standalone master。默认端口是7077

  * yarn-client 以客户端模式连接YARN集群。集群的位置可以在HADOOP_CONF_DIR 环境变量中找到

  * yarn-cluster 以集群模式连接YARN集群。集群的位置可以在HADOOP_CONF_DIR 环境变量中找到

  * mesos://HOST:PORT 连接到指定的Mesos集群。默认接口是5050

- 在Spark中采用本地模式启动PySpark的命令主要包含以下参数：
  --master：这个参数表示当前的PySpark要连接到哪个master，如果是local[*]，就是使用本地模式启动PySpark，其中，中括号内的星号表示需要使用几个CPU核心(core)，也就是启动几个线程模拟Spark集群
  --jars： 这个参数用于把相关的JAR包添加到CLASSPATH中；如果有多个jar包，可以使用逗号分隔符连接它们

比如，要采用本地模式，在4个CPU核心上运行PySpark：

```
$ cd /usr/local/spark
$ ./bin/pyspark --master local[4]
```

或者，可以在CLASSPATH中添加code.jar，命令如下：

```
$ cd /usr/local/spark
$ ./bin/pyspark --master local[4] --jars code.jar
```

可以执行“pyspark --help”命令，获取完整的选项列表，具体如下：

```
$ cd /usr/local/spark
$ ./bin/pyspark --help
```

执行如下命令启动PySpark（默认是local模式）：

```
$ cd /usr/local/spark
$ ./bin/pyspark
```

启动PySpark成功后在输出信息的末尾可以看到“>>>”的命令提示符

- 可以在里面输入Python代码进行调试：

```
>>> 8*2+5
21
```

- 可以使用命令“exit()”或“ctrl+D”退出PySpark：

```
>>> exit()
```

2

PART TWO

PySpark及Jupyter Notebook

在Jupyter Notebook中使用PySpark

- sudo ln -s /usr/local/spark-3.0.0-bin-hadoop2.7 /usr/local/spark
- sudo vim /etc/profile.d/spark_var.sh

```
export SPARK_HOME=/usr/local/spark
export PATH=$SPARK_HOME/bin:$PATH
```

- 检查
- echo $PATH
- PySpark命令即可启动PySpark

- 再在上述/etc/profile.d/spark_var.sh中加入如下两句：
- export PYSPARK_DRIVER_PYTHON=jupyter
- export PYSPARK_DRIVER_PYTHON_OPTS='notebook’
- 这是设置PySpark的驱动器，从而使得它能够使用Jupyter Notebook:在PySpark启动运行时自动打开Jupyter Notebook

3

PART THREE

开发Spark独立应用程序

开发Spark独立应用程序

- 编写程序
- 通过spark-submit运行程序

运行程序

WordCount.py

```
1  from pyspark import SparkConf, SparkContext
2  conf = SparkConf().setMaster("local").setAppName("My App")
3  sc = SparkContext(conf = conf)
4  logFile = "file:///usr/local/spark/README.md"
5  logData = sc.textFile(logFile, 2).cache()
6  numAs = logData.filter(lambda line: 'a' in line).count()
7  numBs = logData.filter(lambda line: 'b' in line).count()
8  print('Lines with a: %s, Lines with b: %s' % (numAs, numBs))
```

对于这段Python代码，可以直接使用如下命令执行：

```
$ cd /usr/local/spark/mycode/python
$ python3 WordCount.py
```

执行该命令以后，可以得到如下结果：

```
Lines with a: 62, Lines with b: 30
```

通过Spark-submit运行

可以通过spark-submit提交应用程序，该命令的格式如下：

```
spark-submit
  --master <master-url>
  --deploy-mode <deploy-mode>   #部署模式
  ... #其他参数
  <application-file>  #Python代码文件
  [application-arguments]  #传递给主类的主方法的参数
```

可以执行“spark-submit --help”命令，获取完整的选项列表

```
$ cd /usr/local/spark
$ ./bin/spark-submit --help
```

- Master URL可以是以下任一种形式：
  * local 使用一个Worker线程本地化运行SPARK(完全不并行)
  * local[*] 使用逻辑CPU个数数量的线程来本地化运行Spark
  * local[K] 使用K个Worker线程本地化运行Spark（理想情况下，K应该根据运行机器的CPU核数设定）
  * spark://HOST:PORT 连接到指定的Spark standalone master。默认端口是7077.
  * yarn-client 以客户端模式连接YARN集群。集群的位置可以在HADOOP_CONF_DIR 环境变量中找到。
  * yarn-cluster 以集群模式连接YARN集群。集群的位置可以在HADOOP_CONF_DIR 环境变量中找到。
  * mesos://HOST:PORT 连接到指定的Mesos集群。默认接口是5050。

通过 spark-submit 提交到 Spark 中运行，命令如下：

```
$ /usr/local/spark/bin/spark-submit  /usr/local/spark/mycode/python/WordCount.py
```

可以在命令中间使用“\”符号，把一行完整命令“人为断开成多行”进行输入，效果如下：

```
$ /usr/local/spark/bin/spark-submit  \
> /usr/local/spark/mycode/python/WordCount.py
```

上面命令的执行结果如下：

```
Lines with a: 62, Lines with b: 30
```

为了避免其他多余信息对运行结果的干扰，可以修改log4j的日志信息显示级别：

log4j.rootCategory=INFO, console

修改为

log4j.rootCategory=ERROR, console

4

PART FOUR

Spark集群环境搭建及使用

Spark集群环境搭建

- 集群概况
- 准备工作：搭建Hadoop集群环境（**单机伪分布式集群环境搭建参见实验手册文档：CentOS下搭建单机伪分布式Hadoop及HDFS文件系统；多机Hadoop分布式集群环境搭建请参见Hadoop官方文档**）
- 安装Spark
- 配置环境变量
- Spark配置
- 启动Spark集群
- 关闭Spark集群

Thank you