一、 Spark 集群使用

1. 执行第一个 spark 程序

```
/usr/local/spark-1.5.2-bin-hadoop2.6/bin/spark-submit \
--class org.apache.spark.examples.SparkPi \
--master spark://node1.itcast.cn:7077 \
--executor-memory 1G \
--total-executor-cores 2 \
/usr/local/spark-1.5.2-bin-hadoop2.6/lib/spark-examples-1.5.2-hadoop2.6.0.jar \
100
```

该算法是利用蒙特·卡罗算法求 PI。

2. 启动 Spark Shell

spark shell 是 Spark 自带的交互式 Shell 程序，可在任一台机器上启动 spark shell，方便用户进行交互式编程，可在该命令行下用 scala 编写 spark 程序，或用于向 spark 集群提交应用程序。

2.1. 启动 spark shell

启动集群版 spark shell：

```
/usr/local/spark-1.5.2-bin-hadoop2.6/bin/spark-shell \
--master spark://node1.itcast.cn:7077 \
--executor-memory 2g \
--total-executor-cores 2
```

此时，通过 spark 管理界面看到 Running Applications，即 spark-shell 已经连接到 spark 集群。

通过 jps 命令查看，启动了 SparkSubmit 进程。SparkSubmit 是 Spark 客户端，与集群通信，将任务提交到集群。
若不带参数，则启动 spark 本地客户端，没有连接到 spark 集群。

参数说明：
--master spark://node1.itcast.cn:7077 指定 master 地址（如果有多个，可用逗号分隔）
--executor-memory 2g 指定每个 Worker 上启动 Executor 占用的内存为 2G
--total-executor-cores 2 指定任务在整个集群中使用的 cpu 核数为 2 个（cpu 核数即处理器的

个数）

注意：

（1）Executor 是 Worker 的子进程，专为 spark shell 计算提供服务。

（2）Spark Shell 默认将 SparkContext 类初始化为 sc 对象，用户代码如果需要可直接使用。

（3）创建 RDD 的两种方式：从外部存储介质读取数据生成 RDD；将 scala 集合转换成 RDD。

2.2. 在 spark shell 中编写 WordCount 程序

从 hdfs 读取数据，使用 spark 计算，再将结果写入 hdfs。

1.启动 hdfs：sbin/start-dfs.sh；

2.上传文件到 hdfs：hdfs dfs -put words.txt hdfs://node1.itcast.cn:9000/words.txt；

3.启动 spark：sbin/start-all.sh；

4.启动 spark shell 连接 spark 集群（命令同上）；

5.在 spark shell 中用 scala 编写 spark 程序

```
sc.textFile("hdfs://node1.itcast.cn:9000/words.txt").flatMap(_.split(" "))
.map((_,1)).reduceByKey(_+_).saveAsTextFile("hdfs://node1.itcast.cn:9000/out")
```

6.使用 hdfs 命令查看结果

```
hdfs dfs -ls hdfs://node1.itcast.cn:9000/out/p*
```

说明：

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

textFile(hdfs://node1.itcast.cn:9000/words.txt)是到 hdfs 中读取数据；

flatMap(_.split(" "))先切分再压平；

map((_,1))将单词和 1 构成元组；

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

saveAsTextFile("hdfs://node1.itcast.cn:9000/out")将结果写入到 hdfs 中；

补充：

（1）partitioner 决定 spark 计算数据在哪个分区，决定 mapreduce 计算数据在哪个 reducer。

（2）启动高可用的 HDFS 时会陆续启动 namenode，datanode，journalnode，zkfc。zkfc 用来监控 namenode 状态并向 zookeeper 汇报。