计算机科学与技术学院 大数据管理与分析 课程实验报告

实验题目：PageRank 算法实现		**学号：**201605130116
日期：2019.5.5	**班级：**2016 级泰山学堂	**姓名：**杜洪超
Email：1503345074@qq.com		

实验目的：

PageRank 网页排名的算法，曾是 Google 发家致富的法宝。用于衡量特定网页相对于搜索引擎索引中的其他网页而言的重要程度。通过对 PageRank 的编程在 Hadoop 和 Spark 上的实现，熟练掌握 MapReduce 程序与 Spark 程序在集群上的提交与执行过程，加深对 MapReduce 与 Spark 的理解。

实验软件和硬件环境：

软件环境：

系统：Ubuntu16.04 LTS 64 位

软件：openjdk-7-jre,openjdk-7-jdk,java1.7.0_95

Hadoop 2.9.2

Spark 2.3.0

Scala 2.11.8

Eclipse,ssh

硬件环境：

CPU: Intel® Core™ i5-6260U CPU @ 1.80GHz × 4

磁盘：121.8 GB

内存：7.7 GiB

实验原理和方法：

1. 分别 Hadoop 和 Spark 实现 PageRank 算法，对比两种实现

实验步骤：（不要求罗列完整源代码）

1. Hadoop 实现
   1. GraphBuilder 类

      对数据进行处理，将（url，links）格式的数据转换为（url，rank.lists）

      Map 函数负责处理初始数据，去掉多余的 tab 和分隔符，并将初始 rank 值赋给每一个 url；

      Reduce 函数不做处理，直接输出 key 和 value

```
import java.util.StringTokenizer;

public class GraphBuilder
{
        public static class GraphBuilderMapper extends Mapper<Object, Text, Text, Text>
        {
            protected void map(Object key, Text value, Context context)
        }

      public static class GraphBuilderReducer extends Reducer<Text, Text, Text, Text>
      {
          public void reduce(Text key, Text value, Context context)
      }

      public static void main(String[] args) throws Exception
}
```

2. PageRankIter
   用于生成每次迭代结果。
   Map 函数对 key 所对应的 links 中的每一个链接，计算当前 url 对它的 PR 值的贡献，最后生成（key，links）维护链接关系
   Reduce 函数将所有 url 的 PR 值加起来，再次生成（url，rank，links）的 key-value 对，供下一次迭代使用；

```
import java.util.Iterator;

public class PageRankIter
{
      public static class  PageRankIterMapper extends Mapper<Object, Text, Text, Text>
      {
        protected void map(Object key, Text value, Context context)
      }

      public static class PageRankIterReducer extends Reducer<Text, Text, Text, Text>
      {
          protected void reduce(Text key, Iterable<Text> values, Context context)
      }

      public static void main(String[] args) throws Exception
}
```

3. RankViewer 类
   用于生成最后结果
   Map 函数把 rank 作为 key，key 作为 value，这样可以实现按 pr 值排序
   通过重载比较函数可以实现从大到小排序
   Reduce 函数格式化输出结果

```
import java.io.IOException;

public class RankViewer
{
    public static class  RankViewerMapper extends Mapper<Object, Text, DoubleWritable, Text>
    {
        protected void map(Object key, Text value, Context context)
    }

    public static class RankViewerReducer extends Reducer<DoubleWritable, Text, Text, Text>
      {
          protected void reduce(DoubleWritable key, Iterable<Text> values, Context context)
      }

      public static class DescDoubleComparator extends DoubleWritable.Comparator

      public static void main(String[] args) throws Exception
}
```

4. PageRankDriver 类
   总控程序，负责控制迭代次数和调用各类的次序，以及对应的目录参数

2. Spark 实现
   1. 用 scala 语言实现 PageRank，实现思路和 MapReduce 类似，

结论分析与体会：

通过对 PageRank 的编程在 Hadoop 和 Spark 上的实现，掌握了 MapReduce 程序与 Spark 程序在集群上的提交与执行过程，加深对 MapReduce 与 Spark 的理解。

Spark 用 scala 实现相比 Hadoop 的实现，代码长度显著减少，但需要学习 scala 的基本语法和适应编程思路，在实际编程中二者花费时间相差不多，但前者有相当一部分是学习成本。

实际计算中，Spark 要快于 Hadoop，体现了 Spark 的优势。

就实验过程中遇到和出现的问题，你是如何解决和处理的，自拟 1－3 道问答题：

1. Scala 语言直接打包成 jar 比较困难，最终选择借助 Intellij IDEA 实现打包
2. 不知道因为什么原因，spark 所输出的结果精度不够，不能实现 double 的精度，只能保证 float 精度，目前仍没有解决，不过在另一台电脑上执行结果正确，猜测可能是电脑或者环境配置甚至 IDE 的原因。