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：201705130113 |
| 日期：2020.4.24 | 班级：2017级计算机3班 | 姓名：黄瑞哲 |
| Email：jeshrz063@gmail.com | | |
| 实验目的：   1. 编程实现PageRank算法，理解该算法的思想与实现方式 2. 熟练掌握MapReduce程序与Spark程序的编写以及在集群上的提交与执行过程 3. 加深对MapReduce与Spark的理解 | | |
| 实验软件和硬件环境：  操作系统：macOS Catalina 10.15.4  Hadoop版本：2.8.5  Spark版本：2.4.5  JDK版本：1.8.0\_241  Java IDE: IntelliJ IDEA | | |
| 实验原理和方法：  PageRank本质上是一种投票算法。假如一共有ABCD四个页面，如果页面BCD都唯一地链接到了页面A，那么页面A的PageRank将是页面BCD的PageRank的总和。  如果页面B不仅链接到页面A，还链接到了页面C；页面D链接到了ABC。那么在投票的时候，B会给A和C个半票，D会给ABC个三分之一票  那么就可以得到朴素PageRank算法的迭代公式  但是这样会导致排名泄露（某个网页没有外链）以及排名下沉（某个网页没有被任何页面链接）的情况，由此引入了阻尼系数d来解决  其中N是页面总数。这个公式可以被简化为  这个公式不会影响各个页面的相对排名，也是有效的。  具体计算的时候一轮一轮不断迭代即可，可以指定迭代的次数，也可以判断当PageRank变化不大时结束迭代。 | | |
| 实验步骤：（不要求罗列完整源代码）   1. MapReduce 2. Graph Builder阶段   这个阶段的任务是将原输入数据处理一下，得到每个URL的邻接表   1. Map函数   首先把读入的每一行按\t分割得到URL和他的所有的邻接URL，然后把邻接的URL按照逗号隔开，转换成Java可以识别的数据结构links。最后把URL作为key，PR\_init \t links作为value输出。     1. Reduce函数   不需要做任何改动，把Map函数的输入直接输出即可。   1. PageRank Iteration阶段   这一阶段主要处理上一阶段迭代的结果，然后把新一轮的迭代结果输出。   1. Map函数   按照格式把value进行分割，得到当前节点的URL，该节点目前的PageRank以及这个节点的所有邻接点。  对于每个邻接点v，都会有 PageRank/|link| 的贡献，那么把v作为key，贡献值作为value输出。同时为了把拓扑关系传递下去，也需要把link输出，即对于每一个邻接点v，把u作为key，v作为value输出。     1. Reduce函数   Reduce函数收到了Map函数聚合以后的输出。对于每一个value，如果他是贡献值，那么把它累加起来，记为sum，如果是节点则将其组织成为Java可以识别的数据结构link。处理完成以后则套用公式计算出这个节点新的PageRank。然后按照Graph Builder阶段的Reduce输出格式输出。     1. Rank View阶段   这一阶段主要收集最后一轮迭代的输出结果，提取出其中的PageRank以后按照PageRank的降序排序格式化输出。   1. Map函数   对于排序而言，可以把PageRank作为key，URL作为value进行输出，这样根据MapReduce的默认设置会按照key的升序进行排序，此时可以重载key的类型，重写它的比较函数compareTo，使其按照降序排序       1. Reduce 函数   Reduce收到的key是PageRank，value是URL，那么把它反过来进行格式化输出即可，PageRank结果保留10位小数。     1. Spark   Spark主要依靠RDD来计算，相比较MapReduce，它减少了I/O，使得计算速度更快。对于这个问题而言，首先根据数据集构造出邻接表    同时初始化每个节点的PageRank    在此之后就可以开始迭代了。在每一轮迭代中，计算出每个节点得到的贡献值是多少，然后对于每个节点先将贡献值累加，然后套用公式计算出这一轮迭代的PageRank。    当迭代结束以后，ranks这个RDD里存的就是每一个节点的PageRank，将其按照PageRank降序排序以后格式化输出到文件即可。 | | |
| 结论分析与体会：   1. MapReduce的输出      1. Spark的输出       可以看到MapReduce的输出与Spark的输出是一致的，与标准答案没有差别，结果正确。   1. 体会   不得不说，Spark写的程序要比MapReduce写的程序短了不少，在MapReduce里实现PageRank算法的Java代码，我一共用了193行来实现，而在Spark里我只用了56行。而且在自己笔记本上运行的时候可以明显感觉到Spark运行的速度要比MapReduce快。就我而言，Spark唯一不太好的地方就是语法奇怪，有很多奇奇怪怪的用法，没有MapReduce写起来顺手。 | | |
| 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：  Spark在计算结束的时候会输出两个文件，如何让Spark写到同一个文件里去？  答：最终每个节点的PageRank存储在一个RDD中，最终这个RDD在执行saveAsTextFile写文件的时候，RDD里面有多少个分区，就会写多少个文件（一个分区一个文件），如果想写到同一个文件里的话，可以用过函数coalesce(1,true)把多个分区的数据放到同一个分区里。但使用这个函数的时候需要注意：Spark处理的可能是大数据，如果把所有数据放到同一个分区里，需要注意内存是否可以存放这么多的数据。 | | |