**云计算课程实验**

2019.10.16

1. **实验目的**
   1. 熟悉云计算平台Spark原理，熟悉编程语言scala和编程环境IDEA。
   2. 学习使用开源图计算平台GraphX，了解Pagerank算法的实现流程与优化方式。
2. **实验要求**

使用GraphX API，将源数据集数据抽象为图数据格式（实验1），使用pagerank算法解决Wikipedia投票选举问题（实验2）。

1. **实验环境配置**
2. **搭建Java开发环境，配置环境变量**

http://www.oracle.com/technetwork/java/javase/downloads/index.html

1. **安装hadoop，Spark并配置**

**Ubuntu环境搭建：**

**Hadoop配置：**http://www.cnblogs.com/kinglau/p/3794433.html

**Spark配置：**https://www.cnblogs.com/lijingchn/p/5573898.html

**Windows环境搭建：**

https://blog.csdn.net/u013963380/article/details/72677212

1. **安装IDE环境Intellij IDEA**

https://www.jetbrains.com/idea/download/

1. **引入graphX，学习graphX API的使用方式。**

<https://spark.apache.org/graphx/>

https://spark.apache.org/docs/latest/graphx-programming-guide.html

1. **实验数据说明**
2. 数据集来自**SNAP networks**，网址为http://snap.stanford.edu/data/index.html，数据集下载解压后得到txt文件。
3. **Wikipedia图**是由7115个节点和103689条边组成，**Google图**是由875713个节点和5105039条边组成。
4. 每一行代表一条边，FromNodeId是边的起始节点ID，ToNodeId是边的终止节点ID，方向代表他们之间的引用关系，A->B代表A投票给B，或A引用了B，即A认为B很重要，将A的一部分重要性赋予B。
5. **实验流程**
6. **实验1(40%)：graphX API练习**

* 使用Pregel实现SSSP(single source shortest path)：参考Pregel API, GraphOps
* 数据预处理：参考Vertex and Edge RDDs，GraphLoader.edgeListFile

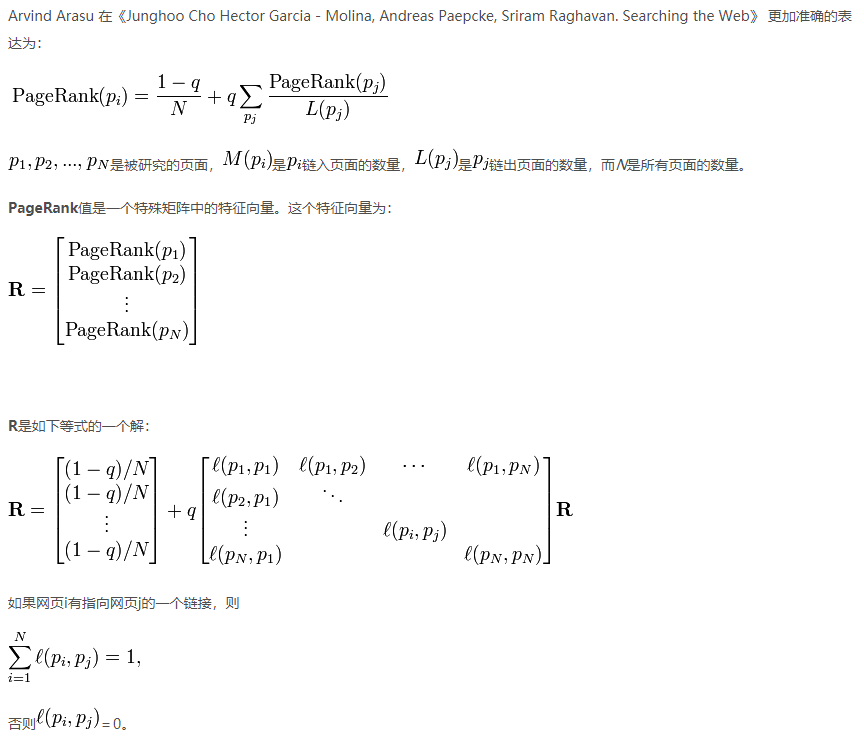
1. **实验2(60%)：使用PageRank解决Wikipedia投票选举问题**
   * + 下载数据集http://snap.stanford.edu/data/wiki-Vote.html
     + 按照**实验1**中的数据预处理方法预处理所给的实验数据
     + 参考使用aggregateMessages，org.apache.spark.graphx.lib
     + **实验结果：选出声望最高的前20名候选人ID。**
2. **实验提交与评分标准**
3. 需要将代码(scala文件和readme.txt)和实验报告(2页及以上)一并打包(zip或rar)并提交到助教邮箱(jing618@sjtu.edu.cn)，每人提交一份。**打包文件命名方式（CC2019\_姓名.rar）**。
4. 报告整洁度，实验设计的清晰度，实验结论的完整度（图表、解释说明等）会影响评分。
5. 请在Spark提供的GraphX API及其他基本API基础上进行操作。未完全使用Spark API会酌情扣分。
6. **一些参考：**
7. **Pagerank的原理**

PageRank,即网页排名，又称网页级别、Google左侧排名或佩奇排名。是Google创始人拉里·佩奇和谢尔盖·布林于1997年构建早期的搜索系统原型时提出的链接分析算法，目前很多重要的链接分析算法都是在PageRank算法基础上衍生出来的。

对于某个网页A来说，该网页PageRank的计算基于以下两个基本假设：**数量假设**：在Web图模型中，如果一个页面节点接收到的其他网页指向的入链数量越多，那么这个页面越重要。**质量假设**：指向页面A的入链质量不同，质量高的页面会通过链接向其他页面传递更多的权重。所以越是质量高的页面指向页面A，则页面A越重要。

**执行步骤：**1).在初始阶段：网页通过链接关系构建起Web图，每个页面设置相同的PageRank值，通过若干轮的计算，会得到每个页面所获得的最终PageRank值。随着每一轮的计算进行，网页当前的PageRank值会不断得到更新。2).在一轮中更新页面PageRank得分的计算方法：在一轮更新页面PageRank得分的计算中，每个页面将其当前的PageRank值平均分配到本页面包含的出链上，这样每个链接即获得了相应的权值。而每个页面将所有指向本页面的入链所传入的权值求和，即可得到新的PageRank得分。当每个页面都获得了更新后的PageRank值，就完成了一轮PageRank计算。

简单计算：



摘自https://blog.csdn.net/hguisu/article/details/7996185

1. **图计算框架GraphX**

GraphX 是Spark中用于图(graph)和图并行计算的一个新组件。在高层次上，GraphX通过引入一个新的图抽象，扩展了Spark RDD:一个有向多重图，属性被附加到每个顶点和边缘。为了支持图计算，GraphX公开了一组基本操作符(例如，subgraph、joinVertices和aggregateMessages)，以及Pregel API的一个优化变体。此外，GraphX还包含了越来越多的图算法和图构建器，用于简化图计算任务。

摘自 https://spark.apache.org/docs/latest/graphx-programming-guide.html