互联网数据挖掘作业报告——PageRank

学号：1400012783　姓名：王君珊

本次作业用C++实现了PageRank算法，并基于ANN论文数据集(2013年)计算Paper、Author和Venue的PageRank值。

PageRank是一种根据网页之间相互的超是链接计算的技术，作为网页排名的重要因素。

PageRank的迭代计算公式如下：$\pi=\alpha P^T+(1-\alpha)\frac{1}{n}e$, $e=(1,1,\ldots,1)^T$

一、程序介绍：

本次作业有三个程序，分别计算了Paper、Author和Venue的PageRank值。其中，Author和Venue的程序结构相似，而Paper的结构与前两者稍有不同。

1. PaperPageRank：

程序：PaperPageRank源.cpp

输出文件：paperPagerank.txt

Paper的PageRank值计算相对麻烦。首先，由于paper的ID是字符串的形式，且位数大，故不能直接将其作为索引，而是运用类似Hash的方法，为其重新映射一个新的ID。同时由于paper数量较大，直接开辟N*N的p矩阵对内存要求较高，故这里运用了类似稀疏矩阵的方法，每个结点为一个结构体，存储了指向它的结点及其p值。在每轮迭代计算PageRank值时，遍历每个结点，根据其所有父节点的PageRank值和相对应的p值，求和得到新的PageRank值。

计算Paper的PageRank值用到了三个文件：paper_outcites.txt, acl.txt, paper_ids.txt。paper_outcites.txt用来得到p矩阵的值，acl.txt得到paper之间的引用关系，paper_ids.txt得到paper的名字。

需要说明的是，一共有21212篇paper，但是存在引用关系的只有18164篇，也就是说有部分paper既没有引用也没有被引用，程序仍然计算了它们的PageRank值。

2. AuthorPageRank：

程序：AuthorPageRank源.cpp

输出文件：authorPagerank.txt

Author的PageRank程序中的数据结构相对直观，用一个N*N的矩阵直接存储p值，每轮迭代直接运用文章开头的公式，计算新的PageRank值。

计算author的PageRank值用到了两个文件：author_citation_network.txt, author_ids.txt。author_citation_network.txt得到author之间的引用关系，author_ids.txt得到author的名字。

需要注意的是，author_ids.txt文件中的ID并不是连续的，中间有部分ID号是被跳过的，若直接以此ID作为索引会影响PageRank的计算，故程序为它们重新安排序号。

3. VenuePageRank：

程序：VenuePageRank源.cpp

输出文件：venuePagerank.txt

Venue的PageRank程序结构与author的类似，唯一不同的是venue之间的引用关系需要通过paper的引用关系获得。所以需要进行数据预处理，即得到每篇paper所属的venue。数据预处理后，用类似Paper的输入方法读入paper的引用关系，转换成venue之间的引用关系，用类似author的迭代方法计算。

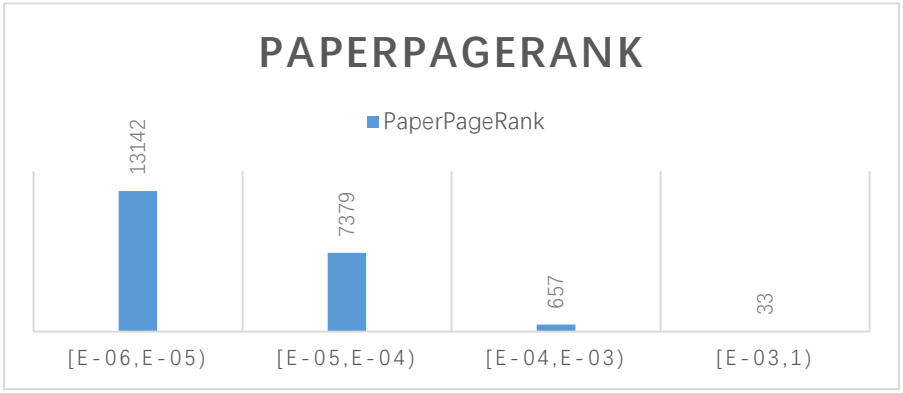
计算venue的PageRank值用到了两个文件：acl.txt, acl-metadata.txt。acl.txt得到paper之间的引用关系，进而得到venue的引用关系，acl-metadata.txt得到paper和venue的对

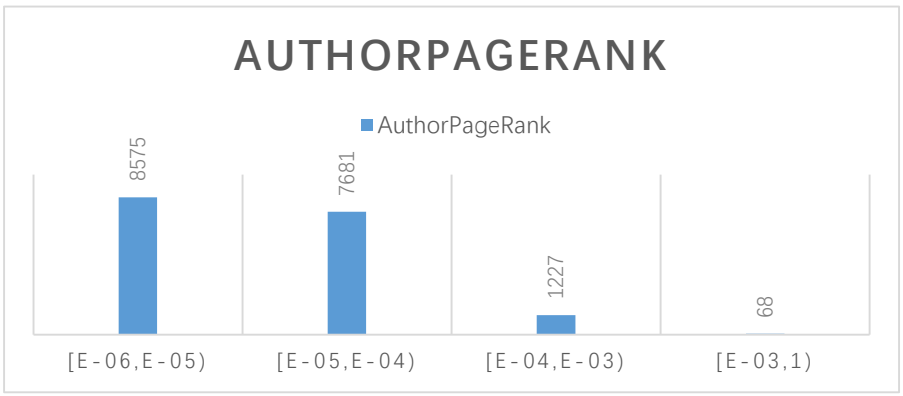
应关系和 venue 的名字。

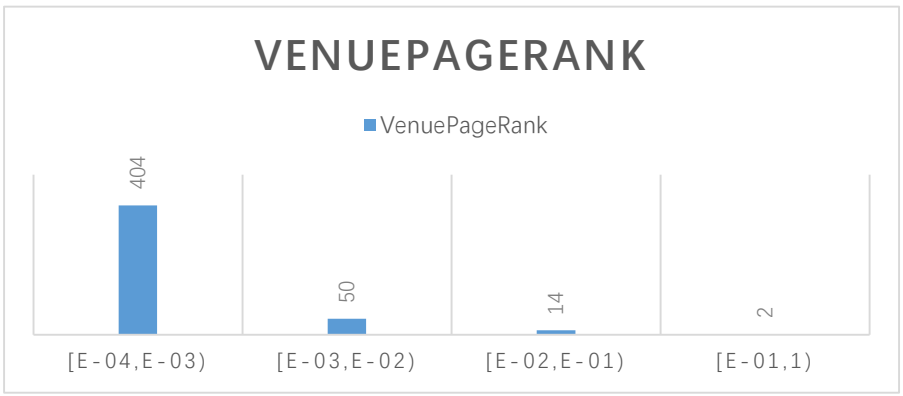
需要注意的是，对于两个 venue 之间多篇 paper 引用的情况，处理方法是相当于两个 venue 之间有多条边，p 值相对应进行加权，即引用越多，p 值会越大。

二、数据汇报与分析分析：

排序后的 PaperPageRank、AuthorPageRank 和 VenuePageRank 存在相应的 txt 文件中，各自 PageRank 的分布情况如下表：







可以看出，无论是 paper、author 还是 venue，数量都会随着 PageRank 值变大而减少，也就是说大部分的 PageRank 值都在较低水平，只有极少量的 Paper、Author 或者 Venue 有较高的 PageRank 值，即有较高的引用量，处于顶尖水平。

三、总结

本次作业的难点我认为有主要两点：一是对文本文件的处理，即如何从大量格式不规范的数据中获得有用信息，并转换为可以计算的数据；二是对于大量的数据，如何在保证计算正确的前提下，降低时间和空间的复杂度。