搜索引擎大作业书面报告

侯昊迪 131180105

1. 问题描述

本次作业的要求是实现一个简单的搜索引擎。具体功能包括实现布尔操作(包括与、或、非)，对搜索结果进行排序(包括按照引用次数、关键词出现次数、pagerank三个标准)。

为了实现这些功能，首先我们需要对文档数据进行读取和处理，将其以方便的形式存储到计算机内存中。然后需要对这些文档进行遍历统计，构建倒排表，并对倒排表中的词条实现与或非的操作。最后，需要实现对返回的结果进行排序的算法。

2. 数据结构设计

2.1 链表

链表是本次作业中的基础数据结构类型之一，由于链表容量可变、便于插入操作等特点，在搜索引擎的实现中有着广泛的用途，具体包括：在用邻接表实现图结构时用于存储结点的边，在实现散列表结构时用于处理冲突。

2.2 可变数组

数组具有能够随机访问其中元素的特点，使用非常方便。但是由于基本数组结构的最大容量不可变，在使用中的灵活性较差，因此有必要实现容量可变的数组结构，这样在编程时就不需考虑数组的大小。在本次作业中，可变数组应用到了一下几个方面:一是用于实现字符串结构，二是用于存储论文的信息，三是用于在倒排表中存储词条对应的文章。

2.3 散列表

在本次作业中，散列表主要用于存储倒排表。考虑到搜索引擎需要根据字符串查找匹配结果，并且词条数目较大，使用散列表进行存储，查询时的时间复杂度即为 hash 函数的计算时间，可以认为是常数时间，并且方便用于根据字符串查询结果。

2.4 图

文章之间的引用关系和网页之间的链接关系自然构成了图的结构。在搜索引擎中，由于需要对检索结果进行排序，而无论是根据引用次数还是 pagerank 进行排序

都需要通过图结果对每个节点的出度和入度进行统计，因此图机构也是本次作业中需要实现的数据结构之一。考虑到在论文引用或网页链接构成的图结构中边的数量比较稀疏，因此采用邻接表来实现图结构。另外，论文的引用和网页的链接明显是有向的，因此本次作业中需要实现的是基于邻接表的有向图结构，并提供查询结点出度和入度的函数接口。

2.5 栈

栈的特点是先进后出，常常用于一些递归算法的非递归实现中。在本次作业中，一方面若涉及到图的遍历，就需要用到栈结构的辅助；另一方面，在布尔查询中，为了支持用与或非符号以及括号连接的字符串算式，我们也需要用到栈结构来对算式进行计算。

3. 算法设计

3.1 二分查找算法

在构建词条时，须将包含词条的文章(id 或地址)存储到词条中，而由于一篇文章中可能同一词条出现多次，因此，在检索到一个词时首先需要查询该文章是否已插入到该词条中，这就需要在词条已有的文章目录中查找某一文章是否存在。但是遍历搜索的方法的时间复杂度为 O(n)，考虑到词条中的文章目录是按照文章 id 排序的，因此使用二分查找算法，能够将时间复杂度降到 O(logn)，大大提高了效率。

3.2 归并排序

搜索结果需要进行排序，无论是根据引用次数、词条出现次数还是 pagerank 进行排序，都是根据一个 key 值对结果进行排序。因此排序算法的效率至关重要，基于比较的排序算法时间复杂度的下限是 nlog(n)，虽然快速排序算法在实际应用中效率往往较高，但考虑到其排序结果不稳定，因此本次作业中采取了稳定且高效的归并排序。

3.3 pagerank

Pagerank 的关键在于计算一个表征网页重要程度的值，而根据 pagerank 的计算方法，我们可以将这个值理解为用户上网时浏览到该网页的概率。首先，考虑直接访问的情况，即用户不通过其他网页的链接而直接访问该网页，这种情况下，用户访问该网页的概率为一个固定值，即不会受到链接数的影响，在具体实现中，

我们可以假设对于所有网页这种访问是完全随机的，即每个网页被直接访问的概率相等。另一种情况就是网页通过其他网页的链接被访问到。在这种情况下，网页被访问到的概率不仅取决于链接到该网页的网页数量，还取决于链接到该网页的网页被访问的概率，以及这些网页链接到其他网页的数量。因此，某一网页被访问的概率可由以下公式计算：

$$R_{(u)} = c\sum_{v\in B_u}\frac{R_v}{N_v} + cE_u$$

其中，$R_{(u)}$ 表示网页 u 被访问的概率，B_u 为链接到网页 u 的网页集合，N_v 为网页 v 向外链接的网页数量，E_u 为网页 u 被直接访问的概率，c 为常数。

在具体计算时，可以通过多次迭代计算，直到 $R_{(u)}$ 收敛为止。

4. 具体实现

除了以上数据结果及算法方面的内容外，本次作业完成过程中还有一些需要解决的具体问题，在这里将这些问题中的主要问题做简要说明。

4.1 文档处理

由于本次作业使用的数据集中的内容比较复杂，包括大小写、符号、ASCII 码之外的字符、无意义的词汇等问题。本作业采取的处理方式是只对包括两个及以上各英文字母的词汇进行统计检索，并且在读取、存储时一律用小写字母，查询时也只支持小写字母。

4.2 效率问题

由于本次作业中的数据量较大，文件的读取量也较大，因此除通过前面提到的数据结构及算法设计来提高效率以外，具体的实现中海采取了一些策略来提高效率。具体如下：

(1)整个程序只存储一份完整的文章数据，其他地方例如倒排表等均通过指针或引用进行存取等操作。

(2)将查询公式(包括单个词汇和由与或非算符以及括号连接组成的算式)都视为一个词条，在返回结果时，对每个词汇的词条带入算式中计算得到一个新的词条结果，并返回该词条的副本，然后对副本中的文章目录进行排序并展示，而不影响程序中唯一的一份完整文章数据的内容。