算法设计与分析课程设计（2021-2022）

本次课程设计拟通过少量的代码实现一个**小型搜索引擎**[1]。麻雀虽小，五脏俱全。跟大型搜索引擎相比，实现这样一个小型搜索引擎所用到的理论基础是相通的。

搜索引擎大致可以分为四个部分：搜集、分析、索引、查询。

搜集 就是我们常说的利用爬虫爬取网页。

分析 网页内容抽取、分词，构建临时索引等。

索引 通过分析阶段得到的临时索引构建倒排索引。

查询 响应用户的请求，根据倒排索引获取相关网页，计算网页排名，返回查询结果给用户。

课程设计内容包括根据提供的爬取文件进行分词、索引，最终实现查询功能。实现过程中需要用到的排序等算法需自行完成，算法主体部分不允许调用任何库中已有算法。

具体说明

1. 搜集

任务书以附件形式提供名为“news.csv”的网页爬取文件，包含在人民日报英文版上爬取的新闻页面。文件内容格式为“网址，标题，网页内容”，其中网址以http开头，网址与标题在同一行，网页内容包括多行。

搜集阶段只需将网页相关信息进行提取并对每个网页进行编号（doc_id）。得到页面文件。

[1] 题目灵感及部分内容来自于“极客时间”，https://time.geekbang.org/。

提高：附件还了提供名为“SearchEngineLite.zip”的压缩包，其中包含一个python语言写的scrapy爬虫[2]，是爬取“news.csv”文件的源码。感兴趣的同学可以此为基础修改并自行爬取页面。

2. 分析

对页面信息进行读取并进行分词，创建临时索引。对于英文网页，只需要通过空格、标点符号等分隔符，将每个单词分割开来就可以了。

每个网页的文本信息在分词完成之后，都得到一组单词列表。把单词与网页之间的对应关系写入到一个临时索引文件中。这个临时索引文件将来会用来构建倒排索引文件。临时索引文件的格式如图1。

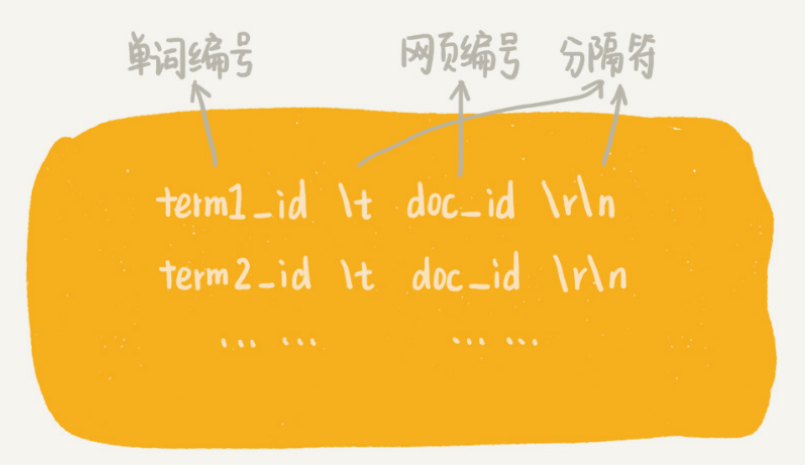

图 1 临时索引文件格式

临时索引文件中存储的是单词编号，也就是图中的 term_id，而非单词本身。其目的是节省存储空间。

需维护一个计数器，每当从网页文本信息中分割出一个新的单词的时候，就从计数器中取一个编号分配给它，然后计数器加一。这个过程中还需要记录已经编过号的单词。

[2] scrapy 学习资料：https://docs.scrapy.org/en/latest/intro/tutorial.html

在对网页文本信息分词的过程中，我们拿分割出来的单词先查找是否已经编号，如果找到，那就直接使用已有的编号；如果没有找到，就去计数器中拿号码，并且将这个新单词以及编号记录下来。

当所有的网页处理（分词及写入临时索引）完成之后，我们再将这个单词跟编号之间的对应关系写入到磁盘文件中。经过分析阶段，我们得到了两个文件，即临时索引文件和单词编号文件。

提高：同学们可以爬取中文页面进行分析。中文分词相对复杂。一种比较简单的思路是基于字典和规则的分词方法。其中，字典也叫词库，里面包含大量常用的词语（我们可以直接从网上下载别人整理好的）。我们借助词库并采用最长匹配规则来对文本进行分词。所谓最长匹配，也就是匹配尽可能长的词语。比如要分词的文本是“中国人民解放了”，我们词库中有“中国”“中国人”“中国人民”“中国人民解放军”这几个词，那我们就取最长匹配，也就是“中国人民”划为一个词，而不是把“中国”、“中国人”划为一个词。

3. 索引

索引阶段主要负责将分析阶段产生的临时索引构建成倒排索引（图2）。倒排索引中记录了每个单词以及包含它的网页列表。

图2 倒排索引示例

可先对临时索引文件按照单词编号的大小进行排序。顺序地遍历排好序的临时索引文件，就能将每个单词对应的网页编号列表找出来，然后把它们存储在倒排索引文件中。步骤如图3所示。

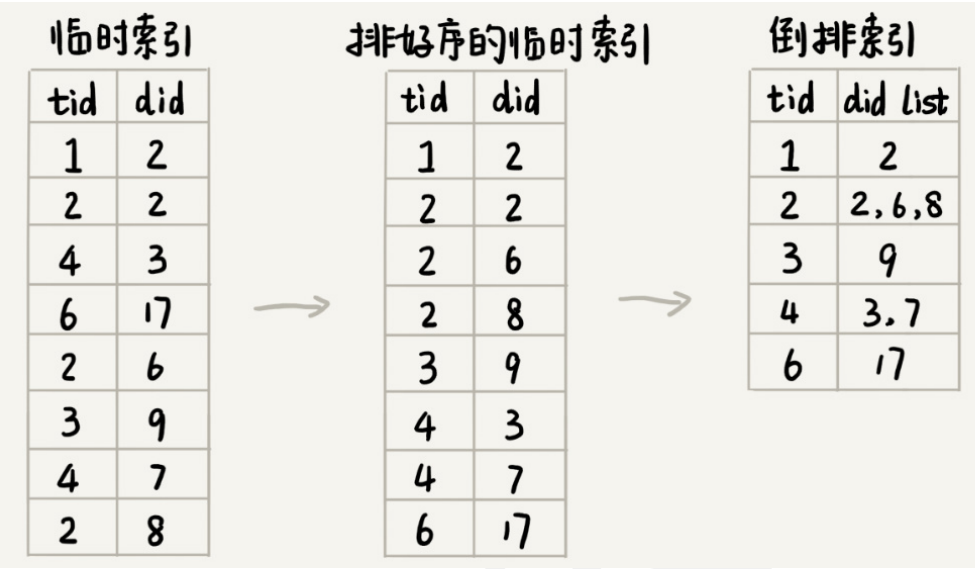

图3 倒排索引创建步骤

提高：现实中的临时索引文件基本都会大到无法全部装入内存，待排序的记录大部分存储在外存储器上。因此可以采用外部排序的方式将临时索引排序并生成倒排索引。

4. 查询

经过前面的步骤，我们得到了页面文件、临时索引文件、单词编号文件和倒排索引文件。

当用户在搜索框中输入某个查询文本的时候，我们先对用户输入的文本进行分词处理。假设分词之后我们得到k个单词。先用这k个单词到单词编号文件中查找对应的单词编号。

用这k个单词编号去倒排索引文件中查找k个单词对应的包含它们的网页编号列表。针对这k个网页编号列表，统计每个网页编号出现的次数。按照出现次数的多少从小到大排序。出现次数越多，说明包含越多的用户查询单词（用户输入的搜索

文本经过分词之后的单词）。经过这一系列查询，我们就得到了一组排好序的网页编号。用这组网页编号去页面文件中查找对应的网页链接，显示给用户。

提高：课设内容中对于搜索引擎的很多优化细节并未涉及，比如计算网页权重的PageRank算法[3]、计算查询结果排名的tf-idf模型[4]等。但我们可以通过一些方法对查询结果排名进行简单优化，例如计算并存储单词在每个页面中出现的次数，查询时考虑查询词在每个页面中出现的次数。在存储了单词出现次数的基础上，还可以考虑基于单词出现的页面数对查询进行优化。

程序设计要求

1. 编程语言不限，推荐使用C++。
2. 程序中涉及排序等算法应自行实现。不可使用STL算法（如sort）或其他与算法主体功能有关的库函数。可使用普通功能函数 (如getline)及STL容器（如vector）。
3. 采用控制台界面。

验收要求

1. 分别展示页面文件、临时索引文件、单词编号文件和倒排索引文件及生成各文件的代码。
2. 各步骤提高部分可选做。列出选做部分的代码及实现结果。
3. 给出生成各个文件所花费的时间（单位：ms）及各个文件的存储空间。如选作优化部分，则需对优化前后所花费时间进行比较。
4. **各组应自行完成，如发现组间雷同或与网上已有代码雷同，则取消成绩！**

[3] https://zh.wikipedia.org/wiki/PageRank
[4] https://zh.wikipedia.org/wiki/Tf-idf

报告撰写要求

1. 采用《计算机学报》模板http://cjc.ict.ac.cn/wltg/new/submit/index.asp。
2. 内容至少包括如下几方面：

1 类、函数列表：将所有类定义和函数名称、参数、功能用表格方式列出。

2 文件格式定义：将各个文件的数据存储方式进行说明。

3 伪代码：分别给出每部分功能的伪代码，并加以文字说明。

4 性能分析：

时间复杂性：给出每个算法的时间复杂性分析过程

空间复杂性：给出每个算法的空间复杂性分析过程

日程安排

- 第一周第一次课后：学生提交分组和分工（在线填写excel模板）。
- 第一周最后一次课：在线会议答疑，组长汇报工作进度。成绩占比20%。
- 第二周最后一次课：在线会议答疑，组长汇报工作进度。成绩占比20%。
- 第三周最后一次课：指导教师评价学生的“验收汇报”，5分钟/组，在线会议，学生汇报、教师提问。成绩占比30%。
- 6月20日（周一）：学生提交报告。成绩占比30%。