

支持通配查询处理的检索系统

知识回顾

通配查询的处理

- mon^* : 找出所有包含以 *mon* 开头的词项的文档
- 如果采用 **B-树词典结构**，那么实现起来非常容易，只需要返回区间 $mon \leq t < moo$ 上的词项 t
- $*mon$: 找出所有包含以 *mon* 结尾的词项的文档
 - 将所有的词项倒转过来，然后基于它们建一棵附加的树
 - 返回区间 $nom \leq t < non$ 上的词项 t
- 也就是说，通过上述数据结构，可能得到满足通配查询的一系列词项，然后返回任一词项的文档

知识回顾

词项中间的 *号处理

- 例子: m^*nchen
- 方法1:
 - 在B-树中分别查找满足 m^* 和 $*nchen$ 的词项集合，然后求交集
 - 这种做法开销很大
- 另外一种方法: 轮排([permuterm](#)) 索引
 - 基本思想: 将每个通配查询旋转，使 $*$ 出现在末尾
 - 将每个旋转后的结果存放在词典中，即B-树中

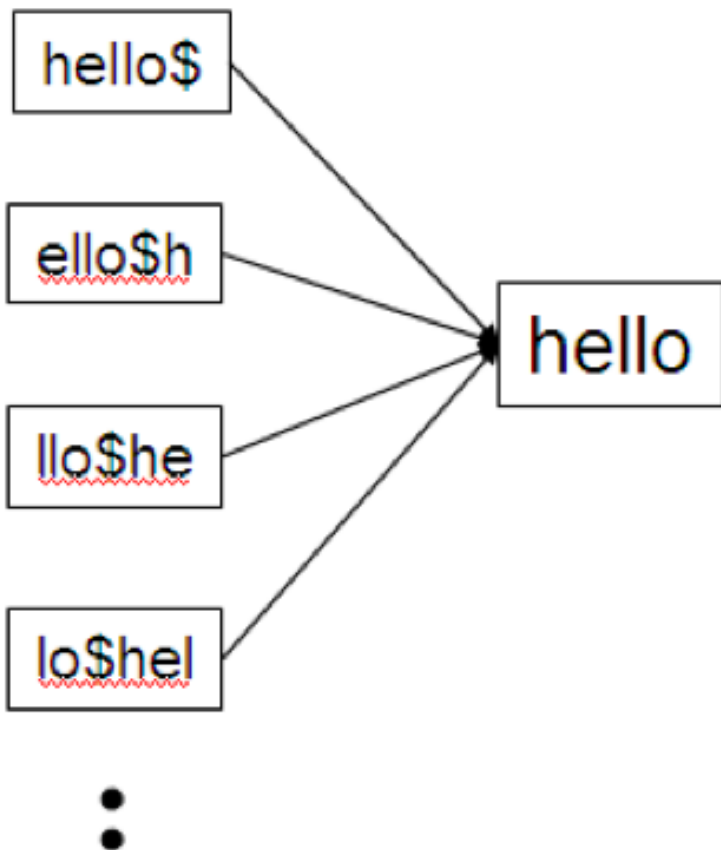
知识回顾

轮排索引

- 对于词项hello: 将 *hello\$*, *ello\$h*, *llo\$he*, *lo\$hel*, 和 *o\$hell* 加入到 B-树中, 其中 \$ 是一个特殊符号
- 即在词项前面再加一层索引

知识回顾

轮排结果 → 词项的映射示意图



知识回顾

轮排索引

- 对于hello, 已经存储了 *hello\$, ello\$h, llo\$he, lo\$hel*, 和 *o\$hell* ***\$hello***
- 查询
 - 对于 X, 查询 X\$
 - **对于 X*, 查询 \$X***
 - 对于 *X, 查询 X\$*
 - 对于 *X*, 查询 X*
 - 对于 X*Y, 查询 Y\$X*
 - 例子: 假定通配查询为 hel*o, 那么相当于要查询o\$hel*
- 轮排索引称为轮排树更恰当
- 但是轮排索引已经使用非常普遍

知识回顾

使用轮排索引的查找过程

- 将查询进行旋转，将通配符旋转到右部
- 同以往一样查找B-树
- 问题：相对于通常的B-树，轮排树的空间要大4倍以上 (经验值)

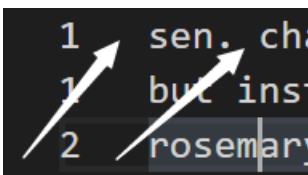
任务1 轮排索引构建

- ① 助教邮箱 1150696014@qq.com 李磊，有任何问题请发邮件咨询
- ② 请编写程序（任意开发语言，推荐python3）为本目录下的1.txt文件构建轮排索引，保存在2_generated.txt(为每个字符串产生其含\$的轮排变换，例如字符串1,100将产生如图的6条轮排变换)
注意，和**倒**排索引一样，字符串的处理顺序也是按字母序，如果你已经完成了倒排索引文件，你可以复用大部分代码。
- ③ 文本处理要求:不要求做词条变化如friends -> friend等;直接用空格作为分割符;都转成小写A->a;符号(例如,)和符号混合字母(例如68-years-old)，空字符串（因split函数产生）等非标准单词均视为单词参与统计，不做特殊处理（即空格分割得到的单个字符串不做进一步处理）;把出现次数排名Top 100的字符串去掉
- ④ 如果你在本目录成功产生了2_generated.txt文件，执行python compare_index.py，将会对比2_generated.txt文件和2_standard.txt文件，如果相同会输出same，代表轮排索引构建正确;如果不同，会输出不同的行，请以2_standard.txt为准调试你的程序

任务1 轮排索引构建 实现思路参考

注意：如下的四步同倒排索引构建是完全一致的，你可以直接复用代码

1.先将1.txt按行读取，每行数据先用\t划分为前后两个部分，前一部分是文档号，后一部分是内容。再将内容用空格分割成数组，顺便转成小写。



The diagram shows a document being split into two parts by a tab character. The first part is the document ID, and the second part is the content. The content is then split into tokens by spaces. The tokens are converted to lowercase and some are underlined to show the process.

```
1 sen. charles e. schumer called on federal safety officials yesterday to reope  
1 but instead there was a funeral , at st. francis de sales roman catholic chur  
2 rosemary antonelle , the daughter of teresa l. antonelle and patrick antonell  
2 one was for st. francis de sales roman catholic church in belle_harbor ; anot  
3 the firefighter , whom a fire department official identified as joseph moore  
3 in st. francis de sales roman catholic church in belle_harbor , queens , the  
4 on nov. 12 , while walking his dog near his home in belle_harbor , queens , h
```

注：符号(例如,)和符号混合字母(例如68-years-old)，空字符串（因split函数产生）等非标准单词均视为单词参与统计，不做特殊处理（即空格分割得到的单个字符串不做进一步处理）

2.对于每个词，实现如下逻辑

if 如果该词已经出现过:

频率加1

记录文档号，某词单篇文章内多次出现只记录一次文档号（即文档号应是一个标记位。而不是追加到数组，追加将导致重复的文档号）

else 没出现过则初始化统计对象:

词频初始化为1，记录本次的文档号

任务1 轮排索引构建 实现思路参考

3.如果你记录词频和文档号的数据结构不是数组，将你的所有统计数据转存到一个数组中，调用编程语言内置的**sort**功能，以词频为键进行排序，然后将前100个高频词去掉。

4.继续调用**sort**,这次以词为键，系统默认会以**ASCII**序排序你的数据

再次提醒上述四步同倒排索引构建是完全一致的，你可以直接复用代码

5.实现轮排函数**rotate(str, n)**，该函数可将第1到n个字符整体移动到n+1到len(str)个字符的右侧。传入不同的n值就可以生成不同的轮排。

6.对于第4步得到的数组，对其中的每个词，再其尾部加上符号\$。然后使用**rotate**函数生成所有轮排变化。按图示的顺序生成，原始字符串为1,100.

```
55  1,100$ 1,100
56  $1,100 1,100
57  0$1,10 1,100
58  00$1,1 1,100
59  100$1, 1,100
60  ,100$1 1,100
```

7.将轮排数据保存在2_generated.txt。如果你在本目录成功产生了2_generated.txt文件，执行python compare_index.py，将会对比2_generated.txt文件和2_standard.txt文件，如果相同会输出**same**，代表轮排索引构建正确;如果不同，会输出不同的行，请以2_standard.txt为准调试你的程序

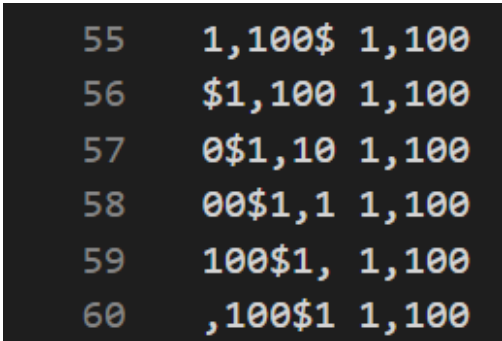
任务2 轮排索引查询

- ① 助教邮箱 1150696014@qq.com 李磊，有任何问题请发邮件咨询
- ② 如果你已经完成了任务1，则你有一个轮排索引保存在保存在 `2_generated.txt`，请编写程序（任意开发语言，推荐python3）加载 `2_generated.txt`和`3.txt`，`3.txt`中包含了通配查询。你还需要加载`1.txt`文件因为最终要输出文档号。
- ③ 通配查询形如`ja*`，它可以匹配的项：`james january`，如果`james`出现在文档1 3 5, `january` 出现在3 5 7。请执行一个类似于倒排索引中的布尔查询（`james or january`），输出文档号以空格分隔1 3 5 7，将`3.txt`中的每条查询输出到`4_generated.txt`中。（查询经过特殊设计，一定会有结果，即`4_generated.txt`中不会有空行且行数和`3.txt`相同）
- ④ 如果你在本目录成功产生了`4_generated.txt`文件，执行`python compare_query.py`，将会对比`4_generated.txt`文件和`4_standard.txt`文件，如果相同会输出`same`，代表查询结果正确;如果不同，会输出不同的行，请以`4_standard.txt`为准调试你的程序
- ⑤ 提示，在本实验中，你会观察到通配符`*`可以匹配0个字符或是多个字符，而不仅是1个字符。

任务2 轮排索引查询 实现思路参考

1.加载2_generated.txt，将其还原为易于使用的形式（能根据含\$号的所有轮排字符串查询文档号数组，即你还需要加载1.txt文件来提供文档号）

。右侧图中6种变体都应该能查询到原始字符串1,100出现的文档号数组



55	1,100\$	1,100
56	\$1,100	1,100
57	0\$1,10	1,100
58	00\$1,1	1,100
59	100\$1,	1,100
60	,100\$1	1,100

2.加载3.txt，3.txt中包含了通配查询。将每条查询使用通配符*进行split分隔。测试数据只会包含一个通配符，所以分隔后得到一个含两个元素的数组parts。实现如下逻辑，得到_query即轮排变化后的查询。

```
if parts[1] == '':
    _query = parts[0]
elif parts[0] == '':
    _query = parts[1] + "$"
elif parts[0] != '' and parts[1] != '':
    _query = parts[1] + "$" + parts[0]
```

任务2 轮排索引查询 实现思路参考

- 3.找到所有以_query为前缀（注意是前缀而不是完全等于，比如a,ab,abc等都是abcdef的前缀）的轮排索引记录，对其文档号数组求并集后输出到4_generated.txt中。（查询经过特殊设计，一定会有结果，即4_generated.txt 中不会有空行且行数和3.txt相同）
- 4.如果你在本目录成功产生了4_generated.txt文件，执行python compare_query.py，将会对比4_generated.txt文件和4_standard.txt文件，如果相同会输出same，代表查询结果正确;如果不同，会输出不同的行，请以4_standard.txt为准调试你的程序

由于排序不稳定等问题，
我会直接看代码实现
Compare脚本用于辅助调
试，不用一定测试通过

提交要求

- 必须有**readme.txt**文件说明如何执行代码和如何进行测试（任务1与任务2请分两个文件编写，代码最好不要通过命令行传参，请将所有逻辑写在代码中，文件也请使用相对路径访问，例如./1.txt）
- 请制作一个**PPT**说明你的解决思路，包含步骤、自定义函数、输出等内容，格式不限
- 评分标准：代码（伪代码）**80%**，文档**20%**
- 将整个文件夹（包含原有数据和你编写的代码）压缩成压缩包
- 压缩文件命名：学号 姓名 倒排索引.zip，在第4周结束前发送到助教邮箱 1150696014@qq.com
李磊