中文分词系统参考实现思路

一 索引结构

考虑使用Hash表或者Trie树作为索引结构。

Hash表的优点在于查找速度非常快，缺点在于依赖散列函数的设计和空间上的浪费。

Trie树的优点在于节省存储空间，缺点在于相比Hash表，查找、插入等的速度较慢。

考虑到第一个字的种类较多，为了加速查找以及简化散列函数的设计，任务书当中描述的使用Hash表加速第一个字的查找的方法是较为优化的方法。

二 索引持久化

Hash表的内部结构其实相当于一个数组，数组是可以直接保存到文件当中的。

这里需要为二级的Trie树建立一个编号，以便读取时将Trie树快速与Hash表中的项关联起来。

接下来需要将Trie树保存到文件中。这类方法在网络上可以找到充分的资料，这里不再详述。

三 词典和索引的转化

不实际在内存中存储词典，而仅存储索引。

将词典导入的操作实际上是对词典中的词依次对索引进行插入操作，而保存词典实际上是遍历整个索引，将每个词顺序写出到文件的过程。

增加、删除和修改词条都可以直接在索引上进行操作。

四 分词算法优化

注意到当输入的结构是 `<M个字><某个标点符号><N个其他字或标点符号><某个标点符号><P个其他字>` 时，正向减字匹配的前N+P次尝试是无效的（因为切分结果中有一个标点符号），同理反向减字匹配的前M+N次尝试也是无效的。

因此，为了减少无效的尝试次数，可以首先使用标点符号对输入进行切分。

其次，为了进一步减少无效的尝试次数，发现当词典的最大词长为M，待匹配块的长度为N>M时，前N-M次匹配是无效的。

利用这个特性，在导入词典、修改、删除词条时需要维护一个“最大词长”的变量。待匹配块的长度将被限制在不大于最大词长的范围内，这样可以保证每一次尝试都是有效的。

注意到在删除词条时如果被删除的词条的长度等于最大词长，需要重新遍历索引得到最大词长，会导致严重的效率损失。

其中一种解决方法是“惰性更新”，也就是这个最大词长不会因为删除词条时的词条操作而改变，而只在重新加载词典或索引时更新。这种方法会导致潜在的分词效率损失。

另外一种解决方法是在Trie树的根结点额外存储“这棵树的深度”，并且随着词条操作而变化。当需要更新最大词长（实际上不是所有的更新都触发，只有*会将最大词长减小的操作之后才会触发*）时，只需要取所有Trie树的深度的最大值即可。这种方法相较重新遍历索引来说要快得多，但是仍然较为缓慢，不过这样没有潜在的分词效率损失。

五 处理英文单词、阿拉伯数字和英文符号

> 假定输入使用 UTF-8 编码

英文单词、阿拉伯数字和英文符号等**在ASCII码范围内的字符**和中文等其他字符在编码上有着显著区别。利用这点，可以完成对英文单词、阿拉伯数字和英文符号的分词。

具体的区别参见附录一或者使用搜索引擎搜索，这里仅简述算法思路：

既然非中文字符（英文单词、阿拉伯数字和英文符号）和中文符号都是**不需要在索引中查找**的“字”，所以可以同时进行处理。具体的方法是使用一个变量记录上一个“字”的类型是中文字符、英文单词、英文符号、阿拉伯数字或中文符号，当正在处理的“字”与上一个“字”的类型不相同时，进行一次切分。

六 被切分字符串的存储结构

考虑到切分操作需要**频繁的**对字符串进行插入操作，所以考虑使用双向链表进行被切分字符串存储。

链表的每一个表项是一个“字”或者一个切分符号。当需要切分时，在相应位置插入切分符号即可。

- **附录一 UTF-8编码下非中文字符与中文字符的区别**

(以下内容摘自 RFC 3629 [传送门]) (这里的“非中文字符”特指英文字母、英文符号和阿拉伯数字)

```
   字符编号范围       |        UTF-8 编码下字节序列
  (十六进制表示)      |       (二进制表示，x为有效位)
--------------------+---------------------------------------------
0000 0000~0000 007F | 0xxxxxxx
0000 0080~0000 07FF | 110xxxxx 10xxxxxx
0000 0800~0000 FFFF | 1110xxxx 10xxxxxx 10xxxxxx
0001 0000~0010 FFFF | 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx
```

也就是说，ASCII码范围内，仅需一个字节即可表示的*非中文字符*的二进制表示以`0`开头，并且仅持续一个字节长度。中文字符等*需要不止一个字节来表示*的字符（实际上，中文一般使用三个字节）的二进制表示以`110,1110`或`11110`开头，并且持续`2,3或4`个字节长度。

通过这一特点，可完成中文字符和非中文字符的区分。

> 如果这个参考思路有问题欢迎在群里提出或私聊找我w（