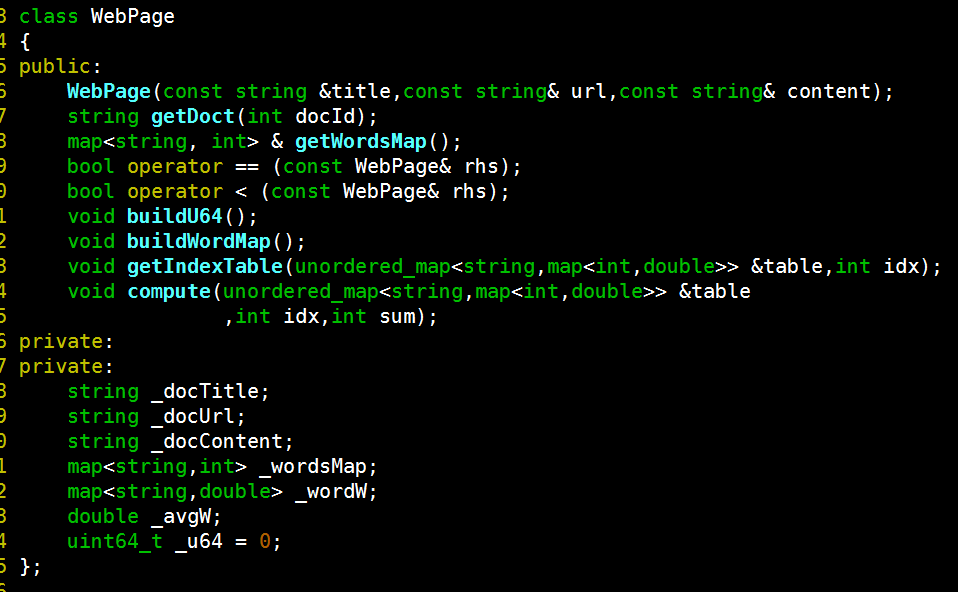
搜索引擎

1. 离线部分



1. 建立网页库

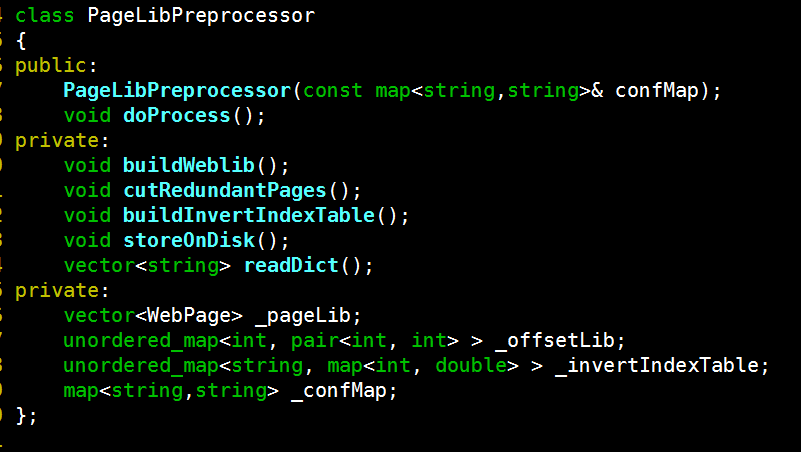
先读取网页库资料文件夹，全部读取

用tinyxml2提取到网页库中

1. 去重

用simhush得到每个网页库中的u64值，自定义<和=运算符

对Vector<webpage>先排序再unique再erase；

1. 

建立倒排索引

统计每个网页的各个单词的词频\_wordsMap;

对每个网页的每个单词填充倒排索引表map的pair为<docid，0>；🡪可以到总的每个单词的词频了（也就是map中该单词对应的map的size）

对每个网页中的每个单词可以计算W值了

然后计算得到每个网页的W’

再用除法得到每个网页中每个单词的真正W值，再填充入倒排索引表中🡪更新<docid,w值>此前都是0暂时使用的。

4 最后再把网页库，偏移库，倒排索引库写入文件中。

计算W时log后要整成double的，不然有的值直接为0了。

二、在线部分

1. 搭建服务端框架

与以前区别不大，有两个坑。

第一个：客户端连接服务器会回个welcome过去，删了

第二个：v4的epoll里面对于断开连接的判断只有ret==0，要把-1也加上

还有就是MyTask类换成了WORDquery，都封在了回调服务器里

1. 提供网页查询服务

计算查询词的W，得到一个对应每个单词vector<string> 的对应W值vector<double>作为基础

然后我的思路是

对于vector中的每个单词去倒排索引表中去找，会得到单词个数个SET集合，然后对这些集合求交集得到包含所有单词的网页的docid

对每个docId得到对每个单词的W值即vector<vector<double>>

然后抄源码上的比较的sturct对这个vector进行排序就好

又一个坑不要用sort，会崩溃（不知道为啥），要用个statyle\_sort。

对关联文章的排序就有了，然后把结果封成Json字符串，关键字为“files”,值又是一个Json数组，填好就行，发送时前面加个字符串型的长度和换行就好了

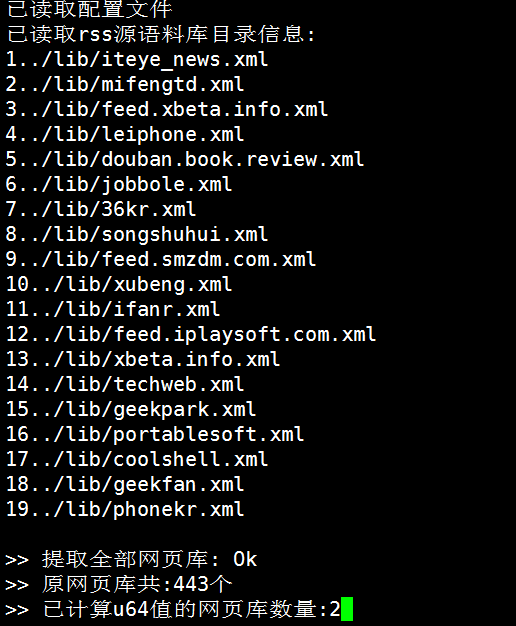
对于缓存直接用redis很简单和上次一样

对于mysql没空写了，接口忘了还得看，没写

写的最捞的。网页中根据查询词得到摘要时，我就把包含关键字的每150个字节的句子都拼在了一起，由于是中文，所以会出现很多乱七八糟的符号

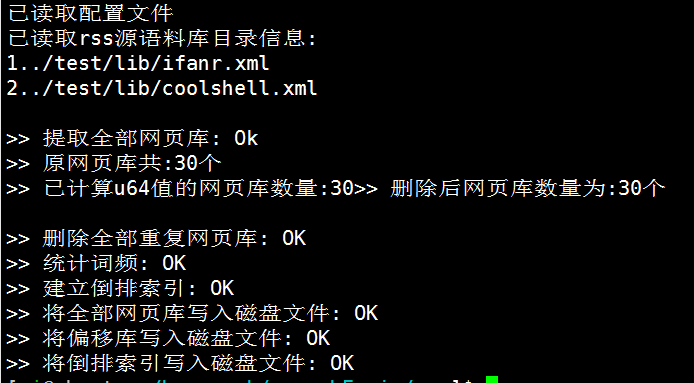
最终结果

离线



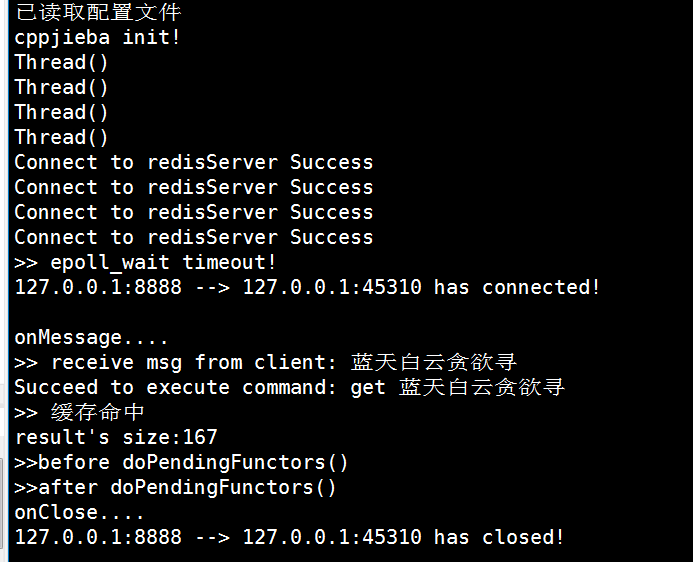
分词特别慢 至少半小时

只有两个文件的测试



我用simhush去重后一共还有399网页。

在线服务器





前端

