第四周报告

1. 实验概览

本实验旨在理解建立索引的基本步骤（分析、索引、统计词频），理解索引查询的基本原理和步骤（语义理解并语言处理，查找索引，相关性排序）；基于 Luence 库和 jieba 中文分词库，实现中文网页索引的创建和搜索。

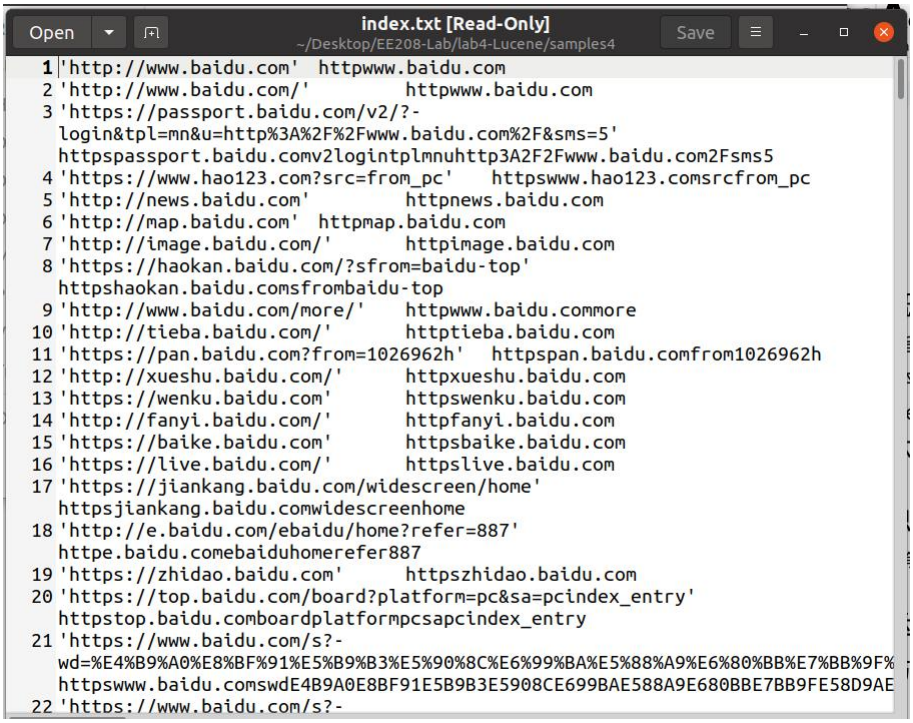
2. 实验环境

Docker: sjtumic/ee208

3. 解决思路

创建索引：

利用第三次实验中的并发爬虫爬取 10k 个网页（图为 index.txt 截图）



创建 Field，分别用于无词频统计内容（文件名等）和含词频统计和位置记录的内容（网页正文内容）

```
t1 = FieldType()
t1.setStored(True)
t1.setTokenized(False)
t1.setIndexOptions(IndexOptions.NONE)

t2 = FieldType()
t2.setStored(False)
t2.setTokenized(True)
t2.setIndexOptions(IndexOptions.DOCS_AND_FREQS_AND_POSITIONS)
```

读取 index 文件，查找每个文件名；利用 Beautifulsoup 解析网页内容，获得网页的标题和编码方法，利用编码方法重编码 contents

```
soup = BeautifulSoup(contents, 'html.parser')
title = soup.find("head").find("title").string
encoding = [i.get('charset') for i in soup.find("head").findAll('meta') if i.get('charset') != None]
if not len(encoding):
    encoding = 'utf-8'
else:
    encoding = encoding[0]
if encoding.upper() != "UTF-8":
    contents = contents.encode('GBK')
```

利用正则表达式去除 contents 中所有的汉字以外的字符

```
contents = re.sub("[^\u4e00-\u9fa5]", "", contents)
```

创建 Document 实例，添加相关内容；其中，利用 jieba 库实现中文分词并中文隔开，利用 WhitespaceAnalyzer 作为分析器根据空格分词，实现索引的建立

```
    doc = Document()
    doc.add(Field("name", filename, t1))
    doc.add(Field("path", path, t1))
    doc.add(Field("title", title, t1))
    doc.add(Field("url", urls[i], t1))
    if len(contents) > 0:
        contents = jieba.cut(contents, cut_all=False)
        contents = ' '.join(contents)
        doc.add(Field("contents", contents, t2))
    else:
        print("warning: no content in %s" % filename)
    writer.addDocument(doc)
except Exception as e:
    continue
```

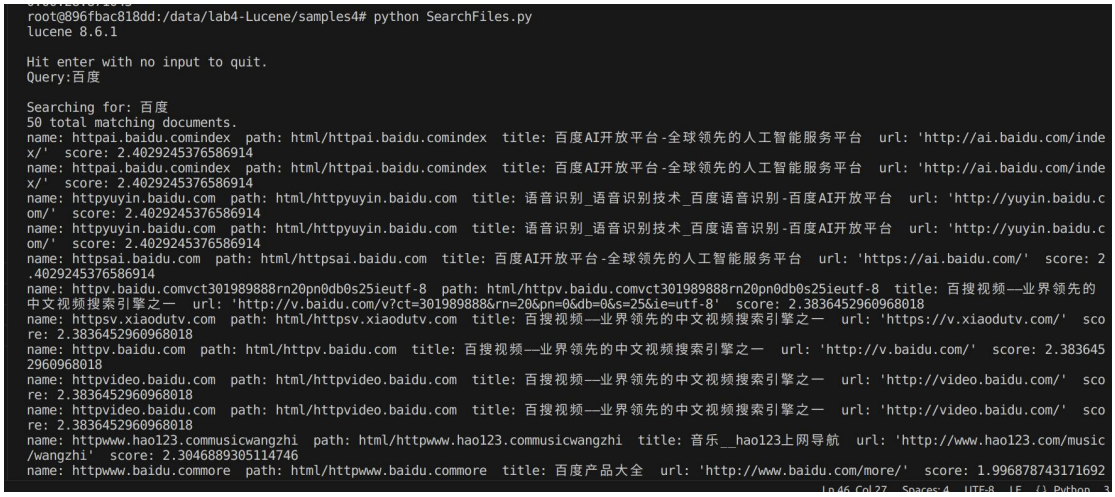
查找索引：

直接基于 lucene 库中的 QueryParser 对查询命令进行解析并调取索引查询

```
print()
print ("Searching for:", command)
query = QueryParser("contents", analyzer).parse(command)
scoreDocs = searcher.search(query, 50).scoreDocs
print ("%s total matching documents." % len(scoreDocs))
```

4. 代码运行结果

以百度为根节点爬取 10000 个网页，索引结果如下



5. 分析与思考

Lucene 的实现：

存储：基于 FST 字典格式

查询：在 lucene 中查询是基于 segment。每个 segment 可以看做是一个独立的 subindex，在建立索引的过程中，lucene 会不断的 flush 内存中的数据持久化形成新的 segment。每个 segment 是不可变的，只有在多个 segment 也会不断的被合成时才被真正删除。