

# “统计电影数目”说明文档

## 数据仓库第二次项目

刘林青 1354361

---

### 1.调用说明:

- `get_id.py`: 输入原始`movie.txt`, 输出所有不同的`product_id`存到名为`list`的文件中, 并且将生成的`list`复制到远程机器, 登陆远程机器
- `core.py`: 由生成的`list`抓取数据存入数据库中
- `crawl.sh`: 抓取url的脚本文件, 由`core.py`调用, 无需单独执行。

**注:** 所有代码都在Linux环境执行 (最好Debian/Ubuntu发行版, python 2.7)

执行步骤(假定测试文件`movie.txt` 在`/tmp/movie.txt`路径)

1 `cd 1354361_liulingqing_homework2`

2 `python get_id.py -i /tmp/movie.txt`

接下来生成`list`之后会提示复制到远程机器, 选择`yes`并且输入密码`tongji`

最后会自动登录远程机器, 密码`tongji`

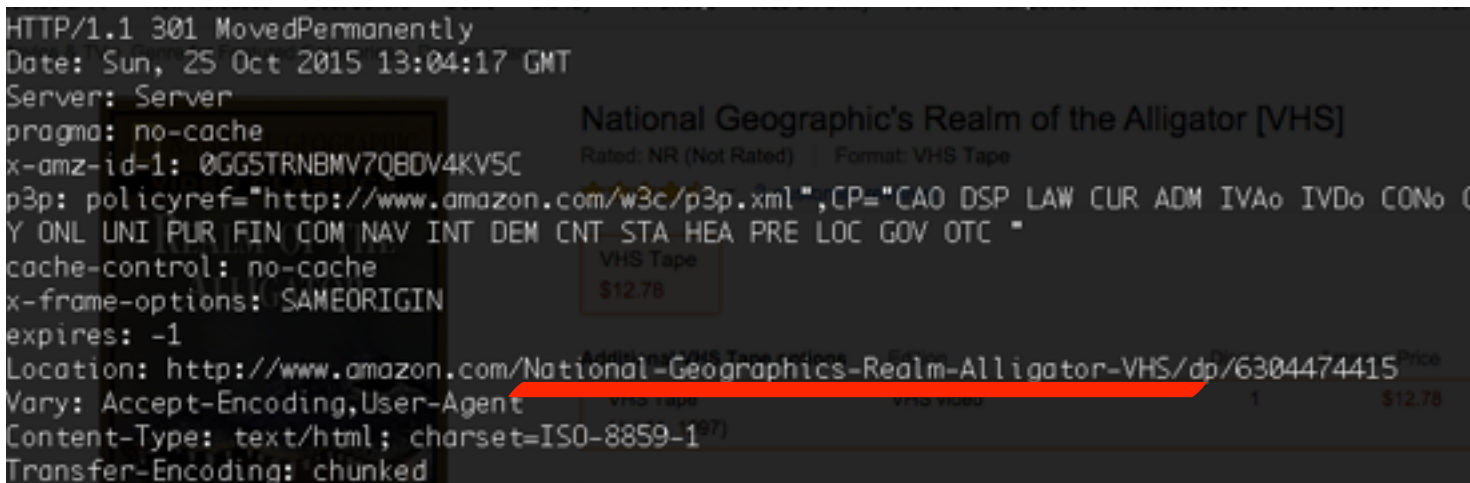
3 在远程机器中, `cd 1354361_liulingqing_homework2`

4 执行 `time python core.py`

执行过程中会显示进度以及时间, 执行完毕会在屏幕打印电影的总数以及具体执行时间(real time)

### 2.流程说明:

- 从输入文件`movie.txt` 中提取无重复的`product_id` 并输出到文件
- 从文件中提取`product_id` 并生成网址url
- 网址输入后会重定向到另一个包含电影名的url,如图中的location所示



```
HTTP/1.1 301 Moved Permanently
Date: Sun, 25 Oct 2015 13:04:17 GMT
Server: Server
pragma: no-cache
x-amz-id-1: 0GGSTRNBMV7QBDV4KV5C
p3p: policyref="http://www.amazon.com/w3c/p3p.xml";CP="CAO DSP LAW CUR ADM IVAo IVDo CONo C
Y ONL UNI PUR FIN COM NAV INT DEM CNT STA HEA PRE LOC GOV OTC "
cache-control: no-cache
x-frame-options: SAMEORIGIN
expires: -1
Location: http://www.amazon.com/National-Geographics-Realm-Alligator-VHS/dp/6304474415
Vary: Accept-Encoding,User-Agent
Content-Type: text/html; charset=ISO-8859-1
Transfer-Encoding: chunked
```

所以只需要从网页的header当中提取信息即可(通过`curl -i`命令调用), 不需要完全下载完整的网页再解析 (第一次作业中的代码是下载的全部网页)

- 建立数据库，并且建3个表：

表名	movie_name	503_err	404_err
列名	id	id	id
列名	info	err_info	err_info

movie\_name（保存电影的ID和名称）

503\_err（保存由于503错误未能正常输出header的网址对应的product\_id）

404\_err（已被亚马逊官方取消的网页对应的product\_id）

- 重新提取503\_err中id对应的网址，每次访问完清空该表，并将重新访问再次发生503\_err的id重新存入该表。循环访问直至该表内id数少于3.

### 3. 细节处理

#### 1>通过捕捉HTTP 状态码判断爬取网页的状态

##### **301:**

被请求的资源已永久移动到新位置，header爬取成功，将提取的信息直接输入数据库表。

##### **404:**

请求失败，请求所希望得到的资源未被在服务器上发现，该网址已被亚马逊官方清楚，直接把id和错误类型输入表中。

##### **503:**

由于临时的服务器维护或者过载，服务器当前无法处理请求。

这个状况是临时的，并且将在一段时间以后恢复。捕捉到该信息后，把对应id和错误信息输入表中。

当所有网址都处理完之后，再次处理503信息，每次爬取一次就清空该表，再把新出现的错误id存入。循环处理直至该类id数量小于3.

#### 2>多线程处理任务：

使用python threading库，建立线程池，因为测试环境为1G内存单核机器，所以在多次试验之后选择140个并发。但是由于亚马逊官方限制，单个ip并发抓取数据会返回大量503错误，所以利用3个socks5代理和本机一起进行并发抓取数据，每次并发线程开启时，随机选择一个代理使用。从而实现1万数据在4-7分钟内抓取完毕，具体时间由服务器所在San Francisco 地区网络繁忙程度决定。

**注1：**因为助教作业评估时间不确定，并且具有多种不确定因素以及安全性稳定性考虑，所以没有写脚本自动搜寻互联网可用免费代理。抓取数据仅仅使用3个私人代理进行

**注2：**因为众多端口以及各种库依赖配置问题，所以在本地处理完数据后，抓取以及存储数据均远程进行。

### 4.测试机器配置

单核1G内存DigitalOcean VPS

系统 Ununtu LTS 14.04 Python版本2.7.6

### 5.测试结果

```
root@CVPR:~/b1354361_liulingqing# time python core.py
=====
There are 237 diffent movies in total! You can view more details in the database
MySQL user: root passwd: root
All the movie related data are stored in DATABASE amazon.
TABLE movie_name stores all the name. TABLE 404_err and 503_err restore few invaild id.
=====

real    0m7.600s
user    0m1.855s
sys     0m3.160s
```