HADOOP 数据分析实验

姓名：张聪海

学号：21851193

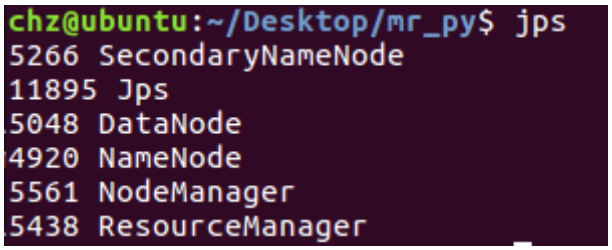
班级：软件工程 1801

目录

一、Hadoop 平台搭建

官网下载安装包版本是 2.9.1，集群环境为 centos7，配置实验室固定 ip 建立集群，实际使用的时候发现局域网的网关性能原因等限制，实际的运行效率较差，运行性能比较差，对此 Hadoop 提供了伪分布方式进行模拟。具体的模拟方式为：

这种模式也是在一台单机上运行，但用不同的 Java 进程模仿分布式运行中的各类结点，例如本次可以看到启动了一个 NameNode、一个 SecondaryNameNode、一个 DataNode、一个 NodeManager、一个 ResourceManager。



图：当前 java 节点运行项目

1.1 Hadoop 上传数据文件

搜狗数据集中数据的当前结构如下：

时间戳 T	用户访问时间
uid	来访用户的网络标示
keyword	用户查询词
rank	在返回结果中的位置
ordered	用户点击的顺序号
url	用户点击的

首先需要对数据进行预处理，使用 bash 脚本进行数据预处理，首先需要将时间戳字符串拆分为年月日的分拆字符串。

Bash 脚本如下：

扩展字段：

```
infile=$1
outfile=$2
awk -F '\t' '{print $0"\t"substr($1,0,5)"\t"substr($1,5,2)"\t"substr($1,7,2)"\t"substr($1,9,2)}' $infile > $outfile
```

扩展字段脚本的内容是将根据 tab 分段之后的数据拆分之后拼接到字符串末尾

其次数据中存在较多的 NULL 位置，编写一个删除空串的脚本进行删除操作

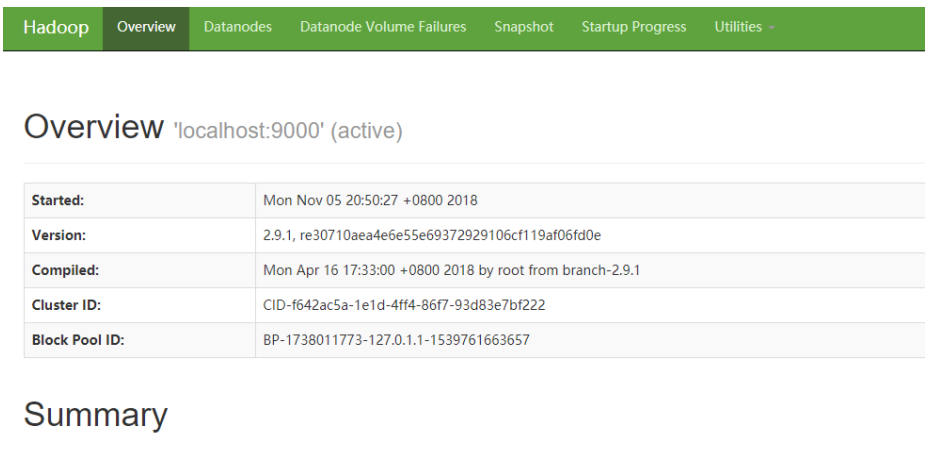
```
infile=$1
outfile=$2
awk -F "\t" '{if($2 != "" && $3 != "" && $2 != " " && $3 != " ") print $0}' $infile > $outfile
```

通过对分类后的关键字的统计分析，我们可以得出用户的兴趣类和搜索需求，进而我们可以对网站进行优化，对于用户搜索较多的内容给予足够的重视，提供优化的服务。

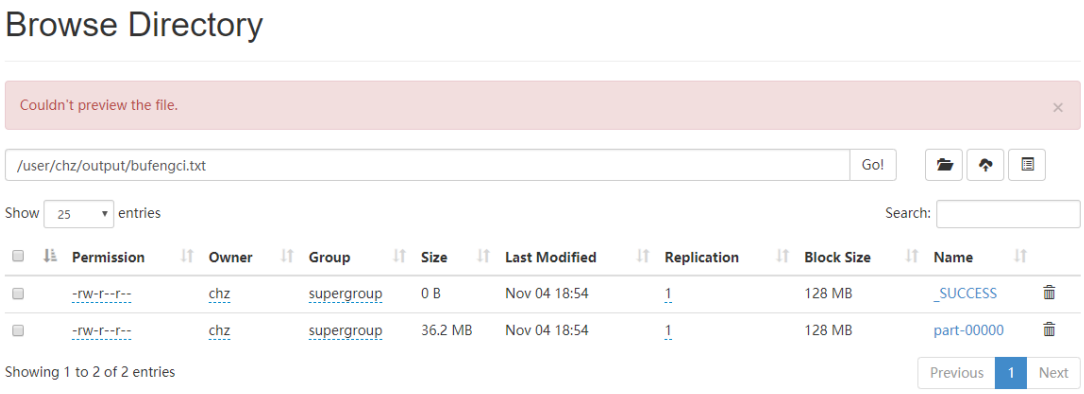
```
hdfs dfs -mkdir -p /sogou/20111230
```

1.2 通过节点管理页面查询

Hadoop 提供网页上的数据查询管理操作，默认端口为 50070，其次对作业也有一个管理端口，默认端口为 8088，通过访问这两个端口，能够查询反馈系统集群状态以及任务进度，日志记录等。



节点管理界面



由于 linux 系统默认安装 python2.7 环境，因此集群中所有的节点都可以运行 python 程序，因此可以借助 hadoop 自带的 mapreduce 中的 streaming 流，将编写的 mapper 以及 reducer 脚本应用。

二、MapReduce 过程

2.1 Mapper 过程

```
#!/usr/bin/env python
# coding=utf-8
import sys,os
import jieba

all_line=[]
for line in sys.stdin:  # 遍历读入数据的每一行
    line = line.replace(' ','') # 将多余的空格去除
    line = line.replace('.','') # 将多余的符号去除
    words = line.split("\t")    # 按tab将句子分割成单个单词
    all_line.append(words[2])
all_line.sort() # 进行内部排序
for line in all_line:
    print ('%s\t%s' %(line, 1)) # 流式输出
```

图：mapper 过程

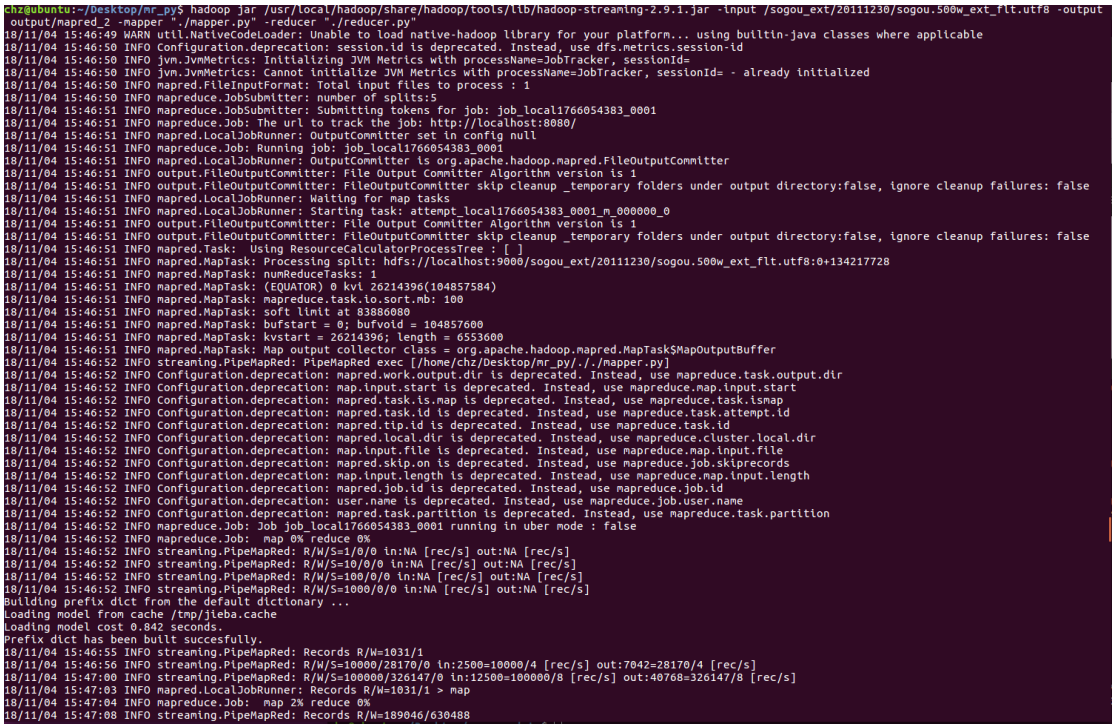
Mapper 采用流式处理，直接输入输出标准输入输出流作为中间传递内容，此处直接传递为 keyword 和次数，直接在 py 脚本调用内置排序，为下一步的 Reducer 提供输入。

2.2 Reducer 过程

```python
#!/usr/bin/env python
# coding=utf-8
from operator import itemgetter
import sys

current_word = None
current_count = 0
word = None

# 输入必须经过预先排序
for line in sys.stdin:
    word, count = line.split('\t', 1)
    try:
        count = int(count)
    except ValueError:
        continue
    if current_word == word:
        current_count += count
    else:
        if current_word:
            print "%s\t%s" % (current_word, current_count)
        current_count = count
        current_word = word

if word == current_word:    # 流式输出（无序）
    print "%s\t%s" % (current_word, current_count)
```

图：Reducer 过程

第一阶段是把输入文件按照一定的标准分片(InputSplit)，每个输入片的大小是固定的。默认情况下，输入片(InputSplit)的大小与数据块(Block)的大小是相同的。此处配置的数据块(Block)的大小是默认值 128MB，可以看到默认的 JobSubmitter 将作业 Split 为 5 份，然后提交这个作业给 application manager，之后 Job 类接收到作业之后，会开启一个作业追踪的链接，并同步更新页面中的状态。之后 Hadoop 会确认资源的输入输出流，之后按顺序对 split 的文件进行处理，处理中会调用 streaming 类中的 PipeMapReduce 类驱动.py 文件。



图：分配 Job 过程

接下来是开始执行 reduce 过程，执行过程中由于使用 log4j 的原因，会在不同的节点中显示日志 log 记录下当前的操作以及文件系统的独写报告情况，可以通过统一的资源平台进行管理和调用查看。

图：开始执行 map 过程

```
18/11/04 17:44:09 INFO streaming.PipeMapRed: mapRedFinished
18/11/04 17:44:10 INFO mapred.Task: Task:attempt_local1128249237_0001_r_000000_0 is done. And is in the process of committing
18/11/04 17:44:10 INFO mapred.LocalJobRunner: Records R/W=13142021/304526 > reduce
18/11/04 17:44:10 INFO mapred.Task: Task attempt_local1128249237_0001_r_000000_0 is allowed to commit now
18/11/04 17:44:10 INFO output.FileOutputCommitter: Saved output of task 'attempt_local1128249237_0001_r_000000_0' to hdfs://loc
18/11/04 17:44:10 INFO mapred.LocalJobRunner: Records R/W=13142021/304526 > reduce
18/11/04 17:44:10 INFO mapred.Task: Task 'attempt_local1128249237_0001_r_000000_0' done.
18/11/04 17:44:10 INFO mapred.LocalJobRunner: Finishing task: attempt_local1128249237_0001_r_000000_0
18/11/04 17:44:10 INFO mapred.LocalJobRunner: reduce task executor complete.
18/11/04 17:44:10 INFO mapreduce.Job:  map 100% reduce 100%
18/11/04 17:44:12 INFO mapreduce.Job: Job job_local1128249237_0001 completed successfully
18/11/04 17:44:12 INFO mapreduce.Job: Counters: 35
        File System Counters
                FILE: Number of bytes read=1051597551
                FILE: Number of bytes written=1617979493
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=2629558568
                HDFS: Number of bytes written=4183521
                HDFS: Number of read operations=61
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=8
        Map-Reduce Framework
                Map input records=4999877
                Map output records=16866593
                Map output bytes=148833786
                Map output materialized bytes=182567031
                Input split bytes=580
                Combine input records=0
                Combine output records=0
                Reduce input groups=376729
                Reduce shuffle bytes=182567031
                Reduce input records=16866593
                Reduce output records=376729
                Spilled Records=47805517
                Shuffled Maps =5
                Failed Shuffles=0
                Merged Map outputs=5
                GC time elapsed (ms)=624
                Total committed heap usage (bytes)=1233453056
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=643670164
        File Output Format Counters
                Bytes Written=4183521
18/11/04 17:44:12 INFO streaming.StreamJob: Output directory: output/mapred_4
```

图：Reduce 结束

Reduce 运行结束会输出运行结果如 complete successfully，之后 hadoop 会输出统计报告，分别是统计文件系统的计数器如总读写的的字节数以及独写的操作数、mapreduce 框架的输入条目数，输出条目数，split 块数，输入的总块数以及 shuffle 总块数等详细参数信息。

运行结束之后可以得到不使用分词的方式得到的数据如下：

```
chz@ubuntu:~/Desktop/mr_py$ head -200 sorted.log
百度    38447
baidu   18366
人体艺术        14502
4399小游戏      11472
qq空间  10359
优酷    10162
新亮剑  9666
馆陶县县长闫宁的父亲    9127
公安卖萌        8192
百度一下你就知道        7810
百度一下        7207
4399    7057
魏特琳  6665
qq网名  6173
7k7k小游戏      6018
黑狐    5614
儿子与母亲不正当关系    5496
新浪微博        5384
李宇春体        5310
新疆暴徒被击毙图片      4997
hao123  4927
123     4841
4399洛克王国    4119
qq头像  4093
nba     4027
龙门飞甲        3927
qq个性签名      3887
```

观察可以得到，“baidu”，“百度”，“百度一下你就知道”，“百度一下”均指向百度，因此可以视作近义词，进行合并操作，同时注意到 4399 小游戏，4399 洛克王国、7k7k 小游戏以及 4399 在当时具有极高的搜索量，结合 11 年左右的童年经历可以得到一致的结论。同时结合生活经验，一般人不会在浏览器的搜索栏内输入完整的输入馆诸如“陶县县长闫宁的父亲”之类的搜索内容，因此可以得出当时的搜狗搜索已经具有联想输入的功能，能够在输入前半段之后自动联想后半段的内容。

同时观察可以得到主要的搜索内容集中在娱乐，功能类的搜索上，一些花边新闻诸如公安卖萌、馆陶县县长、魏特林、两次处决美女罪犯、新疆暴徒被击毙等占据了前排，另外如新浪微博、李宇春体、人体艺术、小说电影等娱乐类，具有明显的性暗示的内容同样也跻身前列，

由于长段的输入中虽然意思不同，但是关键词相同，对于这样的输入情况，采用开源的 python 工具 jieba（中文名：结巴）进行简单分词，分词之后可以将例如： 为什么电灯会发光、电灯会发光吗、电灯为什么会发光 这样的三句表述不同的句子中的电灯作为关键词截取出来，最终统计得到的结果为电灯，进一步可以联想电灯在当时的实时环境中为何会被多次提及，是由于全国范围的大停电导致人们对电灯引发的思考，抑或是当时的电灯生产厂商爆出了

大新闻。接下来采用简单的分词方法对搜索的关键词做分割统计，统计结果如下：

1	的	402597	
2	下载	213277	
3	全集	98640	
4	qq	89561	
5	电影	85629	
6	视频	71862	
7	百度	65257	
8	2011	64989	
9	中国	60133	
10	图片	58572	
11	网	56270	
12	年	55758	
13	是	55117	
14	2012	54383	
15	/	50725	
16	什么	50485	
17	我	50057	
18	在线	49470	
19	查询	48499	
20	大全	46561	
21	免费	45553	
22	电视剧	42642	
23	-	41985	
24	与	41723	
25	怎么	40338	
26	最新	40277	
27	手机	38754	
28	官网	37680	
29	2	35098	
30	小游戏	34739	
31	观看	34702	

1	下载	213277
2	全集	98640
3	qq	89561
4	电影	85629
5	百度	84249
6	视频	71862
7	2011	64989
8	中国	60133
9	图片	58572
10	2012	54383
11	在线	49470
12	查询	48499
13	大全	46561
14	免费	45553
15	电视剧	42642
16	最新	40277
17	手机	38754
18	官网	37680
19	小游戏	34739
20	观看	34702
21	总结	31442
22	工作	30979
23	网上	30538
24	4399	30290
25	txt	29910
26	游戏	29614
27	高清	29157
28	空间	28932
29	小说	27625
30	cf	26491

左图是经过过滤处理之前，右边是过滤后

经过简单分词之后的数据输出如下，可以明显的看到由于分词的原因导致统计结果数据出现严重问题，的字占据了最主要的搜索记录，总计 500 万的数据中，的字出现的次数有 40 万次，其次是下载以及全集，这也是不完全的记录统计，应当为“xx 下载”或者“xx 全集”的形式，从此向下的电影以及视频等也是一样的道理，即便分词出现问题，也可以从中观察得到 11 年时互联网逐渐兴起，当时的上网者对于当时的较为稀缺的文化产业需求比较旺盛，时值 2011 年末，玛雅人对于 2012 的世界末日的预言导致 2012 的搜索相对靠前。

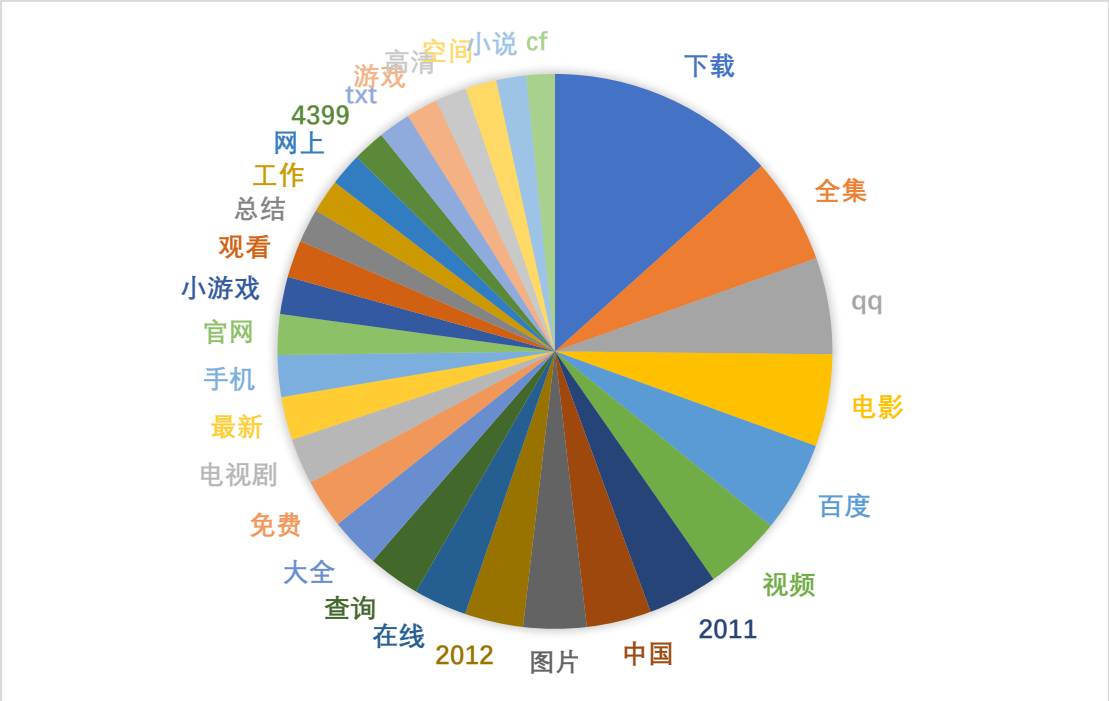
观察排行结果能看到当时正在兴起的即时聊天工具 QQ 占据了前五，同时有很多人在搜狗上搜索百度，可以看出当时的百度已经兴起并且占据了一定的稳定用户群体，用户开始表现出了一定的依赖性，同时存在由于搜狗搜索系统设计不完善导致用户体验度差，用户流量转入百度，另一个问题是分词的结果中

有较多的无明显意义单字，可以考虑去除。

综上总共考虑两种方式对数据进行过滤，第一是增加单字的过滤

第二是考虑通过字典的方式去除典型的无意义字符如“ ”“abc”等

第三点可以实现的是使用近义词替换，通过创建手动的近义词词典，完成近义词的替换。最终可以得到图形化的结果如下：



同理对统计 URL 最终的执行结果如下：

```
chz@ubuntu:~/Desktop/mr_py$ head -n 30 url.txt
http://www.baidu.com/    73732
http://www.4399.com/     19015
http://www.hao123.com/   14338
http://www.youku.com/    14085
http://qzone.qq.com/     12920
http://www.7k7k.com/     8326
http://weibo.com/        7547
http://cf.qq.com/        7544
http://www.xixiwg.com/   7043
http://www.12306.cn/     6961
http://dnf.qq.com/       6835
http://bbs1.people.com.cn/postDetail.do?id=112546724    6325
http://www.a67.com/      6048
```

对于 URL 可以看到，第一名百度远远超过第二名，说明当时的百度以及开始成长，第二 4399 也可以从搜索关键字中得到验证，当时刚刚新办的 12306 也跻身进入前十，由于此时的搜索记录不完善，注意到其中有一个“www.xixiwg.com”此网站是一个游戏外挂网站，竟然也跻身进入前十着实让人意外，但游戏往往是一阵火一阵凉，在全年的搜索记录中似乎并没有看到这个网站的影子，现该网站现在已经关闭多年。

虽然URL分析比搜索词分析要简单，但是由于URL存在多种不同的后缀以及子链接，单纯的统计单页面的访问次数具有局限性，可以设置使用通配符进行过滤，统计各个公司的访问量。

三、使用 hive 进行数据分析

3.1 基础环境

3-1-1、建立 hive 表

```
hive> set hive.exec.dynamic.partition.mode=nonstrict;
hive> CREATE EXTERNAL TABLE sogou.sogou_ext_20111230(
    >  ts STRING,
    >  uid STRING,
    >  keyword STRING,
    >  rank INT,
    >  ordered INT,
    >  url STRING,
    >  year INT,
    >  month INT,
    >  day INT,
    >  hour INT
    >  )
    >  COMMENT 'This is the sogou search data of extend data'
    >  ROW FORMAT DELIMITED
    >  FIELDS TERMINATED BY '\t'
    >  STORED AS TEXTFILE
    >  LOCATION '/sogou_ext/20111230';
OK
Time taken: 0.292 seconds
```

使用过滤过的数据对 hive 表格进行初始化，hive 表中指定文件位置以及分割符号“\t”能够自动的将文件映射到表中，修改文件系统中文件内容会实时的影响数据库中条目数。

3-1-2、使用 hive 查询数据

```
2018-11-02 18:05:58,552 Stage-2 map = 0%,  reduce = 0%
2018-11-02 18:06:00,555 Stage-2 map = 100%,  reduce = 100%
Ended Job = job_local1922809182_0010
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 33856270666 HDFS Write: 6883896324 SUCCESS
Stage-Stage-2:  HDFS Read: 11446093350 HDFS Write: 2294632108 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
百度    38437
baidu   18312
人体艺术        14474
4399小游戏      11438
qq空间  10316
优酷    10158
新亮剑  9654
馆陶县县长闫宁的父亲    9127
公安卖萌        8192
百度一下 你就知道       7504
百度一下        7104
4399    7041
魏特琳  6665
qq网名  6148
7k7k小游戏      5985
黑狐    5610
儿子与母亲不正当关系    5496
新浪微博        5369
李宇春体        5310
```

使用 hive 进行上一问的查询，结果和使用自定义的 mapreduce 实现结果一致

```
http://www.qqwangming.org/    6003
http://www.a67.com/     6048
http://bbs1.people.com.cn/postDetail.do?id=112546724    6325
http://dnf.qq.com/      6835
http://www.12306.cn/    6961
http://www.xixiwg.com/  7043
http://cf.qq.com/       7544
http://weibo.com/       7547
http://www.7k7k.com/    8326
http://qzone.qq.com/    12920
http://www.youku.com/   14085
http://www.hao123.com/  14338
http://www.4399.com/    19015
http://www.baidu.com/   73732
```

3.2 数据处理

数据处理采用 python 以及正则表达式匹配，基本只需要处理空值、异常内容

3.3 展示数据

展示数据采用饼图的形式，采用常用的 matplotlib，python 的总体的展现效果不强，因此采用 excel 来辅助实现可视化，整体效果明显强于 python 的几种可视化方法。

3.4 数据分析

3.4.1 基本数据特征

总条目数：

```
2018-11-05 16:55:39,133 Stage-1 map = 0%,  reduce = 0%
2018-11-05 16:55:44,139 Stage-1 map = 100%,  reduce = 0%
2018-11-05 16:55:58,818 Stage-1 map = 67%,  reduce = 0%
2018-11-05 16:56:00,821 Stage-1 map = 100%,  reduce = 0%
2018-11-05 16:56:01,823 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local15863853_0003
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 4667351928 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
4999877
Time taken: 24.832 seconds, Fetched: 1 row(s)
```

经过 hive 查询返回非空数据总条目数是 4999877 条

```
Ended Job = job_local1081318862_0002
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 3380011600 HDFS Write: 0 SUCCESS
Stage-Stage-2:  HDFS Read: 1287340328 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
4999149
Time taken: 100.381 seconds, Fetched: 1 row(s)
```

可以得到无重复总条数为 4999149 条

```
Ended Job = job_local1600159134_0003
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 4667351928 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
1352645
Time taken: 27.293 seconds, Fetched: 1 row(s)
```

可以得到独立 UID 总数为 1352645 条

3.4.2 关键词分析

使用 hive 实现关键词词频排序，输出结果与之前的 MapReduce 一致，但是不能直接实现分词，需要将数据条目输出之后才能处理。

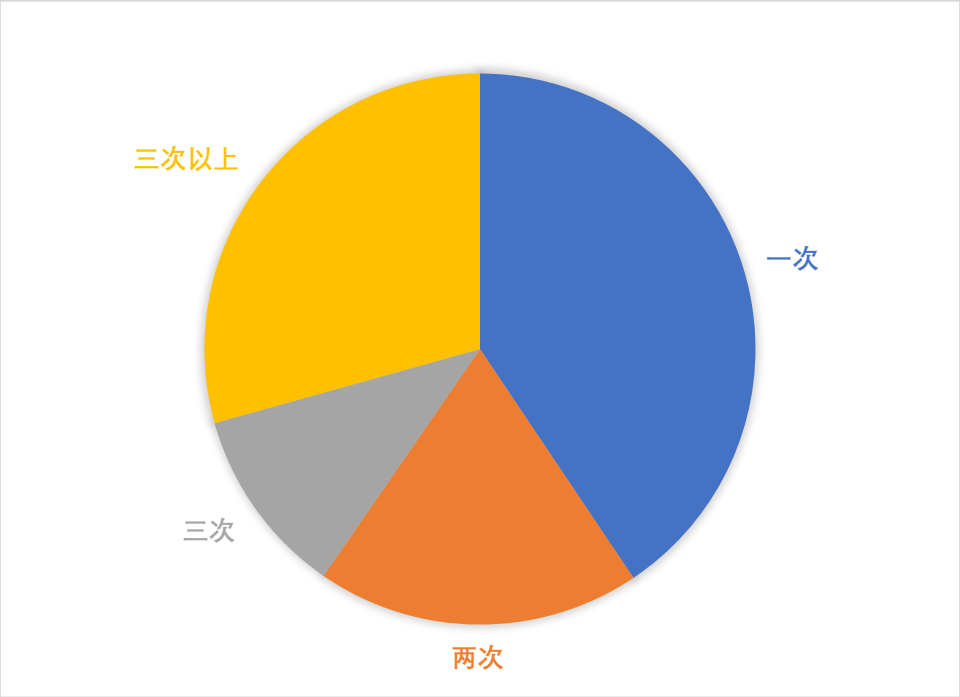
```
2018-11-02 18:05:58,552 Stage-2 map = 0%,  reduce = 0%
2018-11-02 18:06:00,555 Stage-2 map = 100%,  reduce = 100%
Ended Job = job_local1922809182_0010
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 33856270666 HDFS Write: 6883896324 SUCCESS
Stage-Stage-2:  HDFS Read: 11446093350 HDFS Write: 2294632108 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
百度    38437
baidu   18312
人体艺术        14474
4399小游戏      11438
qq空间  10316
优酷    10158
新亮剑  9654
馆陶县县长闫宁的父亲    9127
公安卖萌        8192
百度一下 你就知道       7504
百度一下        7104
4399    7041
魏特琳  6665
qq网名  6148
7k7k小游戏      5985
黑狐    5610
儿子与母亲不正当关系    5496
新浪微博        5369
李宇春体        5310
```

3.4.3 UID 分析

查询分别只有查询一次、两次、三次以及三次以上的 UID 的总用户数

```
Ended Job = job_local1993734024_0001
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2018-11-05 20:56:54,793 Stage-2 map = 100%,  reduce = 100%
Ended Job = job_local100996691_0002
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 3380011600 HDFS Write: 0 SUCCESS
Stage-Stage-2:  HDFS Read: 1287340328 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
549138  257164  149563  396780
Time taken: 46.163 seconds, Fetched: 1 row(s)
```

使用可视化工具将其可视化可以看出只查询一次的用户占据大多数、而三次以上的用户反而居于次位，所以可以看出来用户对搜狗搜索引擎的依赖不强，大多数人只是偶然性的使用。



所有用户的查询平均次数约为 3.69 次，大于三次，但是上图查询三次及三次以上的用户数不占据大头，因此可以推理得出查询三次以上的用户对于搜索的依赖较大，三次以上的用户的搜索总次数较大从而将平均值拉升。

```
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2018-11-05 21:05:40,736 Stage-2 map = 100%,  reduce = 100%
Ended Job = job_local565697954_0002
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 3380011600 HDFS Write: 0 SUCCESS
Stage-Stage-2:  HDFS Read: 1287340328 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
3.69637044457341
```

对用户参与的时间进行分析可以得到，记录始于 2011 年 12 月 30 日 0 时结束于 12 月 31 日 16 时。

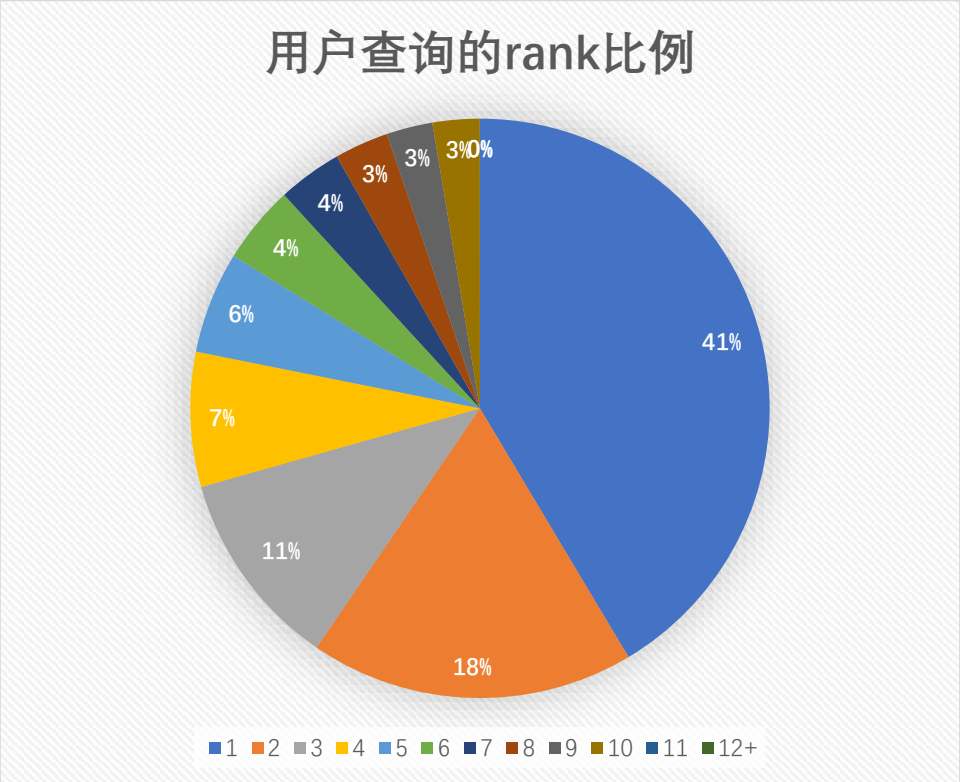
3.4.4 用户行为分析

Rank 在 10 以内的点击次数占比

Rank 在 10 以内的点击次数是：4999742

这五百万条中的非空总条目数是 4999877

最终可以得出 rank 在 10 以内的点击次数占比是 99.9973 几乎接近于 1，虽然样本数量不全面，但是基本上得出用户的习惯在于 rank 在 10 以内的搜索结果。



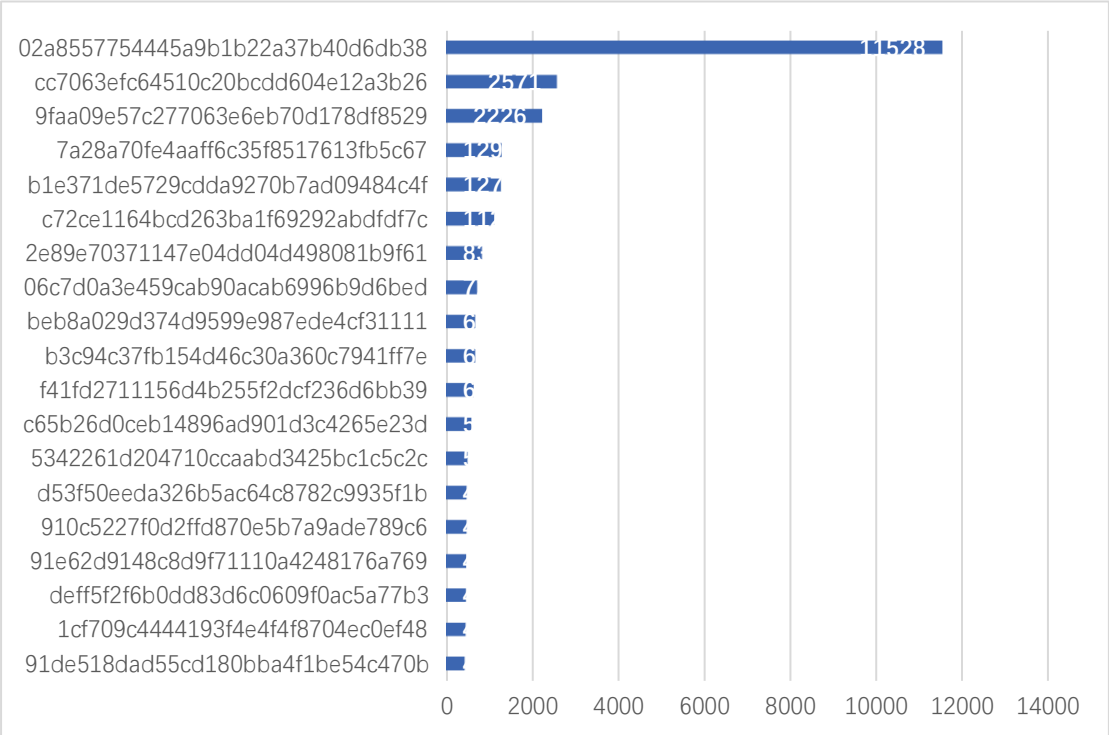
统计直接输入 URL 的查询比例

直接输入 URL 的条目数为 73979 条

总的数据条目数为 4999877

因此可以得出直接输入 URL 的查询比例为 1.48%，说明实际搜索网页的用户并不典型，而查询 URL 的条目中，用户实际点击的就是这条条目的总数为 27561 条，实际的用户转化率为 37.25%。很大一部分用户提交含有 URL 的查询是由于希望能够借助搜索引擎访问没有记全等原因的网址。

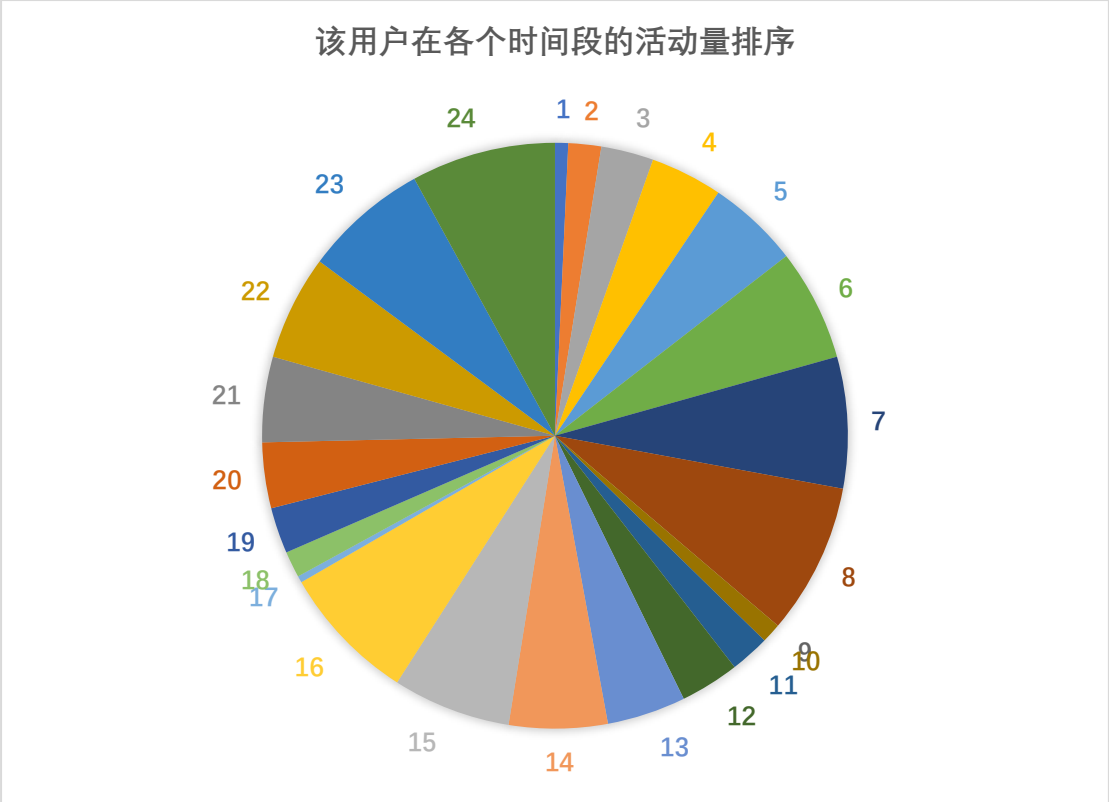
以上的分析全部都只针对条目数，并不指定实际的用户。之后对用户进行单独分析，例如查找重复查询多次的用户 uid。



图中可以看到有一个典型用户占据了远远超过其他人的搜索数量，进一步分析其搜索时段可以发现其在 24 个小时内都有活动，但活动量明显和时间有关。

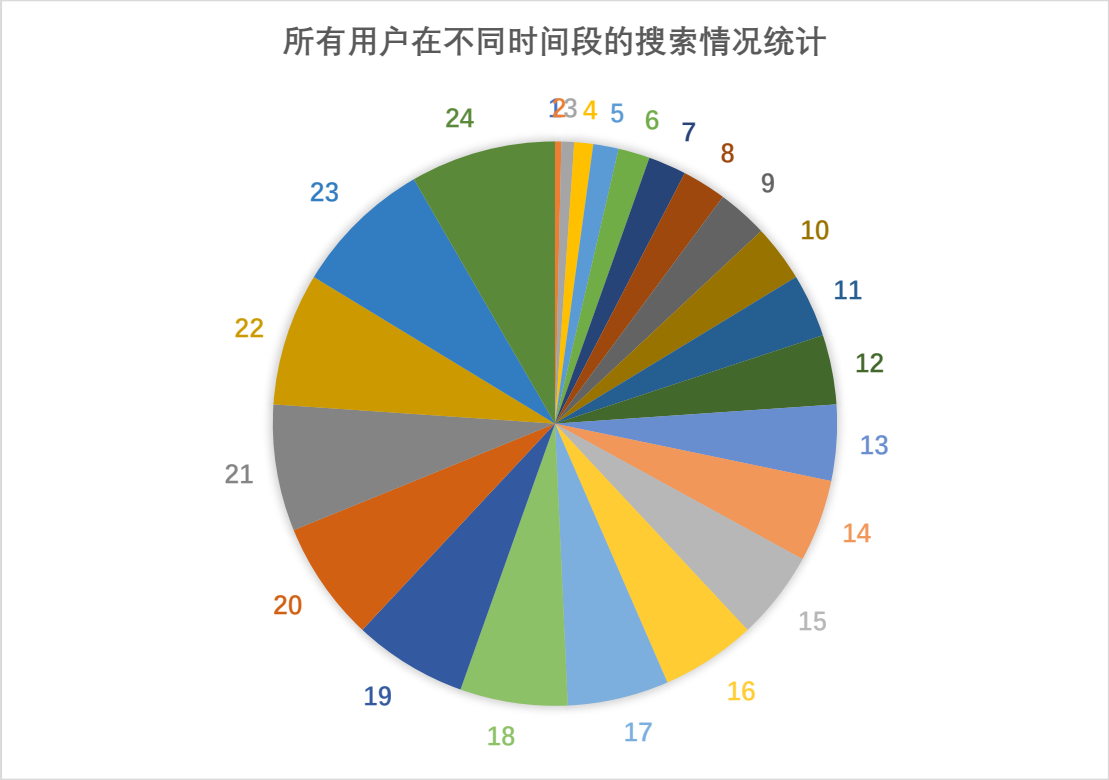
```
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 65172445648 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
02a8557754445a9b1b22a37b40d6db38        62      2011    12      30      2
02a8557754445a9b1b22a37b40d6db38        40      2011    12      30      5
02a8557754445a9b1b22a37b40d6db38        308     2011    12      30      8
02a8557754445a9b1b22a37b40d6db38        610     2011    12      30      11
02a8557754445a9b1b22a37b40d6db38        661     2011    12      30      14
02a8557754445a9b1b22a37b40d6db38        697     2011    12      30      17
02a8557754445a9b1b22a37b40d6db38        885     2011    12      30      20
02a8557754445a9b1b22a37b40d6db38        501     2011    12      30      23
02a8557754445a9b1b22a37b40d6db38        239     2011    12      30      0
02a8557754445a9b1b22a37b40d6db38        44      2011    12      30      3
02a8557754445a9b1b22a37b40d6db38        49      2011    12      30      6
02a8557754445a9b1b22a37b40d6db38        529     2011    12      30      9
02a8557754445a9b1b22a37b40d6db38        638     2011    12      30      12
02a8557754445a9b1b22a37b40d6db38        723     2011    12      30      15
02a8557754445a9b1b22a37b40d6db38        711     2011    12      30      18
02a8557754445a9b1b22a37b40d6db38        887     2011    12      30      21
02a8557754445a9b1b22a37b40d6db38        186     2011    12      30      1
02a8557754445a9b1b22a37b40d6db38        29      2011    12      30      4
02a8557754445a9b1b22a37b40d6db38        90      2011    12      30      7
02a8557754445a9b1b22a37b40d6db38        591     2011    12      30      10
02a8557754445a9b1b22a37b40d6db38        663     2011    12      30      13
02a8557754445a9b1b22a37b40d6db38        753     2011    12      30      16
02a8557754445a9b1b22a37b40d6db38        892     2011    12      30      19
02a8557754445a9b1b22a37b40d6db38        740     2011    12      30      22
Time taken: 7.526 seconds, Fetched: 24 row(s)
```

由于数字的展示效果不好，因此重新借用可视化工具进行可视化分析。



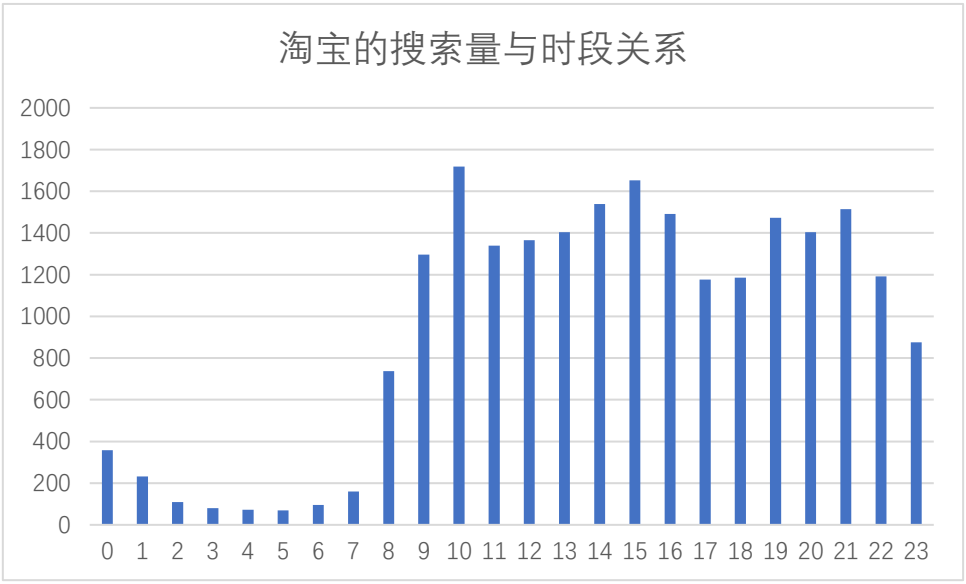
此时可以明显发现，用户在饭点（中午 12 点以及晚上 17-19 点）左右活动量明显下降，并且用户在深夜的活动较少，因此认为是统计失误，或者技术原因导致用户在统计的过程中被绑定在一起，因此产生如此巨大的搜索情况。

为此可以做数据统计，将所有的时间端和搜索量进行统计，最终得到下图：

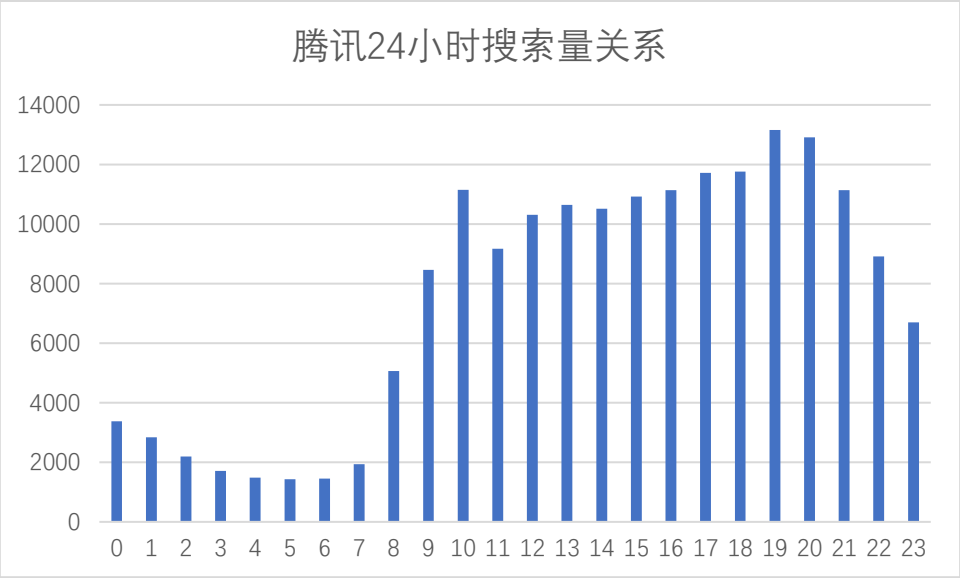


可以看到用户在上午的搜索量远远低于下午的搜索量，因此我们可以大致看出用户在当时已经养成晚睡早起的生活习惯，搜索活动主要集中在晚上 19 点至 24 点的时间段内，看来并不只是当今的 90 后，之前的 80 后的生活习惯也基本一致。

继续进一步分析，用户在各个时间端都会干什么呢？首先想到的是淘宝，淘宝于 2008 年创立，因此 11 年正处于快速上升期，查询的结果是总条目是 22539 条，并不多，并且搜索的时间关系基本和用户的活动时间保持重合。



接下来分析腾讯的相关访问量，由于腾讯的业务较广，涉及新闻，游戏，社交等多方面，此处就仅以域名中包含 qq 的条目数进行统计，查询可以得到总计 18 万的条目记录，基本可以得到一致的结果，但是腾讯在饭点上并没有衰减，而是反而有一个较小的提升。



四、总结

此次由于设备条件限制，无法部署多台的设备进行分布式集群部署，只能在单台机器上采用虚拟机的方式进行部署，因此无法发挥 Hadoop 的实际能力，在可用性和可靠性上问题比较大，同时还有 hive 的版本于 hadoop 的兼容性问题，导致 hive 出现许多问题，例如奔溃，丢失元数据，hive 经历了元数据丢失，经历了多次的重置，因此在之后在部署的时候要开始先开始进行调研版本环境的兼容性，由于 hive 支持数据直接将 hdfs 指定位置的文件直接导入，虽然经常丢失表结构，但是重建表却十分快捷，建立数据表的同时能够支持数据的自动生成，对于数据的分析可以有多个角度和分析方式，分析数据要能够有一个全面的思维方式，对于数据要有足够的敏感性同时能够学会利用各种数据可视化工具加强对于数据的分析以及展示能力。

此次的实验了解到了很多的 linux 系统操作，理解了 mapreduce 对于流操作的支持，了解例如 cat、grep、head、tail 等操作，学习了如何排查由于配置文件不正确造成的 hadoop 以及 hive 报错。

附录

Mapper 源码：

```
#!/usr/bin/env python
# coding=utf-8
import sys,os
import jieba

all_line=[]
for line in sys.stdin:  # 遍历读入数据的每一行
    line = line.replace(' ','') # 将多余的空格去除
    line = line.replace('.','') # 将多余的符号去除
    words = line.split("\t")    # 按 tab 将句子分割成单个单词
    all_line.append(words[2])
all_line.sort() # 进行内部排序
for line in all_line:
    print ('%s\t%s' %(line, 1)) # 流式输出
```

Mapper_url 源码

```
#!/usr/bin/env python
# coding=utf-8
import sys,os
import jieba

for line in sys.stdin:  # 遍历读入数据的每一行
    line = line.replace(' ','')      # 将多余的空格去除
    words = line.split("\t")         #按 tab 将句子分割成单个
    #cut=jieba.cut(words[5])
    #for word in words:
    print ('%s\t%s' %(words[5], 1))
    # print("")
```

Reducer 源码：

```
#!/usr/bin/env python
# coding=utf-8
from operator import itemgetter
import sys
current_word = None
current_count = 0
word = None
# 输入必须经过预先排序
for line in sys.stdin:
    word, count = line.split('\t', 1)
    try:
        count = int(count)
    except ValueError:
        continue
    if current_word == word:
        current_count += count
    else:
        if current_word:
            print "%s\t%s" % (current_word, current_count)
        current_count = count
        current_word = word

if word == current_word:    # 流式输出（无序）
    print "%s\t%s" % (current_word, current_count)
```