云计算

spark streaming

1 应用场景介绍

应用场景

在 Spark Streaming 中，处理数据的单位是一批而不是单条，而数据采集却是逐条进行的，因此 Spark Streaming 系统需要设置间隔使得数据汇总到一定的量后再一并操作，这个间隔就是批处理间隔。批处理间隔是 Spark Streaming 的核心概念和关键参数，它决定了 Spark Streaming 提交作业的频率和数据处理的延迟，同时也影响着数据处理的吞吐量和性能。

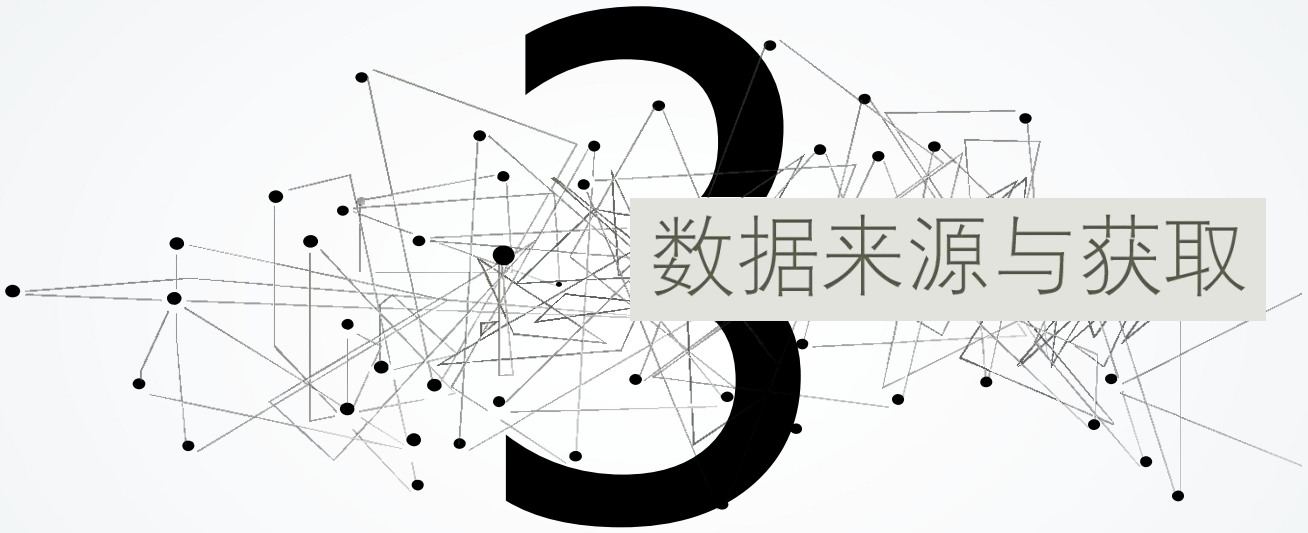
HDFS不适合大量小文件的存储,HDFS适用于高吞吐量，而不适合低时间延迟的访问,流式读取的方式，不适合多用户写入一个文件（一个文件同时只能被一个客户端写），以及任意位置写入（不支持随机写），支持文件尾部apend操作，或者文件的覆盖操作,HDFS更加适合写入一次，读取多次的应用场景。

2

研究问题描述

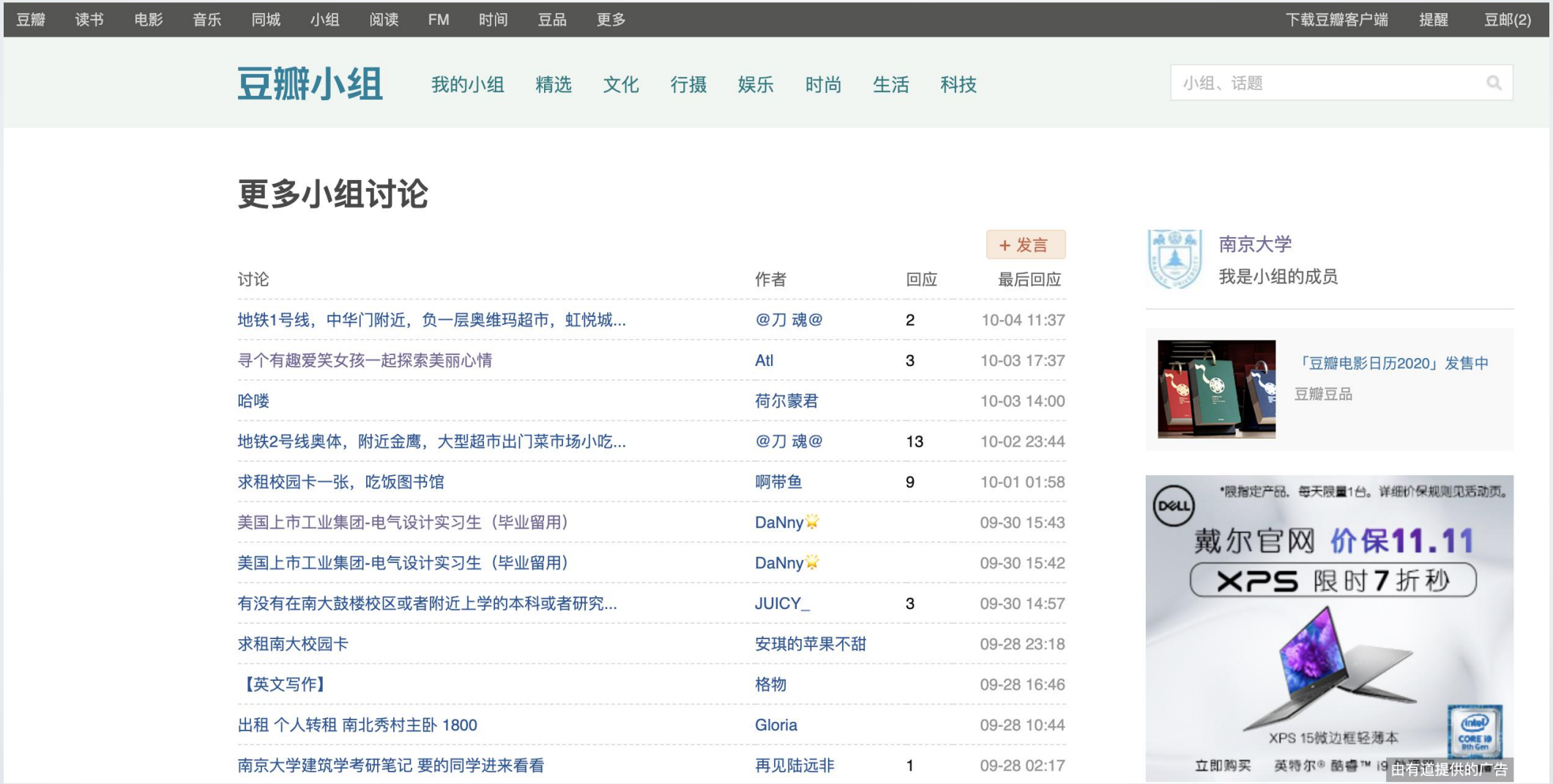
研究问题描述

- 我们学校的同学讨论最多的话题是什么？对哪些事情最感兴趣？
- 在不同的时间段同学们关注的话题有什么变化？



数据来源与获取

- 数据来源于豆瓣的南京大学小组



Python爬虫获取数据

- 使用selenium+requests模拟登录并获取南京大学小组的网页数据

```
# 豆瓣登录页面URL
# login_url = 'https://www.douban.com/accounts/login'
login_url = 'https://www.douban.com'

# 获取chrome的配置
opt = webdriver.ChromeOptions()
# 在运行的时候不弹出浏览器窗口
if self.headless:
    opt.set_headless()

# 获取driver对象
self.driver = webdriver.Chrome(chrome_options = opt)
# 打开登录页面
self.driver.get(login_url)

print '[login] opened login page...'
self.driver.implicitly_wait(10)

# 向浏览器发送用户名、密码，并点击登录按钮
self.driver.switch_to.frame(self.driver.find_elements_by_tag_name('iframe')[0])
self.driver.find_element_by_xpath('/html/body/div[1]/div[1]/ul[1]/li[2]').click()
```

Python爬虫获取数据

- 使用lxml库的xpath语法解析网页数据

```
html = etree.HTML(page_html)
# 解析话题讨论列表
trs = html.xpath('//*[@class="olt"]/tr')[1:]

for tr in trs:
    title = tr.xpath('./td[1]/a/text()')[0].strip()
    link = tr.xpath('./td[1]/a/@href')[0].strip()
    # 继续解析话题详情页面，从中解析出发布时间、描述详情
    topic_page_html = self.get_html(link)
    topic = etree.HTML(topic_page_html)
    # 发布时间字符串
    post_time_str = topic.xpath('//*[@class="topic-doc"]/h3[1]/span[2]/text()')[0].strip()
    # 详情
    if topic.xpath('//*[@class="topic-content"]') == []:
        detail = topic.xpath('//*[@class="link-rec"]')[0].xpath('string(.)').strip()
    else :
        detail = topic.xpath('//*[@class="topic-content"]')[0].xpath('string(.)').strip()

    # 根据关键词过滤
    if filter and not self.contains(title, filter) and not self.contains(detail, filter):
        continue
```


4
数据展示

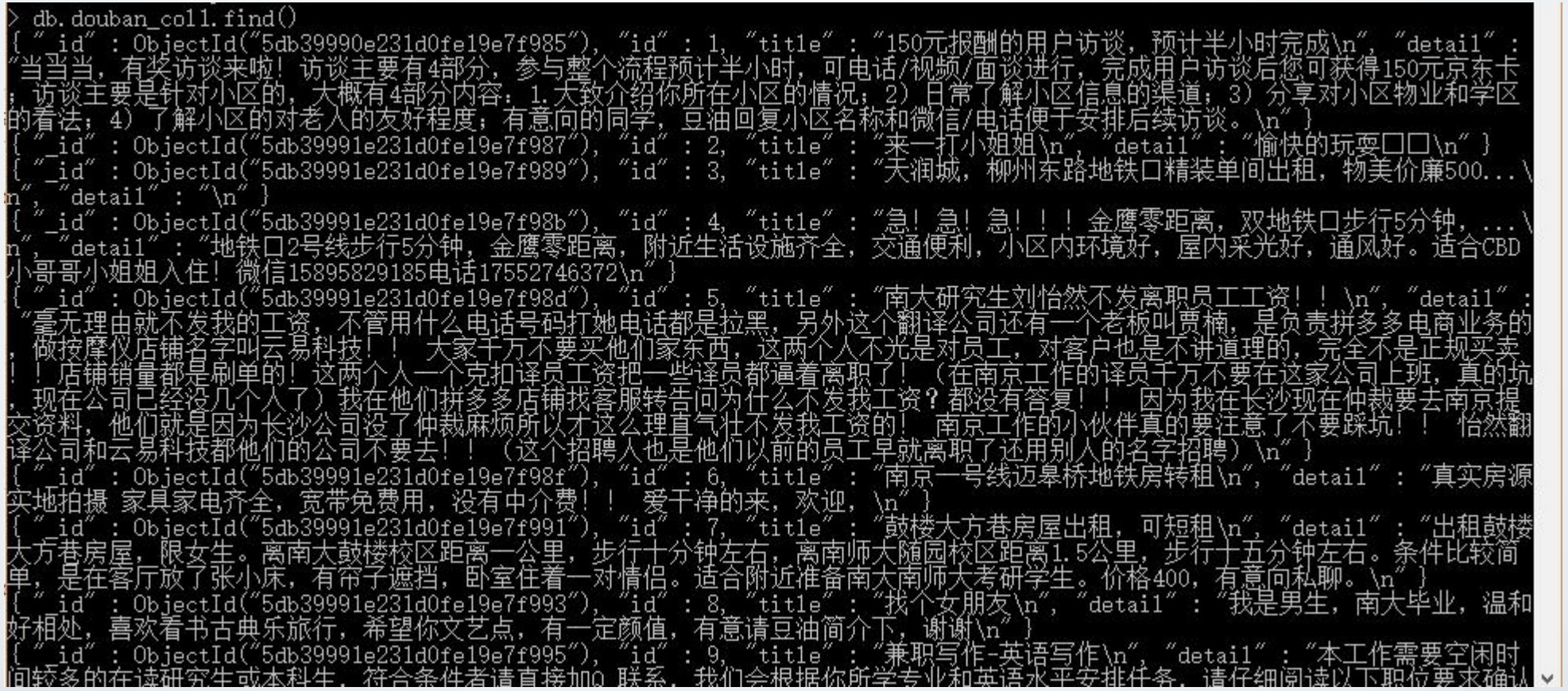
- 将话题数据渲染到HTML网页中，并按发布时间排序

标题	发布时间	链接
柳州东路威尼斯水城转租	2019-10-21 21:24:46	https://www.douban.com/group/topic/155717510/
有没有南大图书情报学专业的童鞋呀？想考这个专业...	2019-10-21 19:16:23	https://www.douban.com/group/topic/155706933/
来一打小姐姐	2019-10-21 12:55:14	https://www.douban.com/group/topic/155672644/
急！急！急！！！金鹰零距离，双地铁口步行5分钟，...	2019-10-21 10:03:05	https://www.douban.com/group/topic/155654758/
南大研究生刘怡然不发离职员工工资！！	2019-10-20 21:09:03	https://www.douban.com/group/topic/155622821/
鼓楼大方巷房屋出租，可短租	2019-10-20 14:35:54	https://www.douban.com/group/topic/155591728/
招聘各专业英文写手，每千词200-300元。	2019-10-19 15:03:28	https://www.douban.com/group/topic/155503740/
南京一号线迈皋桥地铁房转租	2019-10-18 18:02:15	https://www.douban.com/group/topic/155428540/
150元报酬的用户访谈，预计半小时完成	2019-10-18 11:22:33	https://www.douban.com/group/topic/155392574/
直租。西安门附近。瑞金北村精装单间。1660每月，...	2019-10-12 17:15:59	https://www.douban.com/group/topic/154867999/
求essay写手	2019-10-12 15:45:30	https://www.douban.com/group/topic/154858719/
明故宫附近。黄埔鑫园合租1190每月，直租，拒绝中...	2019-10-12 10:46:25	https://www.douban.com/group/topic/154826428/
急！！！！急！！急！！建业区金鹰零距离房源出租...	2019-10-12 10:46:16	https://www.douban.com/group/topic/154826405/
急！！！！急！！急！！建业区金鹰零距离房源出租...	2019-10-12 10:04:49	https://www.douban.com/group/topic/154820843/
直租钟灵街。紫金华府合租。1060每月，精装单间，...	2019-10-11 17:28:19	https://www.douban.com/group/topic/154766680/

5 数据存储

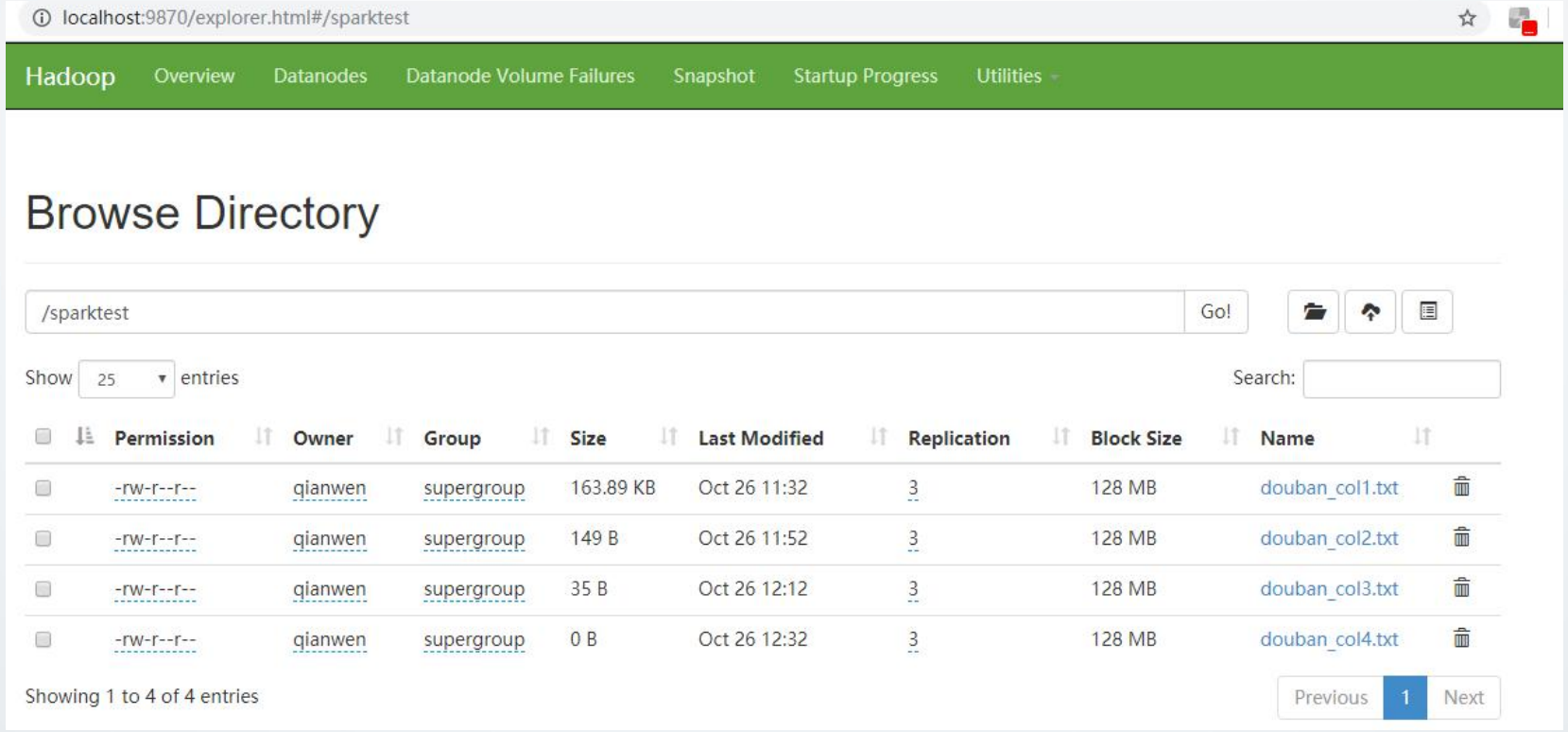
Mongodb&HDFS

- 爬虫爬取的数据存储在mongodb中，不同的时间段存储在同一数据库的不同集合（collection）中



Mongodb&HDFS

- mongodb中的数据通过模拟流上传到HDFS中，由于不同时间段存储在mongodb的不同集合中，故每隔一段时间从不同的集合中读取数据。



6 数据格式

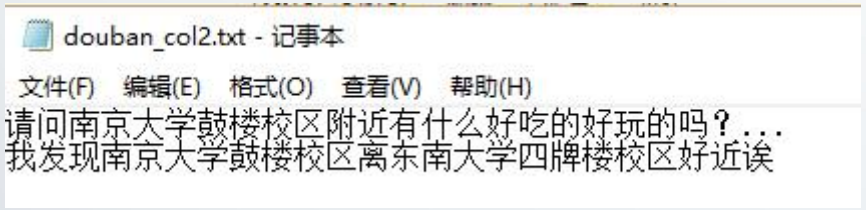
Mongodb&HDFS数据格式

- mongodb中数据的格式包含_id、id、title（豆瓣帖子标题）、detail（豆瓣帖子内容）

```
> db.douban_col2.find()
{ "_id" : ObjectId("5db3b9a9ea3ab9a7ab942f5f"), "id" : 1, "title" : "请问南京大学鼓楼校区附近有什么好吃的好玩的吗？...\n
", "detail" : "请问南京大学鼓楼校区附近有什么好吃的好玩的吗？同学们有什么推荐吗？\n" }
{ "_id" : ObjectId("5db3b9a9ea3ab9a7ab942f61"), "id" : 2, "title" : "我发现南京大学鼓楼校区离东南大学四牌楼校区好近诶",
"detail" : "真的好近" }
```

- 由于我们发现具体内容里面包含了许多冗余的信息，标题的信息更具有代表性，故我们决定研究所有标题的词频，HDFS仅上传了标题部分，我们将HDFS中的文件下载，查看包含的具体内容：

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	qianwen	supergroup	163.89 KB	Oct 26 11:32	3	128 MB	douban_col1.txt
-rw-r--r--	qianwen	supergroup	149 B	Oct 26 11:52	3	128 MB	douban_col2.txt



7 解决方案及代码

transformation操作

- flatMap

```
JavaDStream<String> words = file.flatMap(new FlatMapFunction<String, String>() {
    public Iterator<String> call(String line) throws Exception {
        //return Arrays.asList(line.split("\\W+")).iterator();
        //return strlist.iterator();
        return getSplitWords(line).iterator();
    }
});
```

- mapToPair

```
JavaPairDStream<String, Integer> wordMap = words.mapToPair(new PairFunction<String, String, Integer>() {
    public Tuple2<String, Integer> call(String word) throws Exception {
        return new Tuple2<String, Integer>(word, 1);
    }
});
```

- filter

```
JavaPairDStream<String, Integer> filterRDD = wordMap.filter(new Function<Tuple2<String, Integer>,Boolean>(){
    public Boolean call(Tuple2<String, Integer> t2) throws Exception {
        if(stopWordSet.contains(t2._1)){
            return false;
        }else{
            return true;
        }
    }
});
```

transformation操作

- reduceByKey

```
JavaPairDStream<String, Integer> reduceWord = filterRDD.reduceByKey(new Function2<Integer, Integer, Integer>() {
    public Integer call(Integer v1, Integer v2) throws Exception {
        return v1 + v2;
    }
});
```

- updateStateByKey

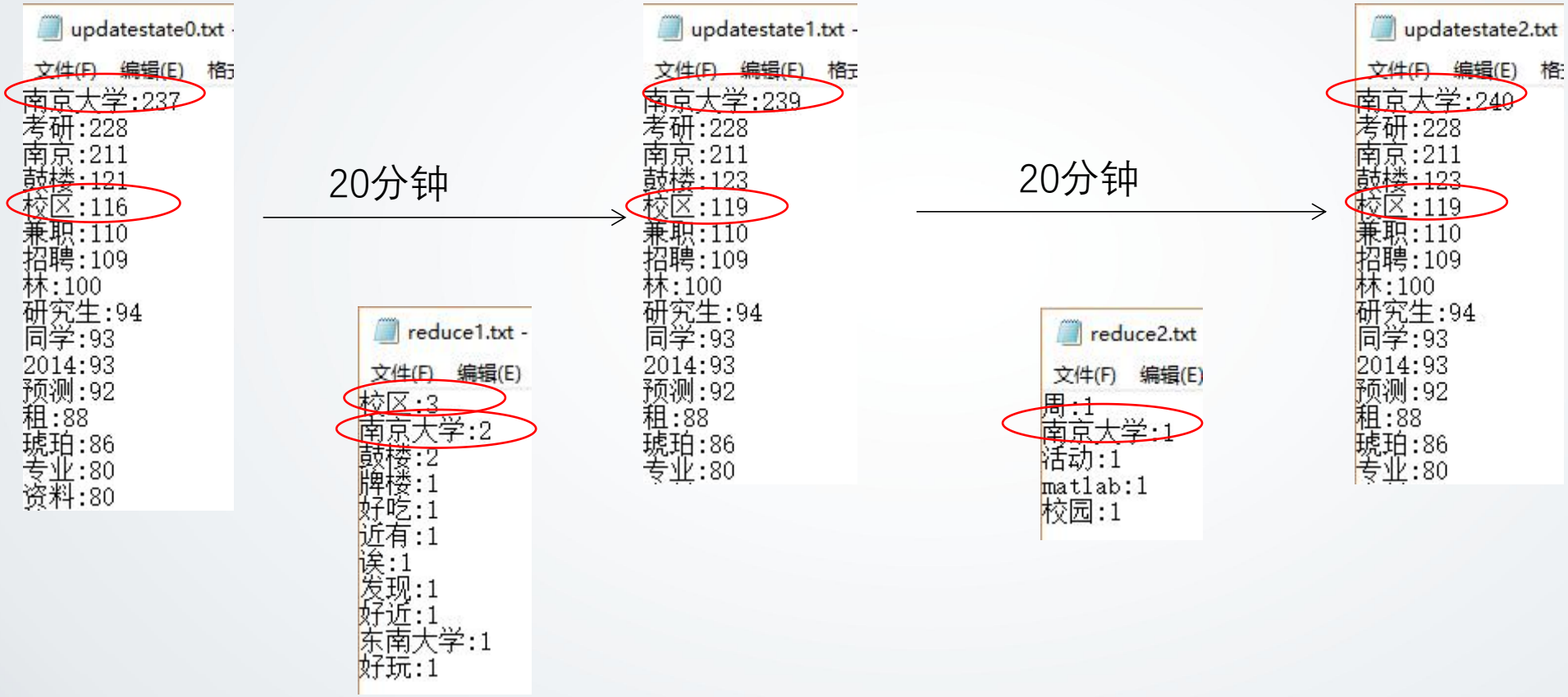
```
JavaPairDStream<String, Integer> stateWord = filterRDD.updateStateByKey(new Function2<List<Integer>, Optional<Integer>, Opti
    private static final long serialVersionUID = 1L;

    public Optional<Integer> call(List<Integer> values, Optional<Integer> state) throws Exception {
        Integer updatedValue = 0 ;
        if(state.isPresent()){
            updatedValue = state.get();
        }
        for(Integer value: values){
            updatedValue += value;
        }
        return Optional.of(updatedValue);
    }
});
```

- transformToPair & sortByKey

```
JavaPairDStream<Integer, String> sortPairRDD = pairRDD.transformToPair(pairRdd -> pairRdd.sortByKey(false));
```

全局统计的量：词频

- 通过updateStateByKey实现



全局统计的量：词频

- 通过updateStateByKey实现

```
JavaPairDStream<String, Integer> stateWord = filterRDD.updateStateByKey(new Function2<List<Integer>, Optional<Integer>, Opti
    private static final long serialVersionUID = 1L;

    public Optional<Integer> call(List<Integer> values, Optional<Integer> state) throws Exception {
        Integer updatedValue = 0 ;
        if(state.isPresent()){
            updatedValue = state.get();
        }
        for(Integer value: values){
            updatedValue += value;
        }
        return Optional.of(updatedValue);
    }
});
```

```
resultRdd1.foreachRDD(new VoidFunction<JavaPairRDD<String, Integer>>() {
    public void call(JavaPairRDD<String, Integer> stringIntegerJavaPairRDD) throws Exception {
        stringIntegerJavaPairRDD.foreach(new VoidFunction<Tuple2<String, Integer>>() {
            @Override
            public void call(Tuple2<String, Integer> t2) throws Exception {
                //ConnectionPool的定义与ConnectionPoolTitle的一样
                System.out.println("出现的单词是      : " + t2._1);
                System.out.println("单词出现的次数是 : " + t2._2);
            }
        });
      //保存结果到文件中
        System.out.println("保存到文件中");
        writetofile(stringIntegerJavaPairRDD,"doc\\updatestate_res\\updatestate"+endtmp+".txt");
        endtmp++;
    }
});
```

Spark streaming监听HDFS文件夹

- 每20分钟监听一次：

```
SparkConf conf =
        new SparkConf().setMaster("local").setAppName("WordCount_Java");
JavaStreamingContext jsc = new JavaStreamingContext(conf, Durations.minutes(20));
jsc.checkpoint("./checkpoint");
```

```
JavaDStream<String> file = jsc.textFileStream("hdfs://master:9000/sparktest");
```

```
jsc.start();
try {
    jsc.awaitTermination();
} catch (InterruptedException e) {
    e.printStackTrace();
}
jsc.stop();
jsc.close();
```

Mongodb读数据写入HDFS

- 写入HDFS：

```
public static void WriteToHDFS(String file, String words) throws IOException, URISyntaxException
{
    Configuration conf = new Configuration();
    FileSystem fs = FileSystem.get(URI.create(file), conf);
    Path path = new Path(file);
    FSDataOutputStream out = fs.create(path);   //创建文件
    out.write(words.getBytes("UTF-8"));
    out.close();
}
```

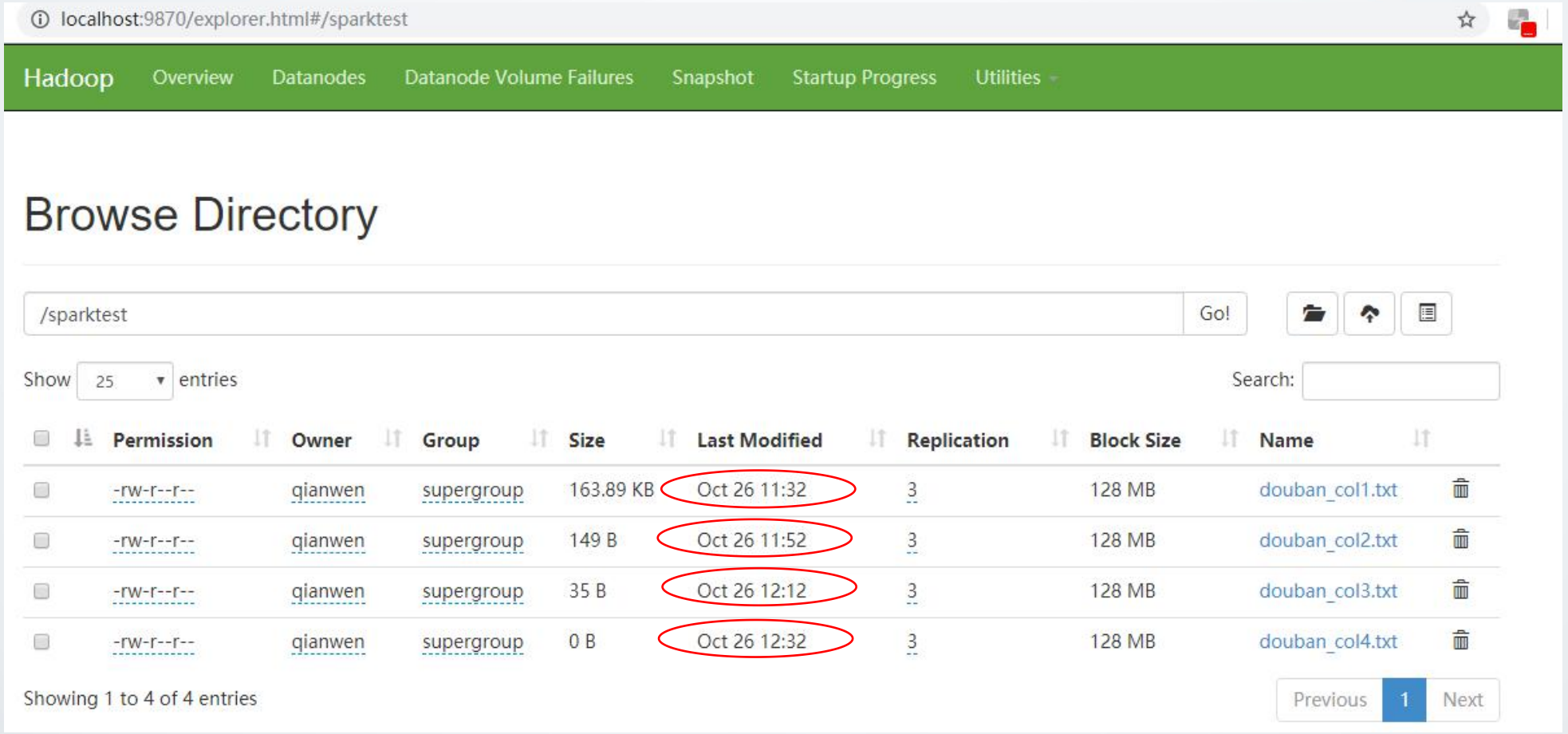
- 从mongodb读数据并调用写入HDFS函数：

```
FindIterable<Document> find = collection.find();
MongoCursor<Document> mongoCursor = find.iterator();
String file = "hdfs://master:9000/sparktest/"+timecol+".txt";
String filecontext="";
while(mongoCursor.hasNext()){
    Document studentDocument = mongoCursor.next();
    System.out.println(studentDocument.getString("title"));
    //System.out.println(mongoCursor.next());
    filecontext+=studentDocument.getString("title")+"\r\n"; //
 }
try {
    WriteToHDFS(file, filecontext);
} catch (IOException | URISyntaxException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
```

8 结果展示与评价

Mongodb -> HDFS

- 每20分钟从mongodb更新数据到HDFS



Spark streaming

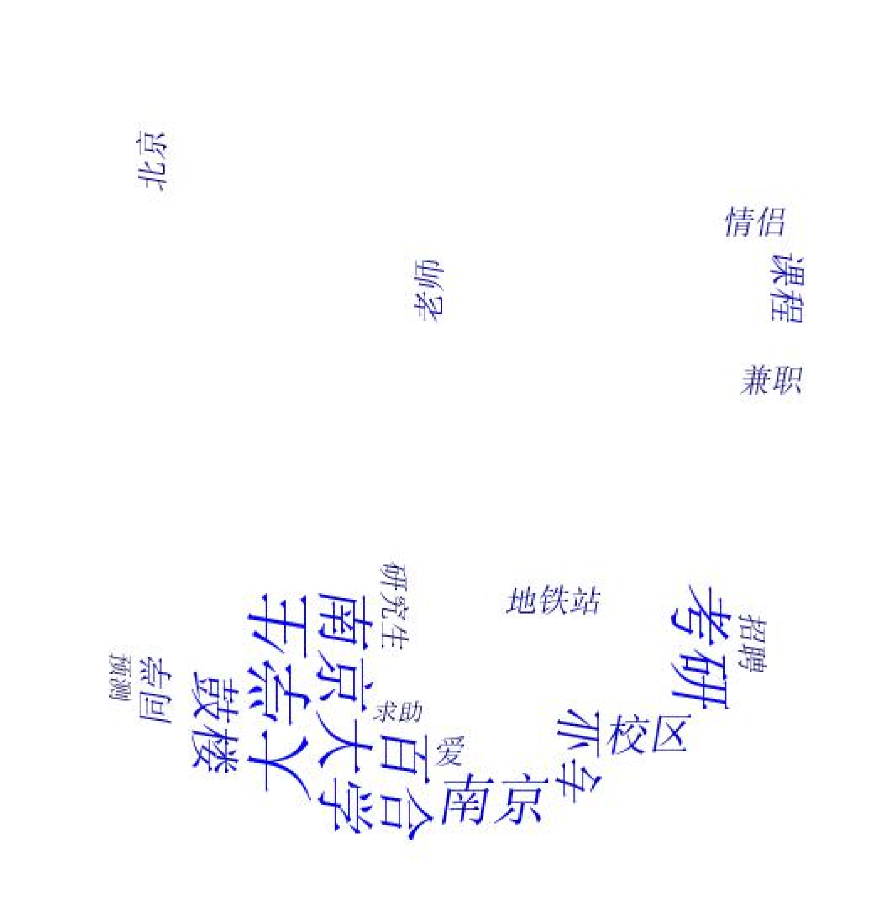
- 每20分钟做一次阶段性和全局的词频统计
- 阶段性：

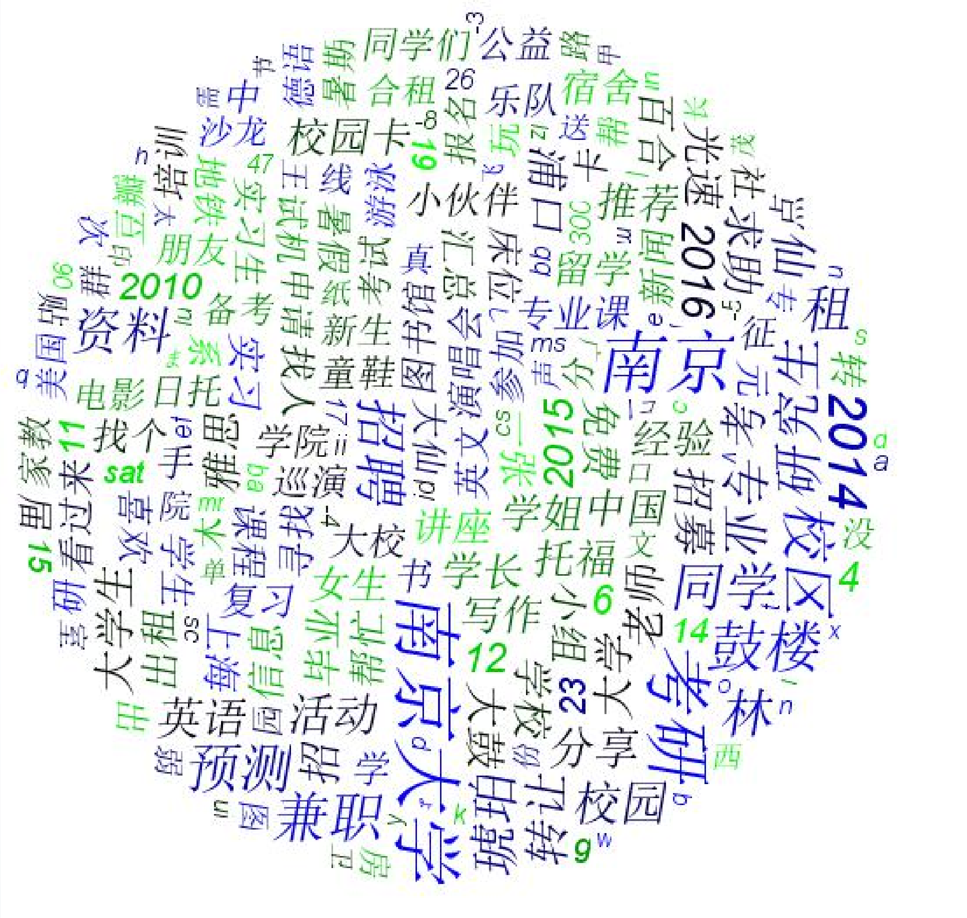
名称	修改日期	类型	大小
reduce0.txt	2019/10/26 星期六 11:40	文本文档	58 KB
reduce1.txt	2019/10/26 星期六 12:00	文本文档	1 KB
reduce2.txt	2019/10/26 星期六 12:20	文本文档	1 KB
reduce3.txt	2019/10/26 星期六 12:40	文本文档	0 KB

- 全局：

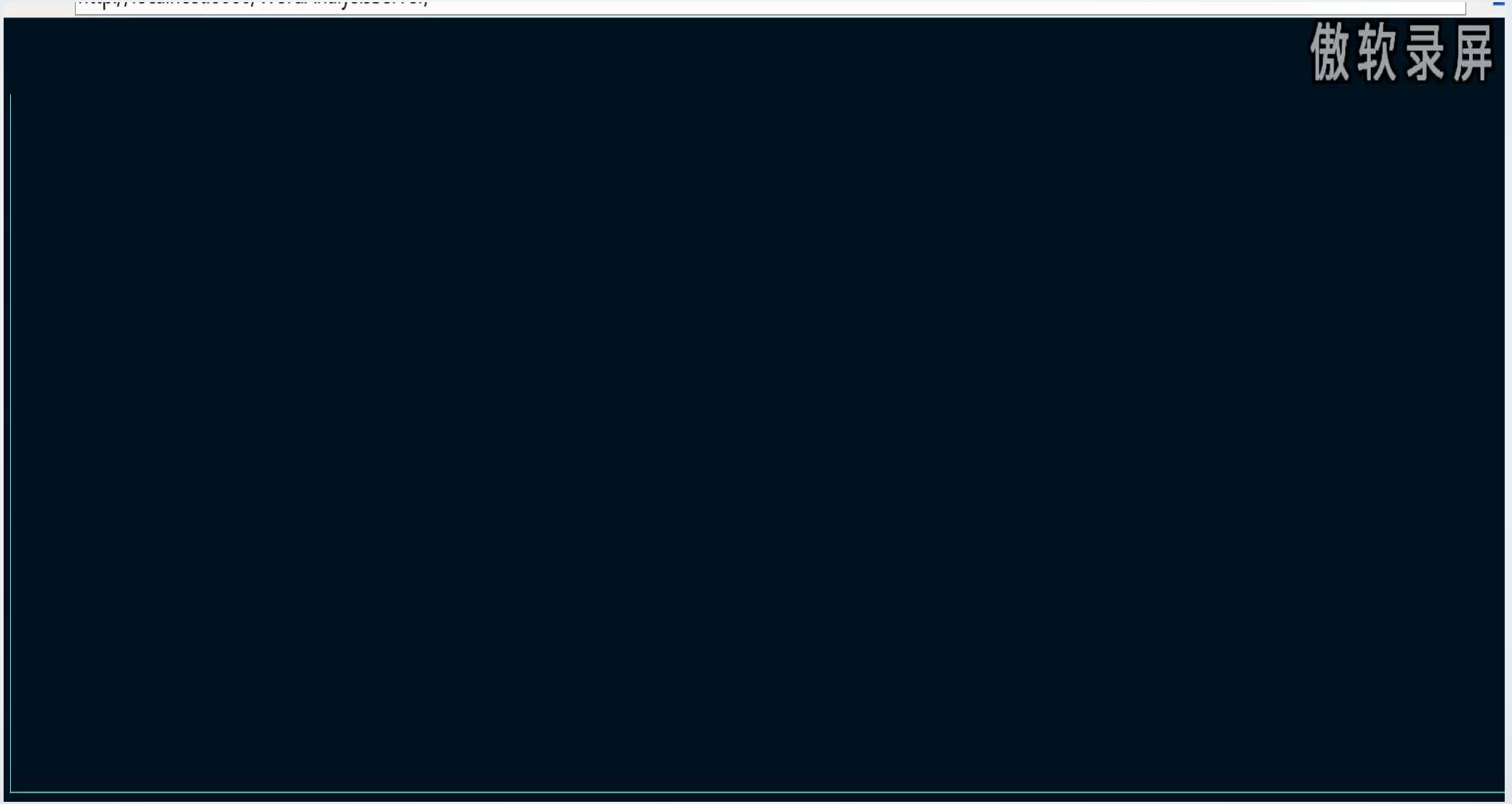
名称	修改日期	类型	大小
updatestate0.txt	2019/10/26 星期六 11:40	文本文档	58 KB
updatestate1.txt	2019/10/26 星期六 12:00	文本文档	58 KB
updatestate2.txt	2019/10/26 星期六 12:20	文本文档	58 KB
updatestate3.txt	2019/10/26 星期六 12:40	文本文档	58 KB

词云展示


北京
情侣
老师
课程
兼职
地铁站
研究生
招聘
考研
南京大学
大学生
鼓楼
同学
求助
百合
爱
南京
专业
校区


南京
大学
考研
兼职
预测
招聘
资料
校区
鼓楼
同学
专业
研究生
2014
2016
2015
2010
校园卡
小伙伴
推荐
光速
宿舍
乐队
公益
同学们
合租
英语
活动
讲座
学姐
学长
托福
雅思
复习
课程
上海
女生
写作
分享
校园
大学生
经验
留学
专业课
新生
实习
电影
朋友
地铁
暑假
sat

柱状图功能展示



代码实现

每次选取文件夹中最新的txt文件：

```
try
{

    Collection<File> fileList = FileUtils.listFiles(new File(dirPath),
            FileFilterUtils.suffixFileFilter("txt"), DirectoryFileFilter.DIRECTORY);
    File resultFile = null;
    if (fileList.size()>0) {
        for (File itemFile : fileList) {
            if (resultFile==null||(resultFile.lastModified()<itemFile.lastModified())) {
                resultFile = itemFile;
            }
        }
    }
```

处理txt中每行数据排序选取前30：

```
if (resultFile!=null) {
    String resultString = FileUtils.readFileToString(resultFile, "GBK");
    String[] glist = resultString.split("\r\n");
    for (String sword : glist) {
        if (sword.length()>1) {
            String[] ggs = sword.split(":");
            DataBean dbBean = new DataBean(ggs[0],Long.valueOf(ggs[1]));
            list.add(dbBean);
        }
    }
    Collections.sort(list);
}
if(list.size()>30){
    for (int i = 0; i < 30; i++) {
        finalList.add(list.get(i));
    }
}else{
    finalList.addAll(list);
}
```

柱状图结果展示

评价

由于小组中信息流量在一定时间内变化幅度很小，在前面做了功能展示，这里的结果展示只有较小幅度变化。

9 总结与展望

总结与展望

本次实验实现了从豆瓣网页爬取信息、存储到Mongodb、从Mongodb模拟流到HDFS、用Spark streaming监听HDFS并统计词频（全局&阶段性词频）、用词云和柱状图可视化动态展示词频结果。

当然，由于时间原因，我们选择了20分钟更新一次，导致研究对象（南京大学豆瓣小组）的词频变化并不明显，也是存在一点小不足。未来如果想要改进的话，会增大更新时间间隔，并且增加与其他大学小组词频统计结果的对比。

总的来说，本次实验是比较成功的，我们完成了基本的功能要求，也从中学到了很多知识。

谢谢！