### 个人规划

1. **既然没有牛逼闪闪的项目，就把基础知识打牢**
2. **每学习一个知识点，要想清楚为什么要学习，不要盲目的去搞半天浪费时间，学到东西总结博客**
3. **基础知识主要是先刷leetcode，看数据结构的书总结，算法一定要过关，遇到不会的算法要好好总结，通过书+博客**
4. **索引很重要！！！一定要把常见的B树那些搞明白，自己代码实现一遍**
5. **大数据先从大数据生态开始，然后往机器学习、深度学习靠，面试可以没有机器学习经验，但是要能说出点东西**
6. **Hadoop生态要把每个细节扣一下，不能草草略过，但是也要控制时间，每一个知道宏观的实现流程，然后一点一点挖细节**
7. **学习一些新的大数据框架，sparkstreaming的项目把整个流程搞通**
8. **广东大数据的项目主要是偏研究，平时多看论文，这就是做研究生的一个毕业东西还是大头，主要可以通过这个项目来学习Hadoop的生态**

**书籍：**

**《大话数据结构》**

**《数据结构与算法分析》**

**《Linux 鸟哥的私房菜》**

**《大数据技术原理与应用》**

**《机器学习》 （西瓜书）**

**《Hadoop权威指南》**

**《HBase权威指南》**

**《数据算法 Hadoop、Spark大数据处理技巧》**

简历项目：

SparkStreaming（实时流处理）

广东大数据项目（离线批处理）

个人学习：

基础知识(java+scala+数据结构+算法+计算机网络+操作系统) ——>Hadoop、Spark等生态——>机器学习

### 时间安排

坚持写总结与计划（每日总结+每周总结）

**第一阶段：**

1. Linux基础

书籍参考：《Linux鸟哥的私房菜》

视频推荐：无

主要学习内容：

1. Linux操作系统介绍以及安装
2. Linux常用命令
3. Linux常用软件安装
4. Linux网络
5. 防火墙
6. **Shell编程**

**。。。。。。。。。。。**

1. Java基础+Java高级特性

书籍参考：《java核心技术》（卷1+卷2）+《深入理解java虚拟机》

视频推荐：李兴华

主要学习内容：

1. 基本的语法基础，编码能力
2. 掌握多线程
3. 掌握并发包下的队列
4. 掌握JVM技术
5. 掌握反射和动态代理

。。。。。。。。。。

1. 算法基础学习

书籍参考：《大话数据结构》、《数据结构与算法分析》、《数据算法》

视频推荐：慕课网

算法网站：leetcode+nowcoder

主要学习内容：

1. 手写递归，动规等
2. 常见排序、查找等算法代码手写
3. OJ代码练习
4. 分析常见算法的时间、空间复杂度等

。。。。。。。。。。

**第二阶段（攻坚阶段）：**

1. Hadoop

书籍参考：《Hadoop权威指南》、官方网站、林子雨大数据

视频推荐：八斗生态介绍、十小时大数据入门

源码：github源码精读

主要学习内容：

1. HDFS

论文：《GFS》

HDFS的概念和特性

HDFS的shell操作

HDFS的工作机制

HDFS的java应用开发

1. MapReduce

运行wordcount实例

了解Mapreduce内部运行机制：

Mapreduce程序运行流程解析

MapTask并发数的决定机制

MapReduce中的combiner组件应用

Mapreduce中的序列化框架及应用

Mapreduce中的排序

Mapreduce中的自定义分区实现

Mapreduce的shuffle机制

Mapreduce利用数据压缩进行优化

Mapreduce与Yarn之间的关系

Mapreduce参数优化

1. Mapreduce的java应用开发
2. Hive
3. Hive 基本概念

Hive 应用场景。

Hive 与hadoop的关系。

Hive 与传统数据库对比。

Hive 的数据存储机制。

1. Hive 基本操作

Hive 中的DDL操作。

在Hive 中如何实现高效的JOIN查询。

Hive 的内置函数应用。

Hive shell的高级使用方式。

Hive 常用参数配置。

Hive 自定义函数和Transform的使用技巧。

Hive UDF/UDAF开发实例。

1. Hive 执行过程分析及优化策略
2. HBase

hbase简介。

habse安装。

hbase数据模型。

hbase命令。

hbase开发。

hbase原理。

1. Scala

Scala概述。

Scala编译器安装。

Scala基础。

数组、映射、元组、集合。

类、对象、继承、特质。

模式匹配和样例类。

了解Scala Actor并发编程。

理解Akka。

理解Scala高阶函数。

理解Scala隐式转换。

1. Spark
2. Spark core

Spark概述。

Spark集群安装。

执行第一个Spark案例程序（求PI）

1. RDD

RDD概述。

创建RDD。

RDD编程API（Transformation 和 Action Operations）。

RDD的依赖关系

RDD的缓存

DAG（有向无环图）

1. Spark SQL and DataFrame/DataSet

Spark SQL概述。

DataFrames。

DataFrame常用操作。

编写Spark SQL查询程序。

1. Spark Streaming

Spark Streaming概述。

理解DStream。

DStream相关操作（Transformations 和 Output Operations）。

1. Structured Streaming
2. 其他（MLlib and GraphX ）
3. Python

廖博客

**第三阶段：（辅助工具学习阶段）**

1. Sqoop
2. Flume
3. Oozie
4. Hue

### 职业规划

### 简历准备

1、写上自己的博客地址（需要将搭建博客的原理搞清楚）

### 本周计划及进展：（2018.2.26-2018.3.4）

1.搭建Spark平台

2.安装scala

3.学习sparkGIS

4.学习GeoSpark

概要设计：

1、基于Spark的空间数据管理的现状（附文献）

2、构建Spark集群环境

3、将GeoSpark调试通过并运行

4、提交上述过程中形成的技术文档

### 本周计划及进展：（2018.3.5-2018.3.11）

1、基于Spark的空间数据管理的现状（附文献）

### 本周计划及进展：（2018.3.12-2018.3.18）

1、阅读了广东大数据项目的整个项目规划（2018.3.12上午）

总体解决问题：多数据采集-时空数据管理-多尺度建模-融合挖掘分析

->高性能处理分析

1. GitHub上cloneGeoSpark项目进行调试运行（2018.3.12下午 未完成）
2. GeoSpark就是一个在Apache Spark官方第三方项目中的一个基础项目（https://github .com/DataSystemsLab/GeoSpark readme介绍）同时是一个处理大规模空间数据的集群计算系统（2018.3.13上午）
3. 翻译对于GeoSpark的官方介绍并整理文档（2018.3.13）
4. 调试并运行GeoSpark（2018.3.13下午+晚上 部分完成）
5. 调试运行GeoSpark（2018.3.14上午完成，但是在本地编译出现异常，查看信息应该是Spark集群的问题，现在一台电脑没法搭建）
6. 阅读一个对于GeoSpark使用的demo

### 本周计划及进展：（2018.3.19-2018.3.25）

1. 运行GeoSpark实现对数据的处理并进行跟踪调试
2. 阅读SparkGIS官方文档，下载并运行调试
3. 阅读Spark环境下空间数据管理的相关论文

进展

1. 汇报上周情况，在北区机器下载运行GeoSpark（2018.3.19全天）

本周计划：

1. 调试运行GeoSpark并测试数据，对比分析

### 本周计划及进展：（2018.4.16-2018.4.22）

1. 学习spark核心编程
2. 学习Scala编程
3. 调试阅读GeoSpark

进展：

2018.4.16

1. 通过GeoSpark项目中pom文件学习maven管理的相关知识，包括scm、、dependencies等
2. Geotools？GeoTools is an open source Java library that provides tools for geospatial data

### 本周计划及进展：（2018.5.28-2018.6.2）

1. 选定测试数据，先用小数据county\_small.tsv 已经上传 需要采用PreprocesserApp对数据进行预处理
2. 将服务器原有的代码进行update到version42，然后进行打包编译mvn package -Dmaven.test.skip=true，显示的编译过程中出现unmappable character for encoding ASCII错误，应该是系统编码的问题
3. 编译以后可以看到在target/classes/目录下产生了每个Java文件编译的class文件

准备直接运行该路径下的class

4、hdfs存放数据路径是：hdfs：//geosciences/data/spatialhadoop

通过logs统计创建的时间

需要搜索idea打包将对应的class打包即可 放在指定的包路径下 远程提交spark-yarn模式运行

了解如何统计spark计算一个application的时间

**5.30**

1. 将PreprocesserApp打包并放在了namenode1服务器上/data/driverLib/geobeam.jar，然后提交processerApp：

spark-submit --class gtl.geobeam.app.PreprocesserApp --master yarn --deploy-mode cluster /data/driverLib/geobeam.jar 10

出现错误：Exception in thread "main" java.lang.IllegalArgumentException: Required AM memory (10240+1024 MB) is above the max threshold (8192 MB) of this cluster!

应该是默认的AMmemory超出了集群的限制 ，

解决方法：指定memory为2g

1. 重新提交运行spark-submit --class gtl.geobeam.app.PreprocesserApp --master yarn --deploy-mode cluster --driver-memory 2g /data/driverLib/geobeam.jar 10

出错：

diagnostics: Application application\_1527076886984\_0002 failed 2 times due to AM Container for appattempt\_1527076886984\_0002\_000002 exited with exitCode: 1

1. 尝试使用如下命令测试sparkPI，第一次失败 提示找不到文件 ，后修改文件路径成功

root@namenode1:/usr/local/spark-2.2.1# spark-submit --class org.apache.spark.examples.SparkPi --master yarn --deploy-mode cluster --driver-memory 2g examples/jars/spark-examples\_2.11-2.2.1.jar 10

1. 怀疑之前出错还是文件路径的问题，搜索如何指定文件读取路径为hdfs ，直接在提交过程中指定hdfs文件路径，结果还是一样，显示Exception in thread "main" org.apache.spark.SparkException: Application application\_1527076886984\_0013 finished with failed status错误，没法看到具体出现问题的位置
2. 接着尝试直接在standalone模式下运行，方便查看调试出现错误的位置

命令：spark-submit --class gtl.geospark.app.GeoSparkApp --master spark://10.186.28.101:7077 /data/driverLib/geospark.jar

这种模式运行出现错误：java.lang.NoClassDefFoundError: gtl/jts/geom/Envelope

查询NoClassDefFoundError产生原因以及解决办法：

<https://blog.csdn.net/jamesjxin/article/details/46606307>

1. 查看log：

<http://dongxicheng.org/mapreduce-nextgen/hadoop-logs-placement/>（logs位置）

yarn application -list -appStates ALL 查看所有的application记录

yarn logs -applicationId application\_1527076886984\_0014显示错误：

/tmp/logs/root/logs/application\_1527076886984\_0014 does not exist.

Log aggregation has not completed or is not enabled.

<https://blog.csdn.net/guohecang/article/details/52579270> （一般查看log方法）

Hadoop 2.x中YARN系统的服务日志包括ResourceManager日志和各个NodeManager日志，他们的日志位置如下：

ResourceManager日志存放位置是Hadoop安装目录下的logs目录下的yarn-\*-resourcemanager-\*.log

NodeManager日志存放位置是各个NodeManager节点上hadoop安装目录下的logs目录下的yarn-\*-nodemanager-\*.log

应用程序日志包括jobhistory日志和Container日志，其中，jobhistory日志是应用程序运行日志，包括应用程序启动时间、结束时间，每个任务的启动时间、结束时间，各种counter信息等。

Container日志包含ApplicationMaster日志和普通Task日志，它们均存放在Hadoop安装目录下的userlogs目录中的application\_xxx目录下，其中ApplicationMaster日志目录名称为container\_xxx\_000001，普通task日志目录名称则为container\_xxx\_000002，container\_xxx\_000003，….，同Hadoop 1.x一样，每个目录下包含三个日志文件：stdout、stderr和syslog，且具体含义是一样的。

1. 最新解决方案：

<https://stackoverflow.com/questions/38678151/how-to-solve-exception-in-thread-main-org-apache-spark-sparkexception-applic>

### 本周计划及进展：（2018.6.4-2018.6.10）

**计划：**

1. **学习十小时入门大数据：简单了解大数据中的一些概念比如安装配置，一些底层的架构设计以及每个架构的优缺点，学会如何使用官方文档，能够借助官方文档安装并运行每一个框架的wc案例**
2. **练习50道Java基础题+50道数据结构题，并每天做对应的总结，周六早上进行错题练习，巩固基础知识**
3. **Leetcode编程通过至少7道，并上传github，查看解析思考更多比较好的解法**
4. **在天河服务器运行简单的案例代码，搞明白在yarn-Cluster模式运行的整个流程**
5. **运行调试自己的代码，这里涉及到mvn编译（打包有带依赖的以及不带依赖的），运行对应的jar包**
6. **实现代码统计结果**

**6.4**

1. 测试PI：

spark-submit --class org.apache.spark.examples.SparkPi \

--master yarn \

--deploy-mode cluster \

--driver-memory 4g \

--executor-memory 2g \

--executor-cores 1 \

examples/jars/spark-examples\*.jar \

10

成功

1. 本地编译：

mvn clean package -DskipTests

1. 编译以后在target可以看到jar包然后拷到服务器（使用scp命令）**6.5**
2. 执行mvn安装：mvn clean install 看输出日志中显示安装的路径，学习如何使用别人的依赖进行开发

首先进行安装 然后在相应maven项目中添加依赖以及版本

1. 打包命令，如果在项目中的依赖只是本地测试使用，不用提交到服务器，那么可以把scope设置为provided，那么这个包就不会打进来
2. 另外想要直接把项目中添加的依赖打包，可以使用命令：mvn assembly:assembly，这时候可以在/target 目录下看到有一个相应的jar-with-dependencies
3. yarn.nodemanager.remote-app-log-dir and yarn.nodemanager.remote-app-log-dir-suffix 通过这里查看yarn的配置文件 可以找到 container log files的位置
4. 如果不能通过yarn logs查看，可以从/tmp/logs or $HADOOP\_HOME/logs/userlogs查看

### Flink实战：

1、官网下载安装包**<http://mirrors.shu.edu.cn/apache/flink/flink-1.5.0/flink-1.5.0-bin-hadoop27-scala_2.11.tgz>**

因为之前对应安装的hadoop版本是2.7.3，Scala版本是2.11.12

2、解压：

hadoop@slave02:~/Downloads$ sudo tar -zxvf flink-1.5.0-bin-hadoop27-scala\_2.11.tgz -C /opt/

3、进入到安装目录

4、./bin/start-cluster.sh启动

5、tail log/flink-\*-standalonesession-\*.log查看日志

6、http://localhost:8081/#/overview查看启动情况

7、操作：Command-Line Interface查看官网这个界面

<https://ci.apache.org/projects/flink/flink-docs-release-1.5/ops/cli.html> 里面有相关的操作案例

8、wordcount：

./bin/flink run ./examples/batch/WordCount.jar --input file:///home/hadoop/Documents/test.txt --output [file:///home/hadoop/Documents/wordcount\_out](file://home\\hadoop\\Documents\\wordcount_out) 成功

**6.6**

### Beam简单实践

1. 安装maven并配置环境变量
2. 直接命令行编译wordcount：

mvn archetype:generate \

-DarchetypeGroupId=org.apache.beam \

-DarchetypeArtifactId=beam-sdks-java-maven-archetypes-examples \

-DarchetypeVersion=2.4.0 \

-DgroupId=org.example \

-DartifactId=word-count-beam \

-Dversion="0.1" \

-Dpackage=org.apache.beam.examples \

-DinteractiveMode=false

可以显示build sucess

1. 运行wordcount(这里由于文档编辑的原因 为了保证成功 命令一条一条复制)

**#直接运行**

mvn compile exec:java \

-Dexec.mainClass=org.apache.beam.examples.WordCount \

-Dexec.args="--inputFile=/home/hadoop/Documents/test.txt --output=/home/hadoop/Documents/counts" -Pdirect-runner

**#flink cluster运行**:

mvn package exec:java \ -Dexec.mainClass=org.apache.beam.examples.WordCount \ -Dexec.args="--runner=FlinkRunner

--flinkMaster=<http://192.168.0.131:8081> \ --filesToStage=target/word-count-beam-bundled-0.1.jar \

--inputFile=/home/hadoop/Documents/test.txt \

--output=/tmp/counts" -Pflink-runner

**#flink-local运行**

mvn compile exec:java -Dexec.mainClass=org.apache.beam.examples.WordCount \

-Dexec.args="--runner=FlinkRunner --inputFile=pom.xml --output=counts" -Pflink-runner

运行完成可以在指定的路径看到count文件 使用more/less命令可以查看文件的内容

参考：

<https://beam.apache.org/get-started/quickstart-java/>

**学习SparkSQL**

1. 编译spark source code 使用

make-distribution.sh 或者mvn编译

**6.7**

1、10道Java基础题，更深入的掌握了++i 和i++的区别 ，学习了异常的捕获可以在调用其的函数中使用 ，学习了jsp的生命周期

1. 学习了Java线性表的原理以及代码实现，阅读Java中arrayList的源码

**Spark环境实战**

1. local环境搭建：直接把官网的安装包解压 然后运行spark-shell --master local[\*]即可
2. 集群搭建 以及运行 master改为Spark的网址

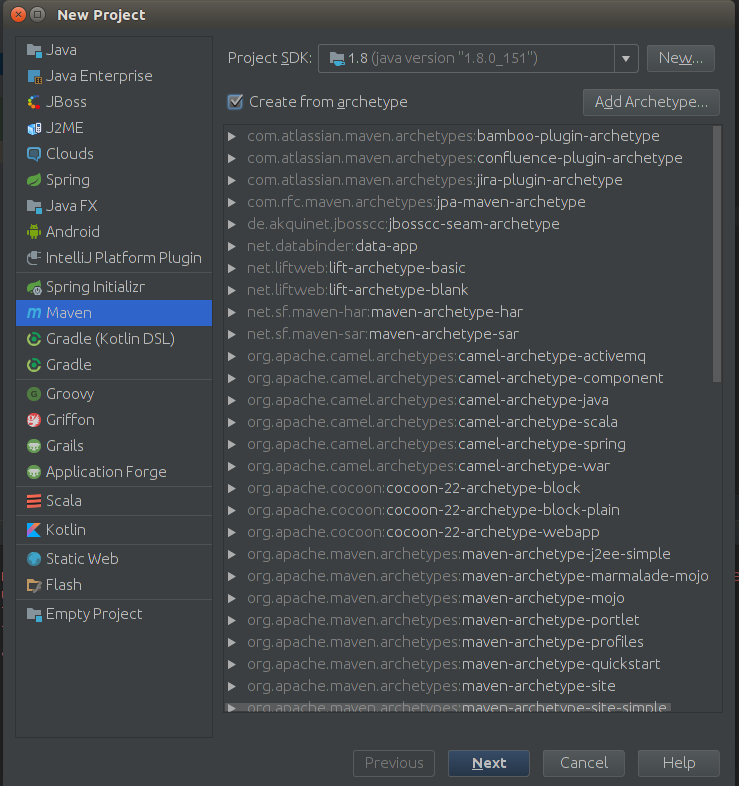
spark-shell --master spark://slave02:7077

SparkSQL学习

1. Spark SQL is Apache Spark's module for working with **structured data**.
2. SparkSQL不局限与SQL
3. 可以访问hive json 等文件
4. 提供了sql的API以及dataframe、dataset的API

**测试案例一：运行SparkSQL--SQLcontext**

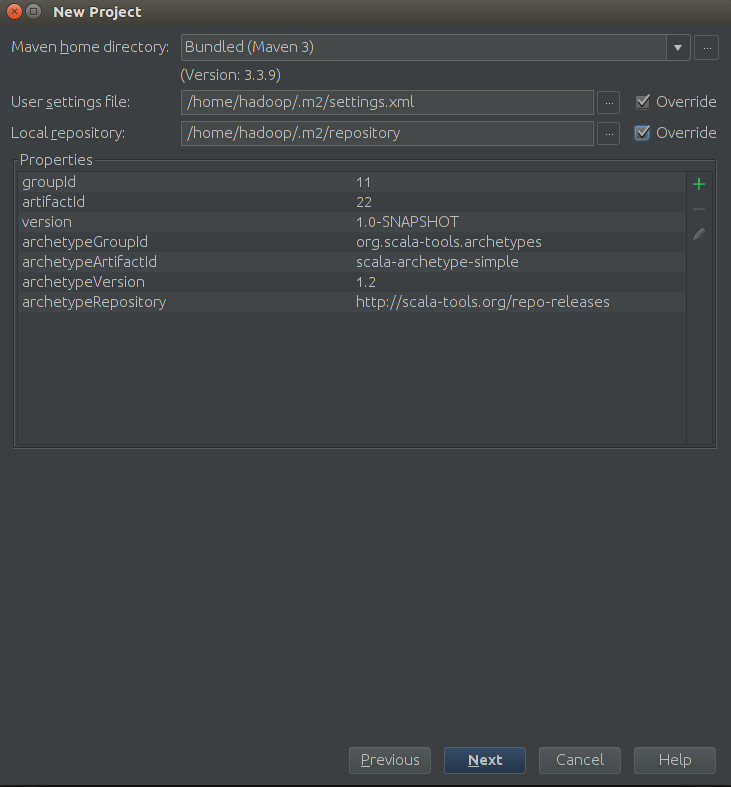
1、新建maven项目



这里需要勾选create from archetype

然后在下面列表中选择 a simple scala project

1. 填写GVA
2. 设置项目属性



这里需要注意填写setings.xml需要是你maven的安装路径

比如我的是：/opt/apache-maven-3.5.3/conf/settings.xml

其他设置按照个人喜好

具体的总结见个人博客：

<https://blog.csdn.net/yabosun/article/details/80615939>

**6.8**

1. 完成10道Java基础练习，总结了maven以及ant的区别，总结了抽象类以及接口的区别
2. 实现了在集群运行SparkContext测试SparkSQL，具体过程见个人博客

<https://blog.csdn.net/yabosun/article/details/80618153>

3、

**测试案例二：运行SparkSQL--HIVEContext**

To use a HiveContext, you do not need to have an existing Hive setup

这部分因为没有配置hive相关 就不做深入的实践，但是测试了整个的流程 由于测试的过程中无法读到hive中的emp表所以无法显示最终的结果

**测试案例三：运行SparkSQL--SparkSession（2.x推荐使用）**

基本的定义格式：

**val** spark **=** **SparkSession**

.builder()

.appName("Spark SQL basic example")

.config("spark.some.config.option", "some-value")

.getOrCreate()

参考：

[http://spark.apache.org/docs/2.1.0/sql-programming-guide.html#starting-point-sparksession](http://spark.apache.org/docs/2.1.0/sql-programming-guide.html" \l "starting-point-sparksession)

具体总结见博客：

<https://blog.csdn.net/yabosun/article/details/80624753>

1. spark-shell&spark-sql使用

spark-shell --master local[2]

spark-sql --master local[2]

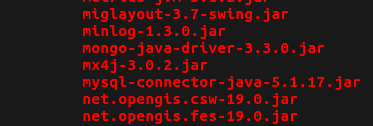
条件：需要将hive-site.xml的配置文件放置在spark的conf中，需要传递mysql驱动包

1. Thriftserver/beelin休息e的使用

参考：

[http://spark.apache.org/docs/latest/sql-programming-guide.html#running-the-thrift-jdbcodbc-server](http://spark.apache.org/docs/latest/sql-programming-guide.html" \l "running-the-thrift-jdbcodbc-server)

因为需要连接jdbc，所以都是需要在这之前配置jdbc-connector，直接把这个驱动放在spark的jars目录下



可以看到有mysql-connector-java-5.1.17.jar

**Thriftserver使用**

接下来开始在本地进行运行 输入：

root@namenode1:/usr/local/spark-2.2.1/sbin# ./start-thriftserver.sh --master local[2]

查看jps进程：

root@namenode1:/usr/local/spark-2.2.1/sbin# jps -m

多了一个这样的进程：

58529 SparkSubmit --master local[2] --class org.apache.spark.sql.hive.thriftserver.HiveThriftServer2 --name Thrift JDBC/ODBC Server spark-internal

**Beeline使用**

输入：

root@namenode1:/usr/local/spark-2.2.1# ./bin/beeline -u jdbc:hive2://localhost:10000 -n root

**对比：thriftserver和spark-shell、spark-sql**

1. spark-shell、spark-sql都是每次启动对应一个spark application
2. Thriftserver是不管你启动多少个客户端（beeline、code），永远都是一个spark application，解决了一个数据共享的问题，多个客户端可以共享数据

### 本周计划及进展：（2018.6.11-2018.6.17）

1. **练习50道Java基础题+50道数据结构题，并每天做对应的总结，周六早上进行错题练习，巩固基础知识**
2. **把Java主要的几个特性的使用仔细看一遍，包括collection、list、set、map等集合的使用**
3. **看大话数据结构，自己实现至少俩类的算法，先把数据结构中数组和线性表做一个详细的学习**
4. **学习大数据最后一个系列，SparkStreaming的相关知识，并进行实际的测试以及总结实践文档**
5. **将导师的项目按照之前的做出结果并总结**
6. **学习Javaweb的项目**
7. **学习数据库MySql**

**6.11-6.13**

1. 学习mysql
2. Nowcoder错题总结
3. **GeoSpark运行成功：**

将代码修改，

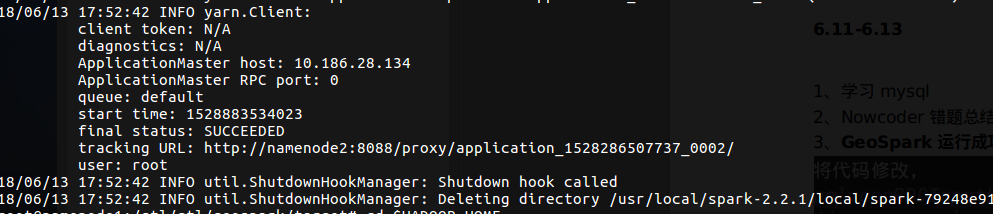
*polygonRDDInputLocation* = args[0];  
// polygonRDDInputLocation = "/home/hadoop/Documents/data/spatialhadoop/counties/county\_small.tsv";  
// polygonRDDInputLocation = "hdfs://geosciences/data/spatialhadoop/counties/county\_small.tsv";

将最初的绝对路径读取本地，以及绝对路径读取hdfs修改为从外部传入参数，然后在shell命令中指定文件的读取路径，

root@namenode1:/gtl/gtl/geospark/target# spark-submit --class gtl.geospark.app.GeoSparkApp --master yarn --deploy-mode cluster --driver-memory 4g --executor-memory 2g geospark-1.0-SNAPSHOT.jar hdfs://geosciences/data/spatialhadoop/counties/county\_small.tsv 10

这里注意项目jar包的路径

运行显示成功：



**6.14**

1. 学习SparkStreaming，主要进行了Flume在集群上环境的搭建，并实现了俩个案例的操作
2. 总结错题

**6.15**

1. 总结nowcoder错题，包括Java内联函数、run方法和start方法、自定义异常、Volatile关键字作用、线程安全的map实现、JSP内置对象及方法以及ThreadLocal类
2. 实现Flume第三个案例
3. 学习Kafka

端午------------

### 本周计划及进展：（2018.6.19-2018.6.24）

1. 学习数据库操作，主要是MySQL数据库，进行简单的实践
2. 学习完SparkStreaming
3. 编程练习5道
4. 学习JavaEE视频教程，在Ubuntu上搭建相关编程环境

主要就是学习了一些mysql数据库

SparkStreaming视频学习完整个Kafka的搭建包括使用相关JavaAPI

### 本周计划及进展：（2018.6.25-2018.7.1）

1. \*\*\*每日坚持做Java基础练习10道
2. \*\*每天学习一个数据结构视频（学习完以后每天刷10道对应的基础练习）
3. \*\*\*每天做一个leetcode
4. \*\*\*\*\*将对应的mysql报告截图写完，并审核学生的电子报告
5. \*\*继续深入学习SparkStreaming
6. \*\*\*\*\*\*完成老师分配的项目任务

6.25

1. 做10道Java基础，调试电脑机器
2. 修改批阅俩个班第一次的课程报告

6.26

1、

6.28

1. 阅读GeoBeam文档，总结文中技术创新点以及重点部分
2. 做Java基础题
3. 了解beam的整体数据处理流程，阅读部分beam源码
4. 急需好好学习一下正则表达式 latex

6.29

1. 练习Java基础题10道，总结JDBC的知识点以及sleep和wait的区别
2. 学习了正则表达式，简单的进行demo练习
3. 学习Java中几个集合框架包括collection、Map等
4. 学习mapreduce操作

6.30

1、

### 本周计划及进展：（2018.7.2-2018.7.8)

1. **\*\*\*每日坚持做Java基础练习10道**
2. **\*\*\*批改第二次课程报告**
3. **专业英语课程报告完成**
4. **编程题5道**
5. **数据库MySQL深入学习**

**6、复习巩固错题**

7.2

1. 将俩个班的第一次报告批改完成
2. 练习剑指offer编程题目---找到不重复的数

7.3

1. 将俩个班的第二次报告批改完成

7.4

1. 将专业英语报告完成，学习了latex的部分操作

包括标题、字体、表格、图片等的排版以及插入方式

7.5

1、10道基础题 总结静态变量、变量、常量的区别

2、批改报告

### 本周计划及进展：（2018.7.9-2018.7.15）

1. 每天10道基础
2. Leetcode每天俩道 了解基础数据结构
3. 学习数据库系统概论后面几章
4. 学习MySQL高性能知识
5. 阅读geobeam源码
6. 阅读Java核心技术 做对应练习

### 暑假任务

1. 每天至少一道leetcode(完成)
2. 学完数据库系统概论 (完成)
3. 将之前的视频重新看完（3个实战）
4. 开始阅读Spark源码
5. 阅读gtl源码
6. 学习linux

上周总结：

1. leetcode刷题6道
2. 每天做10道Java基础
3. 大致学完数据库前7章，并实验测试数据库连接

### 暑假日常

1. 每天一道leetcode，10道Java语言基础（40分钟）
2. 一章大话数据结构（60分钟）
3. 半小时《Hadoop The Definitive Guide》（30分钟）
4. 一个大数据视频（90分钟）
5. 学半小时Linux（30分钟）
6. 总结日报（15分钟）

（总计： 265分钟 ～ 4.5h）

### 本周计划及进展：（2018.7.16-2018.7.22）

1. 继续基础知识
2. 每天一道leetcode
3. 学完数据库后面部分 9、10还有11章
4. 开始学习linux 大话数据结构
5. 配合看视频

7.17.2018

1. 实现并上传leetcode一道
2. 写Java基础一道
3. 完成7章数据库答案编写
4. 批改学生报告3份，整理报告成绩

7.18.2018

1. Java基础10道 并总结错题
2. Leetcode 122
3. 俩个班报告

7.19.2018

1. java基础10道
2. Leetcode 122优化
3. 实验教程写5章
4. 所有课程设计报告登完
5. 写一篇数据库备份恢复博客

### 本周计划及进展：（2018.7.23-2018.7.29）

1. 将数据库参考答案完善(完成)
2. Hdfs3.0新特性，主要是调研一下对于数据的存储是否还是只能进行后续插入，是否可以进行随机的数据插入，写总结调研文档（完成）
3. 参照项目中的DiskStorageManager，实现HdfsStorageManager，主要是索引文件还有数据文件的读写（参考accessfile）
4. 每天一道leetcode
5. 继续Java基础知识 每天10道
6. 每天看一章大话数据结构 俩周把这部分知识看完 基础了解
7. 关系查询处理和查询优化、数据库恢复技术、并发控制学习

2018.7.23

1. 写总结博客 将vs连接数据库的操作写完

2018.7.25

1. 调研hdfs3.0新特性

主要特性还是增加了多namenode，对于文件的操作还是只允许append，无法实现随机存取操作，但是使用hbase是可以实现

<http://aajisaka.github.io/hadoop-project/>

是否可以将文件操作管理按照类似hbase进行增删改查？？？

在HDFS的一个文件中只有一个写入者，而且写操作只能在文件末尾完成，即只能执行追加操作。目前HDFS还不支持多个用户对同一文件的写操作，以及在文件任意位置进行修改

1. 学习RandomAccessFile使用，

<https://blog.csdn.net/Lirx_Tech/article/details/51234056>

A、RandomAccessFile其实是一个文件I/O流，用文件名构造后可以直接对文件进行读写不需要用流进行包装。

B、其最大的一个特点就是可以自由访问文件的任意一个位置（即随意移动位置指针），这对于正常的I/O流是不可思议的，首先输出流不允许移动位置指针，而输入流最多只能向前略过几个位置而已

1. 阅读DiskStorageManager实现代码

这个过程中首先了解索引文件是什么？

索引文件指的是为逻辑记录和物理记录之间建立一个一一对应关系表–索引表

1. 实现hdfs文件系统操作

思路：

先实现对于普通文件内容的读写、删除、flush、关闭等操作

1. 操作系统物理内存管理分页操作

<http://airtrack.me/posts/2015/04/27/%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F%E5%AE%9E%E7%8E%B0%EF%BC%88%E4%BA%8C%EF%BC%89%EF%BC%9A%E5%88%86%E9%A1%B5%E5%92%8C%E7%89%A9%E7%90%86%E5%86%85%E5%AD%98%E7%AE%A1%E7%90%86/>

### 本周计划及进展：（2018.7.30-2018.8.5）

1. 阅读关于索引的文献，从最开始的平衡树开始，比如B+树，使用Java进行实现，这部分涉及到数据结构的学习使用
2. 调研其他分布式文件系统，主要了解针对HDFS的不可写文件有没有改进
3. 深入理解HBase的存储实现，对于文件构建索引的实现方法
4. 阅读hdfs源码，进行相关分析，看能否结合HBase实现自己的文件系统
5. 查阅其他文献 包括spatialGIS等，对比别人是如何实现随机的文件修改
6. 初步想法是在存储过程中将索引文件和数据文件分开存放，比如将原始数据文件存放到hdfs，而将索引文件存放在内存
7. 继续阅读GFS论文
8. 抽时间阅读hdoop权威指南 以及数据算法

个人计划：

每天保证一道算法题，并且10道基础练习

看一章大话数据结构

学习大数据视频

2018.8.1

1. 阅读文献：

MapReduce: Simplified Data Processing on Large Clusters

The Google File System

Google File System 中 文 版

1. 阅读Hadoop The Definitive Guide第一章
2. 阅读博客从B树、B+树、B\*树谈到R 树

2018.8.2

1、每天一道leetcode，10道Java语言基础（40分钟）（完成）！！！

2、一章大话数据结构（60分钟）(完成)！！！

3、半小时《Hadoop The Definitive Guide》（30分钟）（完成）！！！

4、一个大数据视频（90分钟）(完成)！！！

5、学半小时Linux（30分钟）（完成）！！！

6、总结日报（15分钟） （完成）！！！

（总计： 265分钟 ～ 4.5h）

1. 看完大话数据结构--线性表，需要更深入理解实现
2. 继续阅读博客从B树、B+树、B\*树谈到R 树
3. 学习推荐系统，主要是知道了做一个推荐系统的整体架构，包括对于数据的采集-数据的存储-数据的过滤等等流程，这部分抽象出来主要是三层，包括online层、offline层、nearline层，重点关注的是offline以及nearline层，个人理解online层就是不同的系统来如何获取用户的数据，offline层包括像mapreduce等计算，nearline也就是实时层，有storm、spark streaming等技术
4. 学习MongoDB明天学习HBase，觉着这俩个有关联 设计思路
5. 学习Linux第0章，简单了解计算机的底层组成，知道四大组成部分，还有就是用于数据传输的俩个主板芯片组，以及为什么内存传输速度快于磁盘
6. 阅读《Hadoop The Definitive Guide》主要学习mapreduce，主要分为input-map-shuffle-reduce-output过程，数据获取网站：<https://www.ncdc.noaa.gov/> 天气数据 卫星数据等
7. 完成10道Java基础，wait进入等待锁定池，只有针对此对象发出notify方法获得对象锁进入 “就绪” 状态；InputStreamReader可以处理 Unicode；
8. 完成leetcode 448编程，主要学习别人对于in-place操作，以及如何灵活使用arraylist，对于数组的简单操作，因为是【1-n】，找到中间不存在的数，所以可以通过将不存在的数想办法放在对应的index上，然后直接将index的值add到arraylist，很奇特的想法

2018.8.3

1、每天一道leetcode，10道Java语言基础（40分钟）（完成）！！！

2、一章大话数据结构（60分钟）（完成）！！！

3、半小时《Hadoop The Definitive Guide》（30分钟）

4、一个大数据视频（90分钟）（完成）！！！

5、学半小时Linux（30分钟）（完成）！！！

6、总结日报（15分钟） （完成）！！！

（总计： 265分钟 ～ 4.5h）

1. Document-oriented database：面向文档的数据库

翻译自维基百科

**known as [semi-structured data](https://en.wikipedia.org/wiki/Semi-structured_model" \o ").**实质上就是半结构化的数据存储，这部分一般是NoSQL数据库

Relational databases generally store data in separate *tables* that are defined by the programmer, and a single object may be spread across several tables. Document databases store all information for a given object in a single instance in the database, and every stored object can be different from every other.

关系数据库将数据存储在程序员定义的独立的表中，其中的数据对象通过几张表来进行使用，而面向文档的数据库是将所有对象的信息存储在一个单独的数据库实体中，并且每个存储的对象彼此都不相同

This key is a simple [identifier](https://en.wikipedia.org/wiki/Identifier" \o "Identifier) (or ID), typically a [string](https://en.wikipedia.org/wiki/String_(computer_science)" \o "String (computer science)), a [URI](https://en.wikipedia.org/wiki/URI" \o "URI), or a [path](https://en.wikipedia.org/wiki/Path_(computing)" \o "Path (computing))。Typically the database retains an [index](https://en.wikipedia.org/wiki/Database_index" \o "Database index) on the key to speed up document retrieval, and in some cases the key is required to create or insert the document into the database

文档在数据库中是以一个特殊的key来存储的，这个key通常是一个简单的identifier，或者id，典型的有一个字符串，一个url或者一个路径，这个key用来从数据库中检索数据，通常数据库会在key上保留一个索引用于更快速的进行文档检索，并且很多时候这个key用于将文档创建或者插入数据库中

Document databases typically provide some mechanism for updating or editing the content (or other metadata) of a document, either by allowing for replacement of the entire document, or individual structural pieces of the document.

文档数据库提供了一些对于文档更新或者编辑的操作，比如对于整个文档进行替换或者仅仅对于文档的部分进行修改

For example, an address book application will generally need to store the contact name, an optional image, one or more phone numbers, one or more mailing addresses, and one or more email addresses. In a canonical relational database, tables would be created for each of these rows with predefined fields for each bit of data: the CONTACT table might include FIRST\_NAME, LAST\_NAME and IMAGE columns, while the PHONE\_NUMBER table might include COUNTRY\_CODE, AREA\_CODE, PHONE\_NUMBER and TYPE (home, work, etc.). The PHONE\_NUMBER table also contains a foreign key column, "CONTACT\_ID", which holds the unique ID number assigned to the contact when it was created. In order to recreate the original contact, the database engine uses the foreign keys to look for the related items across the group of tables and reconstruct the original data.

比如对于一个电话簿，包括姓名、相片、电话、邮箱，对于传统的关系数据库，应该会设计几张表，包括contact表，其中有first\_name,last\_name,image等属性列，而还有phone\_number表，其中包括country\_code,area\_code,phone\_number以及type等，电话号码表应该包含一个外键，contact\_id等等

In contrast, in a document-oriented database there may be no internal structure that maps directly onto the concept of a table, and the fields and relationships generally don't exist as predefined concepts. Instead, all of the data for an object is placed in a single document, and stored in the database as a single entry. In the address book example, the document would contain the contact's name, image, and any contact info, all in a single record. That entry is accessed through its key, which allows the database to retrieve and return the document to the application. No additional work is needed to retrieve the related data; all of this is returned in a single object

而相反地，对于面向文档的数据库，并没有提前定义的一些概念，比如表，域以及关系等，取而代之的是所有对象的数据都是存储在一个单一的文档中，并且所有的都以一个entry的形式存储在数据库中，比如对于以上的电话簿，文档会包含联系姓名，照片，以及其他任何的连心信息，所有的信息都是在一个单一的记录中，以一个entry的形式，这个entry通过他的key来进行操作。包括检索等

1. 二叉查找树（BST）、B树（B-tree）、B+树、B\*树、AVL树、红黑树（**TODO**）

参考：

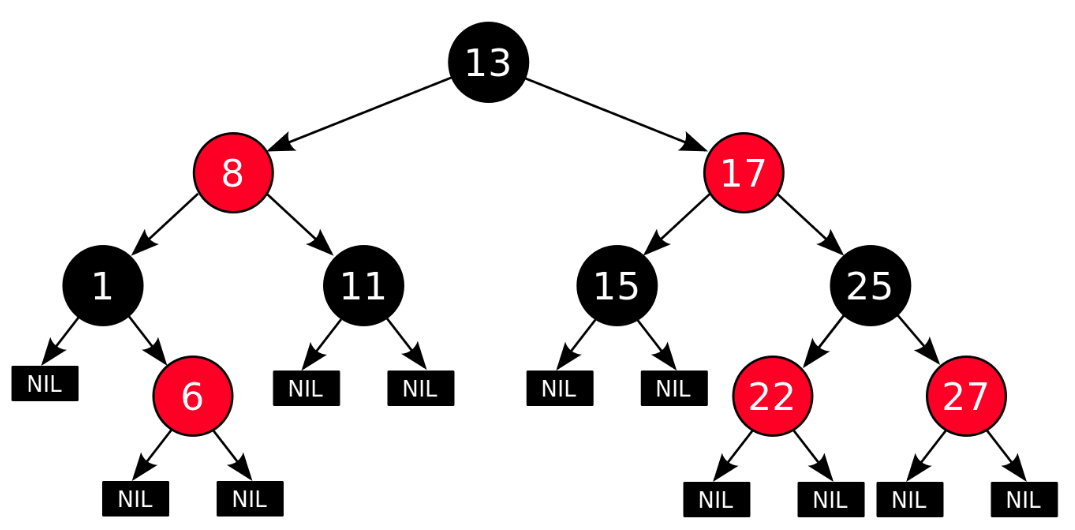
[https://blog.csdn.net/v\_july\_v/article/details/6105630](https://blog.csdn.net/v_july_v/article/details/6105630#comments)

<https://blog.csdn.net/v_JULY_v/article/details/6530142>

*B-树，即为B树。因为B树的原英文名称为B-tree，而国内很多人喜欢把B-tree译作B-树，其实，这是个非常不好的直译，很容易让人产生误解。如人们可能会以为B-树是一种树，而B树又是一种一种树。而事实上是，******B-tree就是指的B树*****

红黑树：

根节点永远是黑色的；节点要么是黑色，要么是红色；一条路径上不能有俩个相邻的红节点，也就是说，红色节点不能有红色父节点或红色子节点；从一个节点到达一个叶子节点都应该有相同的黑色节点，称为这条路径的黑高度



1. Mapreduce

主要跟着视频进行mapreduce的一个初步的了解，首先了解了传统的对于海量数据的分布式处理技术，都是采用分而治之的思想，传统的主要是涉及到hash的思想，通过hash流量分发来将数据请求进行分发到不同的服务器进行处理，另外一种模式是通过一致性hash方法，也就是形成环，可以将一台服务器按照性能分为几个不同的虚拟服务器，这个支持动态增长；

而mapreduce这部分的介绍最形象的就是通过一个数钱的案例来学习分治的思想

1. Linux学习

学习第一章 Linux是什么，首先总体了解了组成的结构，包括内核以及系统调用组成操作系统

另外了解了Linux的发展历史，从Unix开始，包括最原始的Multics的失败，以及后面使用汇编语言Unics的产生，到最后1973年Unix的正式诞生，通过C语言基于原有的Unics，这之后伯克利大学BSD的产生，以及后面产生的Minix，更重要的有GNU项目，开发GCC（Gnu C Compiler），了解了自由软件版权（GPL），也是GNU的项目，到后来就是Linus开发Linux系统的诞生，是基于Minix进行的改进开发，实现多任务，后面学习了Linux的后续版本，以及内核版本分为开发版以及稳定版，学习现有的一些基于Linux内核开发的操作系统，最后学习了一些优缺点

2018.8.5

总结HBase、bigtable存储过程

### 本周计划及进展：（2018.8.6-2018.8.12）

1. 继续刷日常（4h）

{

1、每天一道leetcode，10道Java语言基础（40分钟）

2、一章大话数据结构（60分钟）

3、半小时《Hadoop The Definitive Guide》（30分钟）

4、一个大数据视频（90分钟）

5、学半小时Linux（30分钟）

6、总结日报（15分钟）

（总计： 265分钟 ～ 4.5h）

}

1. 阅读BigTable论文
2. 学习索引B+、B\*树，结合一篇中文论文；阅读gtl中关于index的代码
3. 继续深入学习HBase，通过在集群安装并跑案例学习一下
4. 深入了解一下CEPH
5. 了解一个类似的分布式处理空间数据是对于数据存储、读写是如何处理，比如SpatialGIS等

2018.8.6

{

1、每天一道leetcode，10道Java语言基础（40分钟）（完成）！！！

2、一章大话数据结构（60分钟）(完成)！！！

3、半小时《Hadoop The Definitive Guide》（30分钟）

4、一个大数据视频（90分钟）

5、学半小时Linux（30分钟）

6、总结日报（15分钟） （完成）！！！

（总计： 265分钟 ～ 4.5h）

}

1. 学习B树原理
2. 整理上周学习情况写总结PPT、文档
3. 阅读《大话数据结构》第五章

2018.8.7

{

1. 每天一道leetcode，10道Java语言基础（40分钟）

2、一章大话数据结构（60分钟）

3、半小时《Hadoop The Definitive Guide》（30分钟）

4、一个大数据视频（90分钟）

5、学半小时Linux（30分钟）

6、总结日报（15分钟）

（总计： 265分钟 ～ 4.5h）

}

2018.8.7

{

1. 每天一道leetcode，10道Java语言基础（40分钟）

2、一章大话数据结构（60分钟）

3、半小时《Hadoop The Definitive Guide》（30分钟）（完成）！！！

4、一个大数据视频（90分钟）

5、学半小时Linux（30分钟）（完成）！！！

6、总结日报（15分钟）

（总计： 265分钟 ～ 4.5h）

}

1. Map tasks write their output to the local disk, not to HDFS .

<Hadoop The Definitive Guide> 31页

1. 学习Linux简单了解了如何学习《鸟哥私房菜 第二章》

-------------------------------------请假休息-----------------------------------------------

### 本周计划及进展：（2018.8.13-2018.8.19）

1、继续刷日常（4.5h）

{

1、每天一道leetcode，10道Java语言基础（40分钟）

2、一章大话数据结构（60分钟）

3、半小时《Hadoop The Definitive Guide》（30分钟）

4、一个大数据视频（90分钟）

5、学半小时Linux（30分钟）

6、总结日报（15分钟）

（总计： 265分钟 ～ 4.5h）

}

2、阅读BigTable论文

3、学习索引B+、B\*树，结合一篇中文论文；阅读gtl中关于index的代码

4、继续深入学习HBase，通过在集群安装并跑案例学习一下

5、深入了解一下CEPH

6、了解一个类似的分布式处理空间数据是对于数据存储、读写是如何处理，比如SpatialGIS等

2018.8.13

1、继续刷日常（4.5h）

{

1、每天一道leetcode，10道Java语言基础（40分钟）（完成）！！！

2、一章大话数据结构（60分钟） （完成）！！！

3、半小时《Hadoop The Definitive Guide》（30分钟）(完成)！！！

4、一个大数据视频（90分钟） （完成）！！！

5、学半小时Linux（30分钟）（完成）！！！

6、总结日报（15分钟） （完成）！！！

（总计： 265分钟 ～ 4.5h）

}

2、主要学习mapreduce相关知识，通过使用Python简单的写了一个mapreduce的案例，需要进一步学习

3、

2018.8.14

1、继续刷日常（4.5h）

{

1、每天一道leetcode，10道Java语言基础（40分钟）

2、一章大话数据结构+数据结构与算法视频（60分钟）（完成！！！）

3、半小时《Hadoop The Definitive Guide》（30分钟）

4、一个大数据视频（90分钟）

5、学半小时Linux（30分钟）

6、总结日报（15分钟）

（总计： 265分钟 ～ 4.5h）

}

2、学习HBase，主要学习了列式存储与行式存储的区别以及列式存储的优势，为后面更好的理解HBase以及BigTable

2018.8.15

1、继续刷日常（4.5h）

{

1、每天一道leetcode，10道Java语言基础（40分钟）

2、一章大话数据结构（60分钟）

3、半小时《Hadoop The Definitive Guide》（30分钟）

4、一个大数据视频（90分钟）

5、学半小时Linux（30分钟）

6、总结日报（15分钟）

（总计： 265分钟 ～ 4.5h）

}

1. 通过leetcode学习多维数组的操作，与一维数组的转化等

----------------------------------------颓废了俩天------------------------------------------------------

2018.8.17

1. 找hbase视频学习
2. 刷日常

### 本周计划及进展：（2018.8.20-2018.8.26）

1、继续刷日常（4.5h）

{

1、每天一道leetcode，10道Java语言基础（逛论坛 总结别人的面试题 发博客）（75min）（完成！！！）

2、一章大话数据结构（60分钟）

3、半小时《Hadoop The Definitive Guide》（30分钟）

4、一个大数据视频（90分钟）

5、学半小时Linux（30分钟）

6、总结日报（15分钟）

（总计： 265分钟 ～ 4.5h）

}

1. 找HDFS+index文献 并创建文献库

### 本周计划及进展：（2018.8.27-2018.9.2）

1、继续刷日常（4.5h）

{

1、每天一道leetcode，10道Java语言基础（逛论坛 总结别人的面试题 发博客）（75min）（完成！！！）

2、一章大话数据结构（60分钟）

3、半小时《Hadoop The Definitive Guide》（30分钟）

4、一个大数据视频（90分钟）

5、学半小时Linux（30分钟）

6、总结日报（15分钟）

（总计： 265分钟 ～ 4.5h）

}

2、翻译论文

2018.8.27

1. 翻译论文第一章
2. Leetcode605
3. 10道Java基础

2018.8.28

1. 翻译论文第二章
2. 看了三个flink以及kylin的开源社区视频

2018.8.29

1. 总结之前GeoSpark博客
2. 继续翻译论文第二章、第三章

2018.8.30

1. 学习SparkStreaming
2. 修改gtl/geom代码

2018.8.31

1. 任务分配：

龙：IndexedPolyline（基于点和索引的表示方法）、Texture

赵：IndexedPolygon（基于点和索引的表示方法）、Material

孙：TriangleMesh、IndexedSolid（基于点和索引的表示方法）、Solid

### 本周计划及进展：（2018.9.3-2018.9.9）

1. 完成导师任务，跟进师弟任务完成情况
2. 继续完成SparkStreaming项目
3. 刷leetcode、nowcoder
4. 将论文翻译

2018.9.3

1. 新建要完成的三个类TriangleMesh、IndexedSolid（基于点和索引的表示方法）、Solid
2. 先完成TriangleMesh以及Solid

2018.9.4

1. 学习SparkStreaming项目，整理了前面俩章的博客
2. 刷leetcode

2018.9.5

1. 修改Geometry基类，主要添加数据类型包括Solid、IndexedSolid、TriangleMesh分别为8、9、10，添加相应的创建函数，在相应的类中添加构造函数等

2018.9.6

1. 继续学习SparkStreaming

2018.9.9

1. 完成Kafka以及Flume的整合简单案例实战
2. 改写代码：主要是先看一下原有的Polygon是如何改写，然后照着改一下

### 本周计划及进展：（2018.9.10-2018.9.16）

1. 完成之前布置的七个函数功能
2. 继续学习SparkStreaming
3. 学习Python着手搞项目
4. 写leetcode

2018.9.10

1. 继续学习SparkStreaming第五章第六章
2. 实现Texture

2018.9.13

1. 任务安排：

龙：完善IndexedPolyline，实现TriangleMesh

赵：完善后面的俩个

2018.9.14

1. 学习单元测试
2. 写indexPolyline clone()、getByteArraySize()
3. IndexedPolyline类所有都实现完成

2018.9.15

1. 实现clone（）的一个更一般化的方法