

****

基于订单起止点的乘车需求与调度方案分析

组员：

16211018 许鸿智

16211016 蔡代钢

16211084 蒋予飞

16211083 徐子一

### 一、整体目标说明

本次大数据实践课程滴滴大数据分析部分我们组的题目为基于订单起止点的乘车需求分析与调度规划。我们的主要目标是通过分析订单的起点数据得到区域内乘客的乘车需求分布情况。并通过分析订单终点数据得到区域内司机分布情况，再通过高效二维平面匹配（spatial matching）算法得到最有匹配，即最优调度方案。当然，由于不同时段用户乘车需求、司机分布差别很大。所以我们的分析结果自然都是分时段的。

根据滴滴给出的gps点及时间戳数据，我们可以通过时间戳获取订单起点与终点，行程起点意味着乘客上车点，它可以很好地反应用户的乘车需求。按照一般的常识规律，司机在一个订单结束后，出于最大化收益的考虑，一般会考虑在行程终点附近继续接单。因此终点信息可以很好的反映司机的分布情况。通过这两个分布我们可以通过高效的二维空间匹配算法得到司机与乘客的匹配关系，在大数据分析的结果上看可以为滴滴平台提供每个时段的分布信息与司机调度推荐。

### 二、实验方案与分析

正如上一部分我们整体实验目标所说明的，我们的任务大体上分为两个部分，在这里我们称其为乘车需求分析和调度规划，两者是递进关系，实验整体的方案可以用“数据提取、数据清洗、数据分析”的流程来概括。下面将分两个任务按这四部分阐述更精确的方案：

#### 2.1乘车需求分析方案

数据提取：上文已经提到过，要分析乘车需求必须要考虑订单起点的分布。因此我们第一步是对起点数据进行有效的提取，提取的方法也比较简单，是对原数据的所有路径点先按照主订单号、次关键字时间戳做排序。排序后的数据将满足这样的特征：所有点按照订单分开，且顺序是时间戳由小到大，即起点到终点的排列。之后我们只需要按订单分组提取第一条数据即可得到所有的订单起点数据。上文还提到需求分布是与事件相关的，因此分析一整天的数据对现实的知道意义较小，因此我们选择了分时段进行分析的方法。因此我们得到起点数据之后进行按时间分类的操作，将一天的起点数据根据时间戳转换得到的北京时间每小时分为一个时段，因此我们得到了24个每小时的分段起点数据。

数据清洗：我们得到的起点数据不易受到GPS漂移产生的影响，因为我们研究的是分布规律，因此有极少量的点便宜不会对整体的分布规律产生影响。但是我们在对空间点的分析时发现周围地区的起点分布异常偏高，我们对这个现象分析得到的结论是存在大量“假起点”，因此需要对数据进行“削边”操作。这部分具体的分析与操作会在后文遇到的问题部分中详细展开。

数据分析：研究起点的分布规律最直观的方法是绘制热力图。我们在得到每小时的数据分析结果后我们就可以通过调用百度地图的API绘制叠加到真实地图上的热力图。通过热力图展示的信息我们可以直观看到需求分布情况。并且因为我们每小时绘制了一张热力图，因此通过对比热力图我们可以得到一天内整个取样地区的乘车需求变动情况。

#### 2.2调度规划方案

数据提取：基本与 2.1乘车需求分析方案一致，不同点在于这次不仅需要提取起点数据还需要提取终点数据。终点数据的提取方法与起点数据类似，只需要取订单路径上最后一个点就可以了。同样也需要对数据进行分时段提取。

数据清洗：与乘车需求分析类似，也存在“假终点”的问题，也需要执行削边操作。

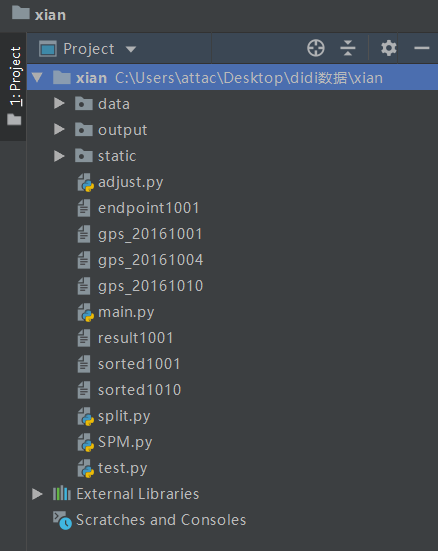
数据派生：我们需要对数据进行一个派生操作，因为即便是大数据的结果，散列而具体的点对进行匹配得到的调度方案未免太过细致，现实指导意义不大。因此我们必须要提取区域特征，因此我们在区域内以更低经度的经纬度格点作为网格点，将数据中的起点终点转化为与其最邻近的网格点的权值。这样得到的调度计划就是以网格点为代表的区域间的调配方案，更具现实指导意义。

数据分析：这一部分的数据分析较为繁琐，是本次实践大作业的重点。这一部分我们的基本任务是根据起点和终点的分布，即需求与司机供给的分布，完成二维平面上的匹配算法得到调度方案。

### 三、实验过程

预先说明：使用2016年10月1日全天数据进行分析，实际上并不严谨，在算力和时间允许的情况下应该使用更多时段数据更具普适性。

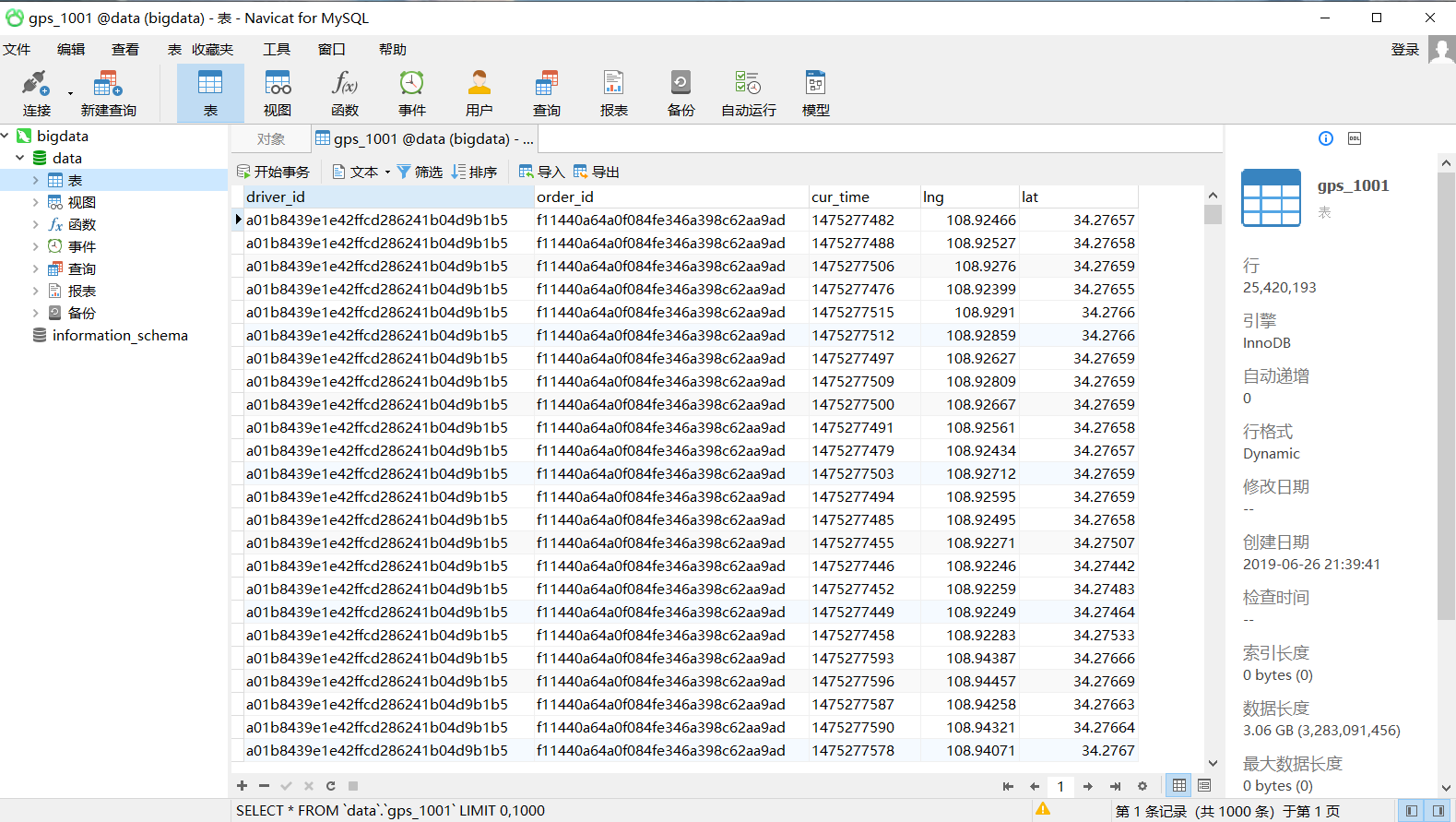
整体项目结构截图如下：



其中data目录下是生成的派生数据。Output中是最后的调度方案。Static中是用于展示热力图的静态网页。

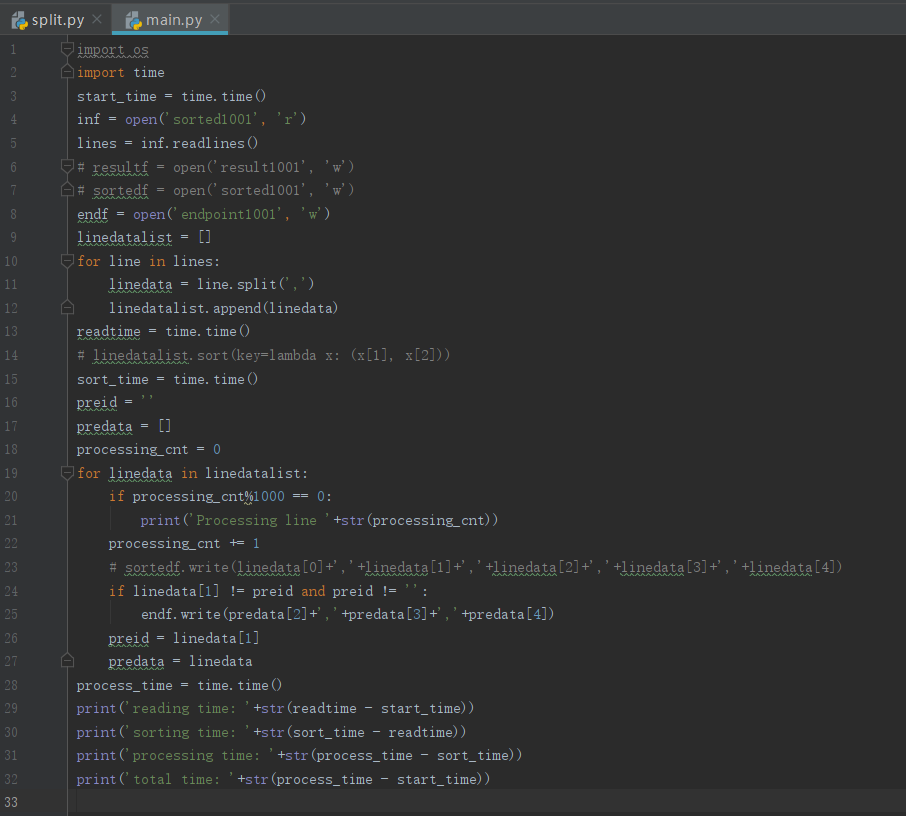
实验过程

1. 使用navicat访问服务器，简单观察数据包含信息，估计数据量。观察发现数据包含司机id，订单id，经纬度和时间戳。我们经过小组讨论决定专注于起止点数据完成本次课题研究。

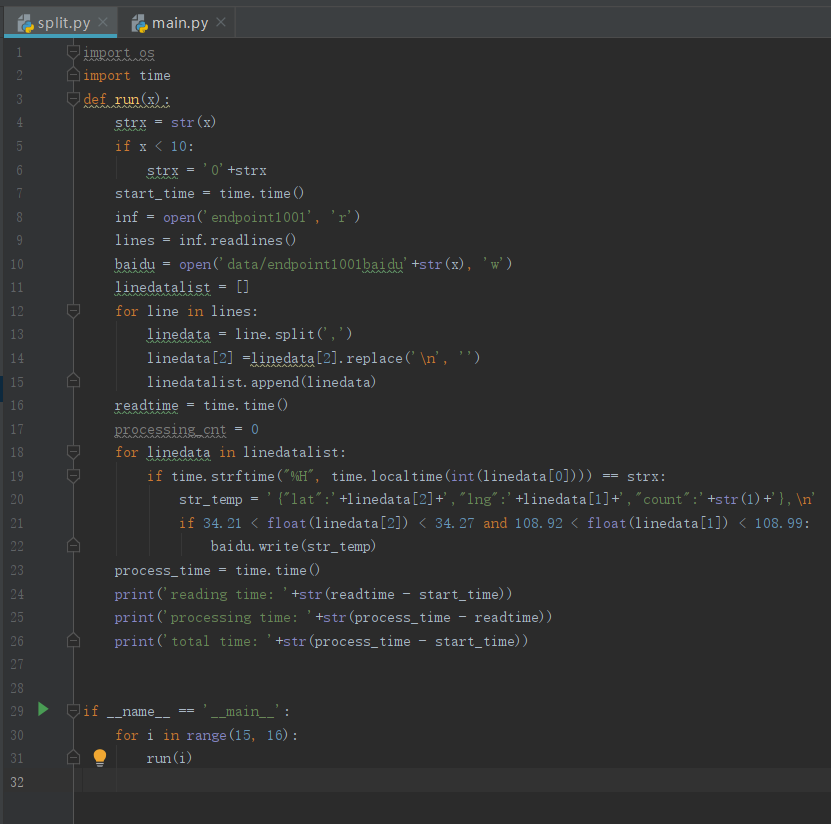


1. 搭建分布式集群用于数据处理。安装hadooop，hive，hbase，MySQL，spark等，详见部署文档。但最终因为云主机算力不足放弃了使用虚拟机的方案，转用个人电脑进行数据处理。主要使用python处理数据。
2. anaconda+python环境搭建。由于这两者较为常见，电脑本来就安装过，这里不再赘述。
3. 数据提取，下载ftp上的滴滴数据包，解压，使用python读取数据并存储到内存中，按照实验方案进行排序操作，排序之后的数据输出并存储为sorted文件以备之后使用。对排序之后的数据提取起点信息并输出为result，这一部分操作主要在main.py中完成。

进行数据“削边”，经过我们的多次试验，最终留下纬度34.21-34.27经度108.92-108.99范围内的数据作为有效的起点数据。后对数据进行按时段分类，将一天的数据分为24个小时数据。这部分主要在split.py中完成。

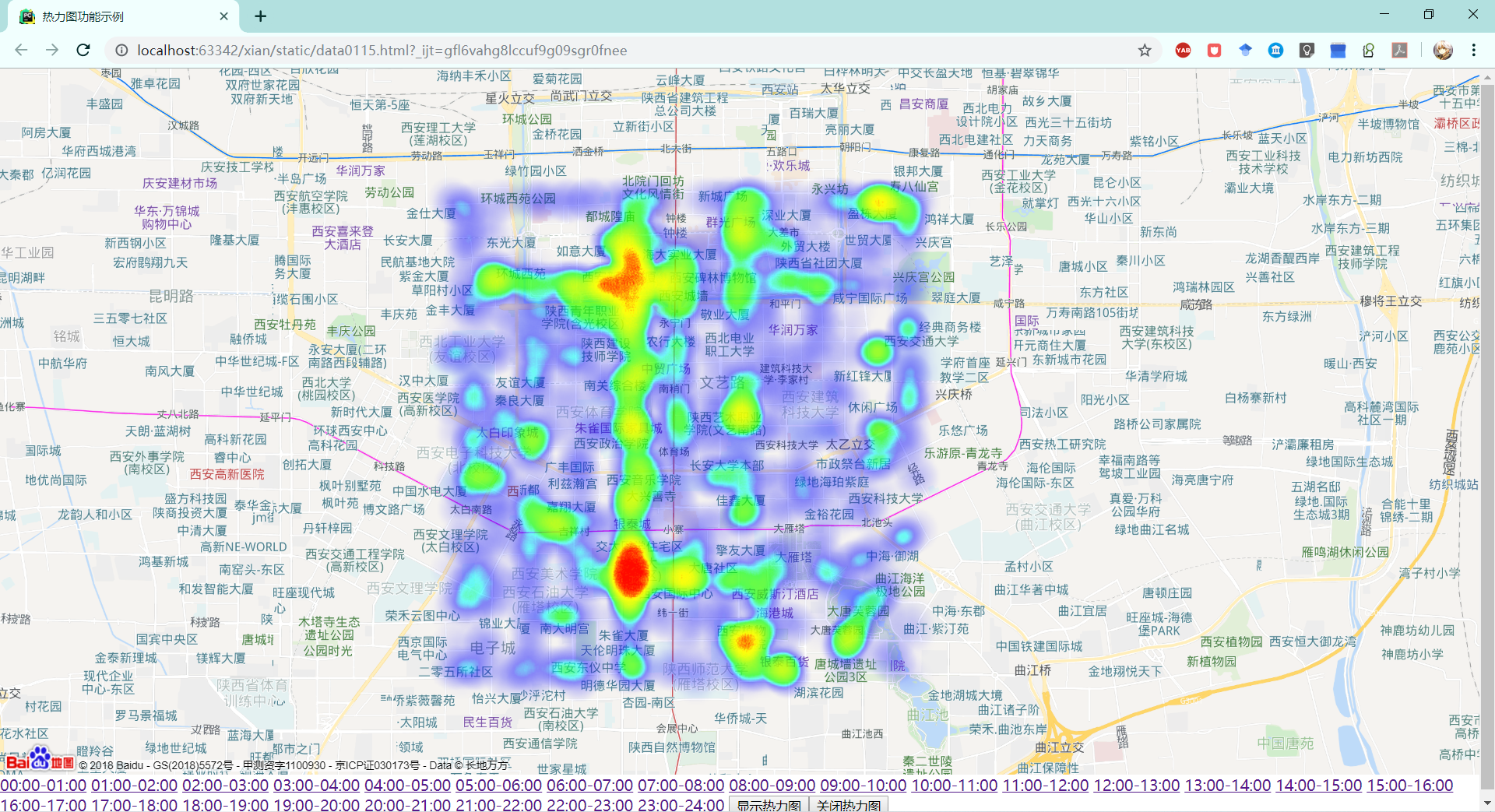


Main.py代码截图



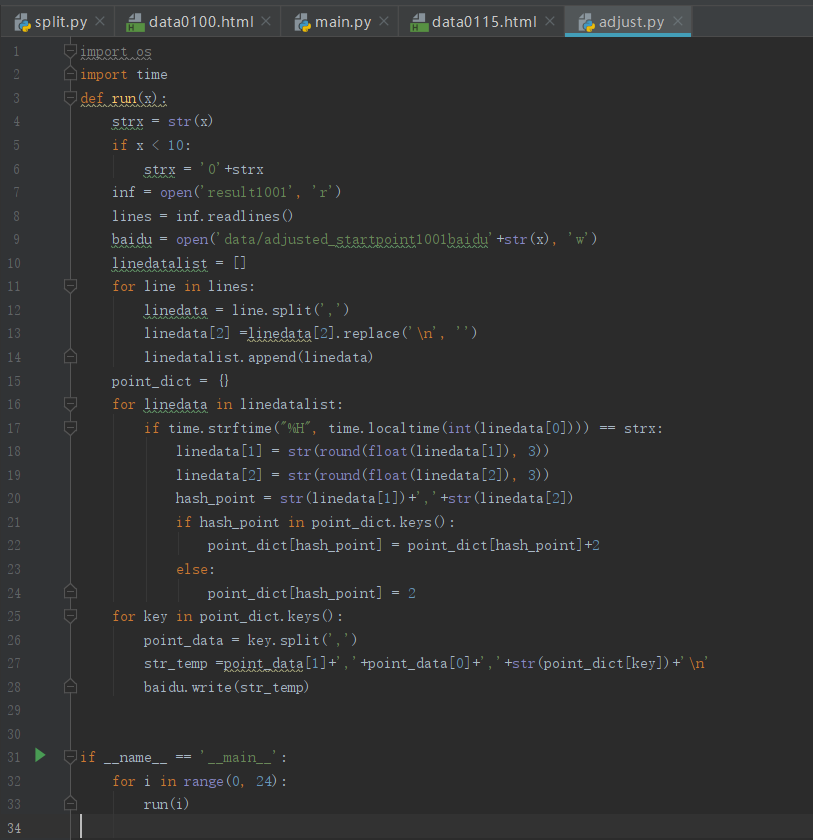
Split.py代码截图

1. 转换为百度热力图api接受的数据格式调用百度api绘制热力图。绘制出的热力图效果如下：



热力图效果图

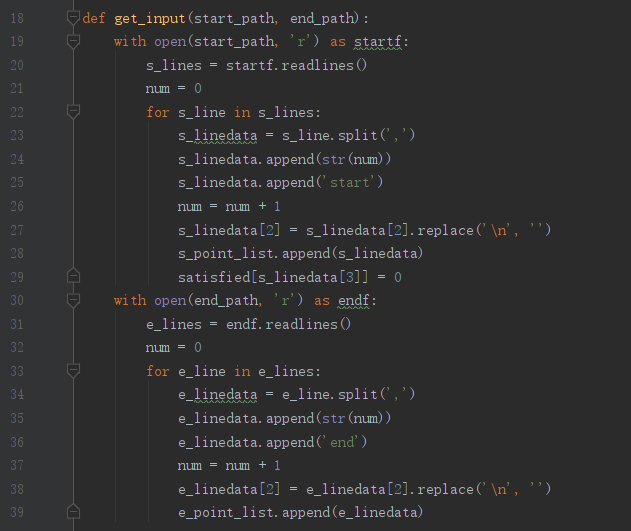
1. 使用sorted数据生成终点数据。终点数据保存在endpoint文件中
2. 将起点终点数据转化为网格点权值，这一部分主要由adjust.py完成



Adjust.py代码截图

1. 使用chain算法与r\*tree进行weighted-spatial-matching，得到合理的调度方案。这部分主要由SPM.py完成。由于代码过长我们将代码按功能函数截图：

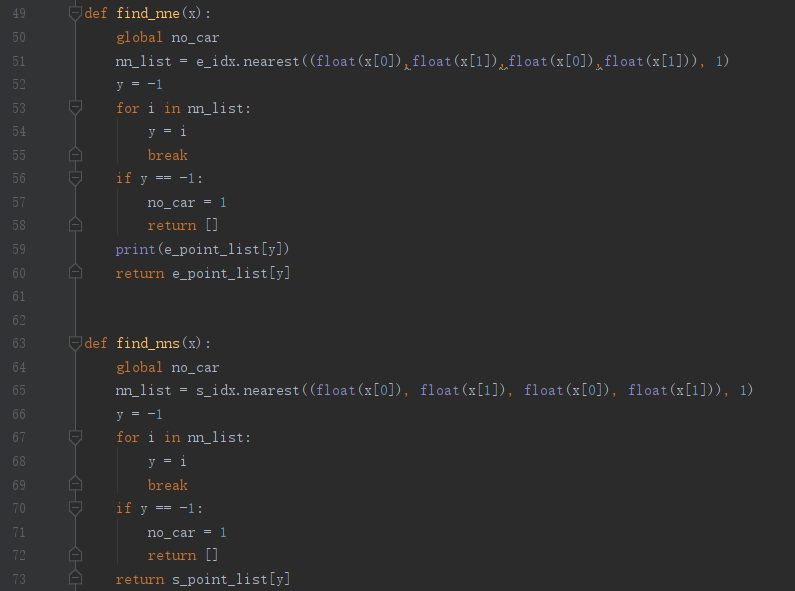
Get\_input()函数完成从文件中读取起点终点数据的操作



Idx\_build()函数完成了二维空间rtree结构的索引建立

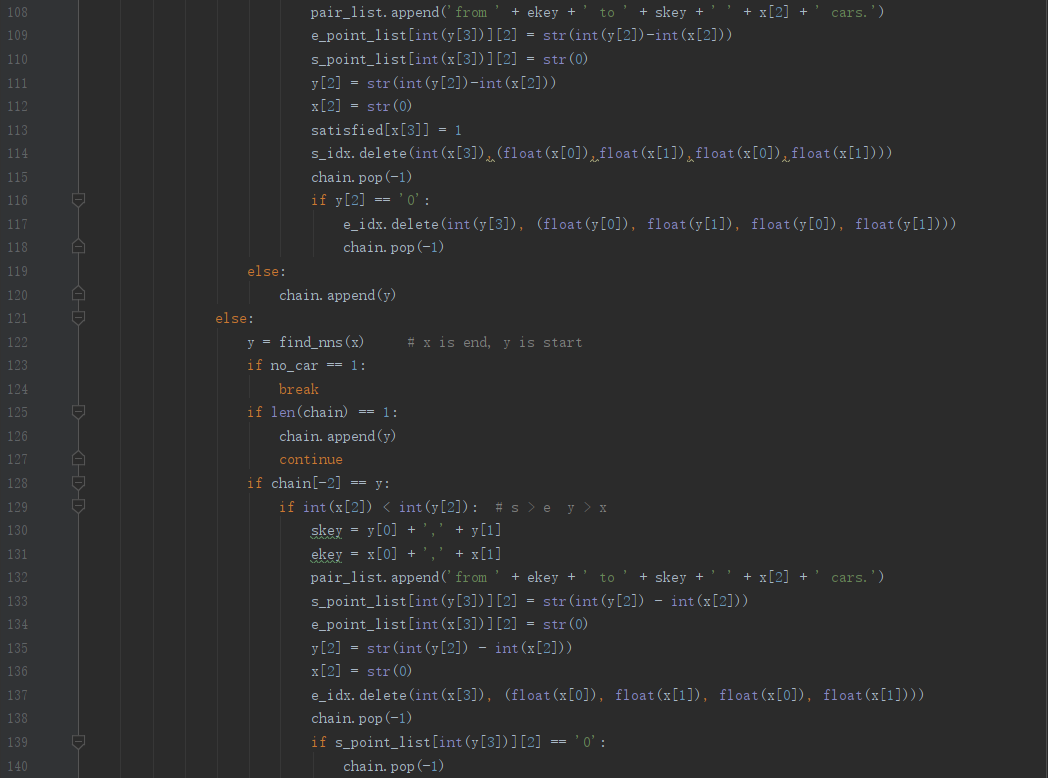


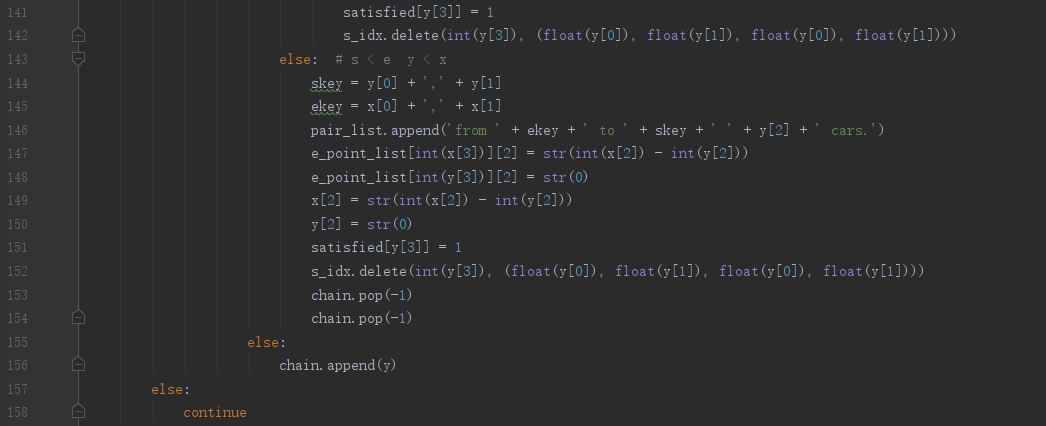
find\_nns()与find\_nne()分别是寻找最近的start point和寻找最近的end point



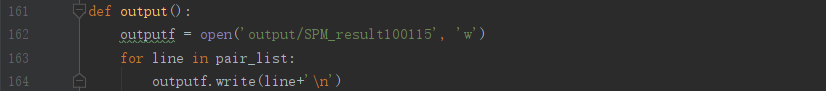
Solve()函数是chain算法的主要流程



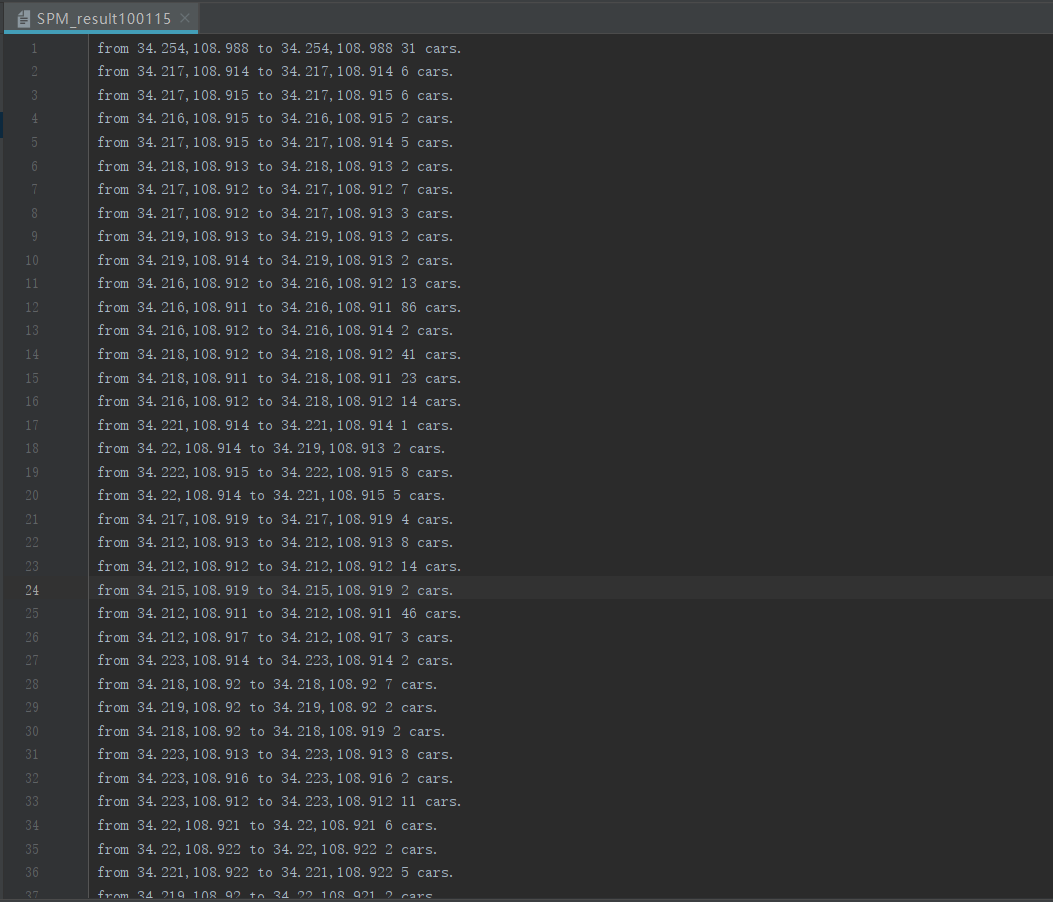




Output()函数是算法的结果输出



生成的调度方案如下图：



数据格式为：From A to B x cars意为需要从A区域调往B区域x车次。

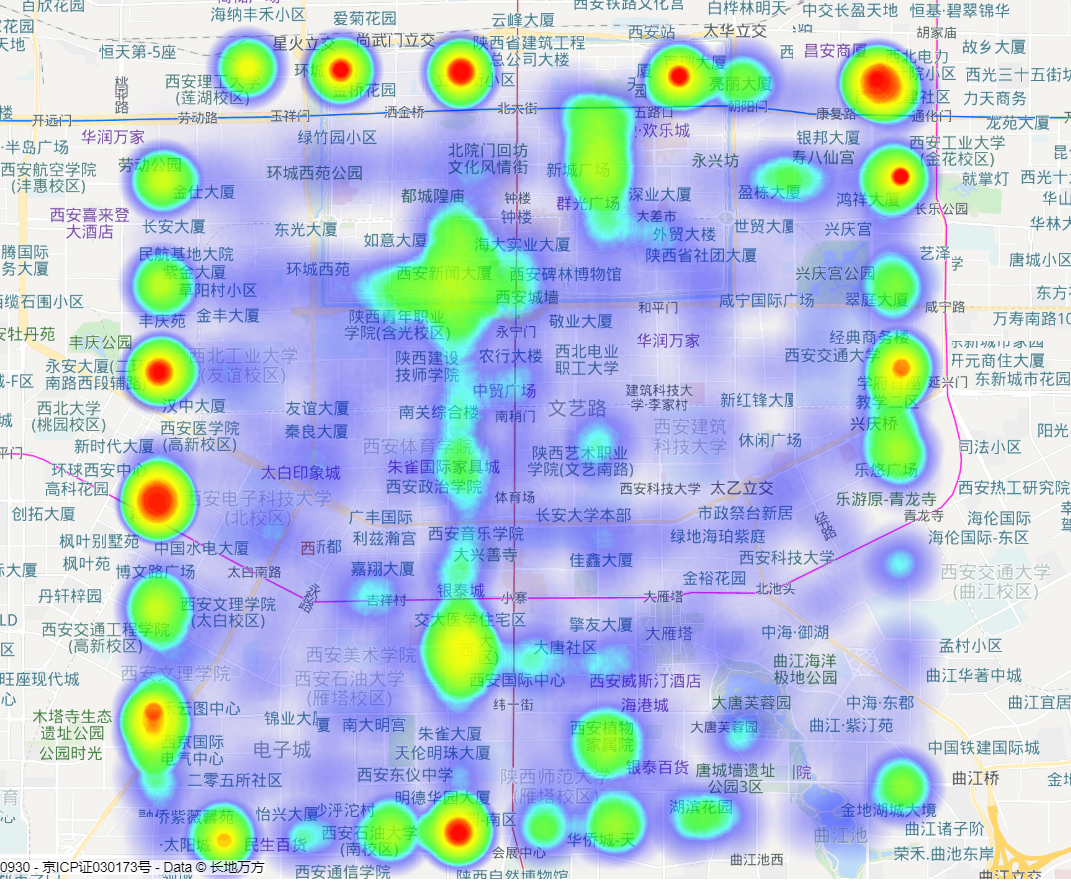
### 四、遇到的问题与解决方案

#### 4.1虚拟机性能不足问题

起初我们研究数据发现30天的数据显然虚拟机集群是不足以支撑其分析的。因此我们考虑减少数据量，只用一天的数据。但是使用一天的数据虚拟机还是要花费太长的时间去分析，因此我们最终改为了本地使用python进行数据分析运算的方案。本地机期配置：cpu i5-8300h，16GB内存，最终我们处理一天的数据排序得到起点信息花费时间约为5~10分钟。

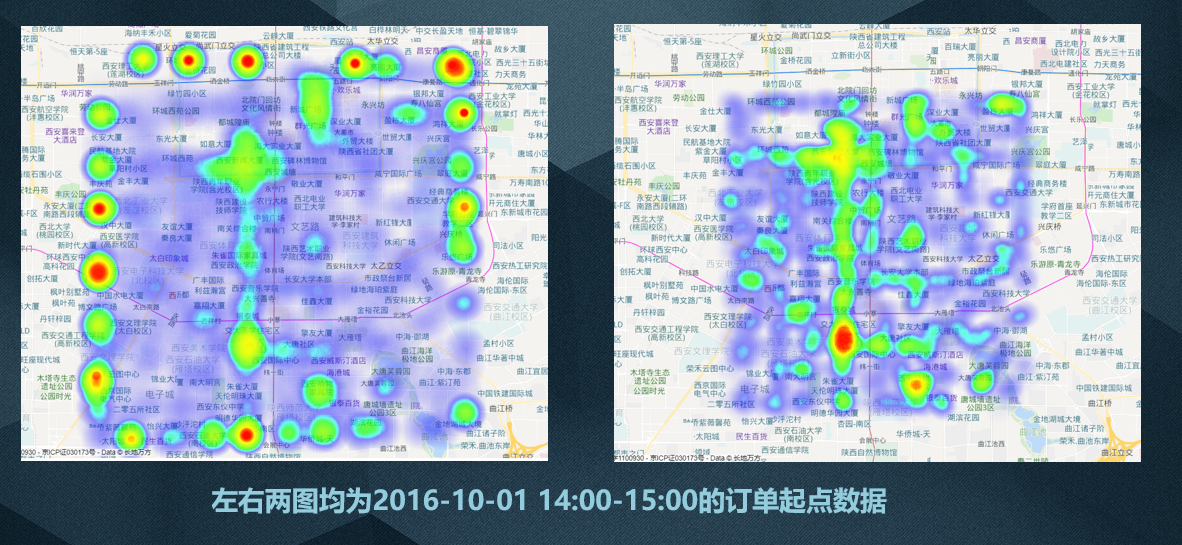
#### 4.2 伪起点问题

我们在实验初期得到起点分布热力图后发现取样地区周边地带起点数明显偏高，我们结合百度地图初步分析认为周边地区是居民区的缘故，但仔细分析发现周围地区数值偏高的量级实在太大不能简单地通过居民区解释。（效果如下图）



我们思考后发现造成问题的原因可能是订单路径点数据按空间区域截取，周边地区的“起点”并不一定是真正的起点，举例来说，当一个订单路径横穿我们取样区域边界时，进入边界的第一个数据点会被记录下来，而我们的时间戳排序寻找起点的算法会误认为这是一条订单的起点。实际上这只是一条路径上普通的点。因此我们决定消除这样的干扰。

但是想要解决这个问题并不容易，我们只通过区域内的数据无法准确的判断出这个点的确是起点还是一条跨界路径的第一个点。因此我们选择舍弃周边地带的数据点以保持分析结果的正确性。经过多次实验我们得到保留纬度34.21-34.27经度108.92-108.99范围内的数据既剔除了错误数据又最大程度的保留了正确数据。（处理前后对比如下）



#### 4.3调度算法的实现与现实意义

在调度算法的实现上我们基本上是按照我们参考的那篇论文的思路来走的。一个合理的调度方案应该满足乘客和司机的双向需求，即乘客想要最近的司机来接他，这样可以缩短等待时间，而司机希望接最近的单，这样可以减少无意义的行驶时间和路程从而减少成本开支。因此我们希望互为最近点对的司机与乘客优先配对。可选的算法很多，传统的找最近点对的方法和稳定婚姻匹配问题的解法都可以应用到本例中，但是chain算法是效率更高的解法。因为配对数据是二维空间内的点，因此可以使用R\*tree数据结构进行维护，它允许在log级别的时间复杂度内求出一个点的最近邻居。这也是论文中所使用的数据结构。最后整体的时间复杂度是O(n log n)级别，n为总点数。[1]

得到的配对方案可以作为调度计划指导平台给司机下达客户订单，通过这样的匹配计划可以最大程度减少用户的等待时间、司机的无意义行程，增加平台订单成交量。

**参考文献**

[1]Raymond Chi-Wing Wong, Yufei Tao, Ada Wai-Chee Fu, and Xiaokui Xiao. 2007. On efficient spatial matching. In Proceedings of the 33rd international conference on Very large data bases (VLDB '07). VLDB Endowment 579-590.