

电 子 科 技 大 学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

**矩阵理论小论文**

**Papers on Matrix Theory**

论文题目  **基于云平台的谱聚类算法研究**

学科专业 计算机技术

学 号 201922230108

作者姓名 **王 冠 桥**

**摘要**

如今，随着家用车辆的数量增加，人们对车辆的出行安全保障有了更多的需求，人们对于车辆出行的安全性的关注度越来越高，驾驶行为分析与研究成了一个交通领域以及车辆领域共同所关注的焦点课题。近年来，云计算的飞速发展使得各行各业的技术日新月异，这对于车辆方面的行为研究也有着极大的益处，在海量车辆驾驶行为数据问题下，大量的数据构成了车联网数据，因此对于车联网数据的分析与处理变得尤为重要。因此，本文基于车联网数据对驾驶行为模型进行研究对于智能识别系统的研究与整个车联网行业有着重要意义与价值。

本文内容主要包括以下几个方面：

（1）首先从OBD收集的车辆数据中提取与安全行为相关的特征参数，随后使用三种的不同的聚类算法对特征参数进行聚类，分别是K-均值聚类算法，二分K-均值聚类算法以及谱聚类算法，最后聚类得到激进型，一般型以及谨慎型这三种不同的驾驶行为

（2）使用Hadoop搭建云计算平台，并在此平台上实现谱聚类的并行运算，对海量的车辆数据进行更快速度的聚类

**关键词：**驾驶行为模式；云计算平台；谱聚类算法；

**Abstract**

Nowadays, with the increase of the number of household vehicles, people have more demand for the safety of vehicle travel. People pay more and more attention to the safety of vehicle travel. The analysis and research of driving behavior has become a traffic field and The focus of the vehicle field is the focus of attention. In recent years, the rapid development of cloud computing has made the technology of all walks of life change with each passing day. This has great benefits for vehicle behavior research. Under the problem of massive vehicle driving behavior data, a large amount of data constitutes the Internet of Vehicles data. The analysis and processing of car network data has become particularly important. Therefore, the research on driving behavior model based on vehicle network data has important significance and value for the research of intelligent identification system and the whole vehicle networking industry. The content of this paper mainly includes the following aspects:

(1) Firstly, the feature parameters related to safety behavior are extracted from the vehicle data collected by OBD, and then the feature parameters are clustered by using three different clustering algorithms, namely K-Means algorithm and bi-K-Means algorithm. And the spectral clustering algorithm, the final clustering gets three different driving behaviors: aggressive driving behavior, general driving behavior and cautious driving behavior.

(2) Use Hadoop to build a cloud computing platform, and implement parallel computing of spectral clustering on this platform to perform faster clustering of massive vehicle data.

**Keywords：**Driving behavior pattern; cloud computing platform; spectral clustering algorithm;

**目 录**

[**摘要** I](#_Toc28519160)

[**Abstract** II](#_Toc28519161)

[1 绪论 1](#_Toc28519162)

[1.1 课题研究目的及意义 1](#_Toc28519163)

[1.2 驾驶行为国内外研究现状 1](#_Toc28519164)

[1.2.1 国外研究现状 1](#_Toc28519165)

[1.2.2 国内研究现状 2](#_Toc28519166)

[1.3 本文研究内容以及章节安排 2](#_Toc28519167)

[1.3.1 本文的主要研究内容 2](#_Toc28519168)

[1.3.2 本文的章节安排 3](#_Toc28519169)

[2 驾驶行为聚类方法研究 4](#_Toc28519170)

[2.1 特征参数提取 4](#_Toc28519171)

[2.1.1 数据预处理 4](#_Toc28519172)

[2.1.2 Pearson相关系数 5](#_Toc28519173)

[2.1.3 依据相关系数提取参数 6](#_Toc28519174)

[2.2 聚类样本数据选取 7](#_Toc28519175)

[2.3 聚类方法研究 8](#_Toc28519176)

[2.3.1 K-Means算法 8](#_Toc28519177)

[2.3.2 BiK-Means算法 9](#_Toc28519178)

[2.3.3 谱聚类算法 10](#_Toc28519179)

[2.4 聚类算法效果 11](#_Toc28519180)

[2.4.1 聚类算法实现环境 11](#_Toc28519181)

[2.4.2 聚类效果 11](#_Toc28519182)

[2.5 本章小结 13](#_Toc28519183)

[3 基于云平台的并行谱聚类研究 14](#_Toc28519184)

[3.1 Hadoop平台研究 14](#_Toc28519185)

[3.1.1 分布式系统HDFS 14](#_Toc28519186)

[3.1.2 MapReduce框架 15](#_Toc28519187)

[3.1.3 分布式数据库HBase 16](#_Toc28519188)

[3.2 并行谱聚类算法研究 16](#_Toc28519189)

[3.2.1 谱聚类算法可并行部分研究 16](#_Toc28519190)

[3.2.2 并行计算相似矩阵 17](#_Toc28519191)

[3.2.3 并行计算特征向量 18](#_Toc28519192)

[3.2.4 并行实现K-Means聚类 20](#_Toc28519193)

[3.2.5 单机与并行下谱聚类算法对比 21](#_Toc28519194)

[3.3 实验仿真 22](#_Toc28519195)

[3.3.1 Hadoop平台搭建 22](#_Toc28519196)

[3.3.2 谱聚类算法实现 24](#_Toc28519197)

[3.4 本章小结 27](#_Toc28519198)

[4 总结与展望 28](#_Toc28519199)

[4.1 总结 28](#_Toc28519200)

[4.2 展望 28](#_Toc28519201)

[参考文献 30](#_Toc28519202)

# 绪论

## 课题研究目的及意义

近些年来，我国的车辆数量飞速上涨，到2018年底全国汽车的数量已经达到了2.4亿辆，比上一年的全国汽车数量同比增长了10.51%[1]，随着汽车数量的增长，人们在享受车辆带来的便捷的同时，一系列的车辆行驶安全问题也随之浮现，对2011到2017的数据统计中，约有86.4%的司机死于不良的驾驶习惯[2]，其中有94%的事故是由司机的个人因素所造成的，这其中有40%是驾驶员分神导致的,有33%是和驾驶员的判断失误有关,还有7%是非常严重的疲劳驾驶导致的，因此，驾驶行为的模型研究对于安全分析具有重大意义和深远影响。

针对实际驾驶环境，已有学者通过一系列的算法对其进行了研究，如K-均值聚类算法，层次分析法，核模糊聚类等算法，但是此类算法所针对的往往是极少量的数据，通常是一个或者几个司机的某几次驾驶行为数据分析所得到的结果，这样的分析无疑是片面且难以令人信服的，往往也是不准确的，因此，传统平台已经无法面对越来越多且杂乱无章的海量数据，随着近年来云计算以及神经网络的高速发展，基于海量车联网数据的驾驶行为分析逐渐成为主流。

## 驾驶行为国内外研究现状

### 国外研究现状

对于驾驶行为的研究从1938年就开始了，在那一年Gibson等人提出了一种叫做车辆行驶区域分析的理论，该理论表明了驾驶人应当将自己与车保持在一定的“安全驾驶范围”内，在一定程度内降低危害[3]。

2011年日本学者Masao Fukuishimat通过GPS技术获取数据以及问卷调查等方式题出了一些能够有效降低事故率的几种智能系统。[4]。

2012年，Eren等学者研究了基于贝叶斯神经网络的驾驶行为分析评估方法，他们对15名驾驶员在不同的天气状况下收集数据，通过收集速度，加速度，减速度等数据作为实验数据，通过贝叶斯神经网络模型和已有的不同安全系数的数据建立分类模型，最后使用该分类模型对新的驾驶员数据进行分析其危险性[5]。

2013年，Salameh等几位学者针对防撞预警系统开发了一种高精度的定位装置，通过收集GPS采集的速度，加速度等数据，提出了一种关于防撞提前预警的模型框架[6]。

2017年，Geqiqi等人通过采集车辆的速度及加速度等信息，利用核模糊聚类算法（KFCM）以及主题模型算法等两种方法实现聚类，一共聚类出三种不同的驾驶行为：激进型驾驶行为，谨慎型驾驶行为和温和型驾驶行为[7]。

### 国内研究现状

2008年，庄明科，王登峰等学者从问卷调查获取心理因素，结合道路状况，气象状况，驾驶行为等方面，对驾驶员的燃油消耗量与驾驶员的行为做出了分析，以一定程度的油耗反应了驾驶员的行为操作是否具有一定的风险性[8]。

2012年，高振海，段立飞等学者对人脑神经网络做了详细的研究，仿造此结构对驾驶员方向控制行为做出了分析，研究了三种不同的分类器，并进行了对比[9]。

2013年，来自于浙江大学的刘雁飞等人通过引入ACT-R模型，建立了驾驶员多任务操作和驾驶过程驾驶行为评估模型[10]。

2015年武汉理工的学者对车联网数据进行数据挖掘，实施了驾驶速度行为聚类研究，通过数据挖掘将速度，加速度等9个因子提取为特征参数，并聚类为四种不同的行为，并给出了一部分较危险驾驶员的驾驶行为参数参考[11]。

2017年朱波通过研究层次分析法和专家评分法对很大一部分的数据进行了处理，将司机的驾驶行为按照得分为三种类型：高风险行为，中等风险行为，低风险行为，并使用软件实现了安全平台的呈现[12]。

## 本文研究内容以及章节安排

### 本文的主要研究内容

在阅读了国内外大量驾驶行为相关的论文以及深入研究了部分云计算以及数据挖掘方面的论文后，基于传统的驾驶行为建模算法，本文研究了基于云计算平台的驾驶行为分析建模技术：云计算平台聚类-神经网络在线评估，首先在云平台上将采集到的参数按照特征参数进行聚类，参数拟定为速度与加速度绝对值两个，在云计算平台上按照谱聚类的聚类方法对OBD所采集到的数据进行聚类，依托于云计算平台的方式用并行的方式实现谱聚类算法，这样在处理大量数据时更加的快速，得到一定的效率，在云平台中将聚类后的数据打上标签，从而建立驾驶行为的标准样本库，此后再依托于此标准样本库，将其作为BP神经网络的训练样本集，从安全角度方面对新的驾驶数据进行建立评估模型，实现在线评估，最后将其应用于软件，实现评估。

### 本文的章节安排

第一章，绪论。主要介绍了驾驶行为-安全模型建立的目的及意义，针对目前国内外对驾驶行为安全性建模的方案与技术进行了研究，从而确定了本文对驾驶行为-安全模型建立的方案与技术手段，并确定了论文的大体框架。

第二章，驾驶行为聚类方法研究。在该章节中本文依据皮尔森（Pearson）相关系数方法提取相应参数，分析了K-Means，biK-Means以及谱聚类这三种聚类算法的可行性，并重点研究了谱聚类算法，比较其与另外两种算法的优势所在。

第三章，基于云平台的并行谱聚类研究。由于谱聚类的算法复杂度过高，因此在处理海量数据时单机速度过于缓慢，因此本章主要研究了云计算平台下并行谱聚类算法的实现。

第四章，总结与展望。总结论文中已完成的研究任务，并在此基础上指出未来需要研究的方向以及需要改进的部分。

# 驾驶行为聚类方法研究

传统的驾驶行为聚类算法，如K-Means，BiK-Means等虽然能在一定程度上处理数据，但是传统的聚类方法容易陷入局部最优值，并且对于海量数据的处理也不甚理想，因此本章主要选用谱聚类算法对OBD收集到的驾驶员行为数据进行分析，聚类驾驶行为。

## 特征参数提取

在聚类之前首要解决的问题便是特征参数的选取，因此需要对收集到的数据进行分析，通过一些数据处理的方法将初始数据处理为特定的数据，然后再对这些数据进行处理，提取出想要的特征参数。

### 数据预处理

由于OBD采集到的数据过多，并不能够全部作为聚类的参数，因此需要预处理大部分数据，并选择部分参数作为特征参数实现聚类工作，原始参数见表2.1。

表 .1 车辆行驶数据原始参数

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 单位 | 参数名称 | 单位 |
| 车辆ID | - | Gps编号 | - |
| 经度 | ° | 维度 | ° |
| 速度 | m/s | 加速度 | m/s2 |
| 当前挡位 | - | 时间 | s |

将车辆的数据处理为csv文件，然后用excel打开处理格式如下，从下图可以得到如表2.1所示的数据的具体展现：

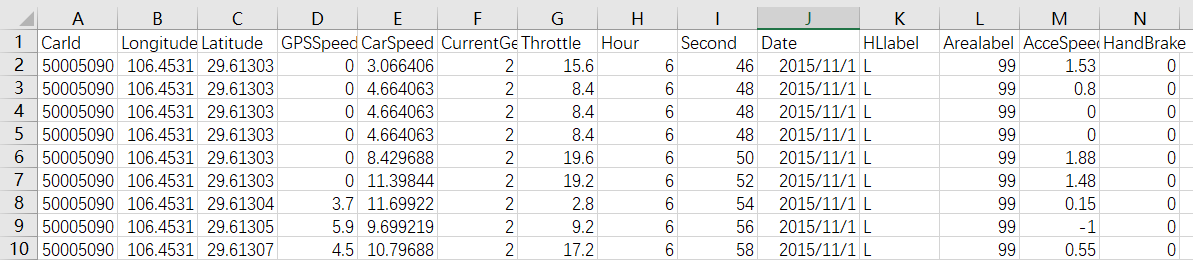


图 .1 车辆数据存储结果

可以从表中得到很重要的一点结论：当车辆的加速度持续一段时间为0时，车辆的速度也不会有所变化，因此在很长一段时间内，都会出现速度为某个具体值而加速度为0的情况。

针对这两点问题，有以下处理方法：对数据进行去重操作，在这里不需要对数据进行排序，因为数据本身就是按照时间进行收集的，而如果有重复的数据，也只是在最近的时间段内会出现，否则的话就不再是同一辆车的数据了，同样也不需要去重，因此只需要对相邻数据去重即可。具体伪代码如下表2.2

表 .2 预处理实现

加载数据：

For data in dataSet：

If 上一次的数据与当前数据相同：

去掉当前数据

### Pearson相关系数

皮尔森（Pearson）相关系数是衡量线性关联度的程度，简而言之就是衡量两个事物之间的相似程度。

如果有个变量：M、N，他们之间计算出的Pearson相关系数可以做如下理解：当相关系数为0时，M和N两变量无关系；当相关系数越接近1时，两者之间的关系越强。

在相关系数上，按照获得的Pearson值具体又可将相关程度划分为极强相关，强相关，中等相关，弱相关和极弱相关或无相关，具体见表2.3。

表 .3 Pearson相关系数程度表

|  |  |
| --- | --- |
| 相关系数绝对值 | 相关程度 |
| 0.8-1.0 | 极强相关 |
| 0.6-0.8 | 强相关 |
| 0.4-0.6 | 中等相关 |
| 0.2-0.4 | 弱相关 |
| 0.0-0.2 | 极弱相关或无相关 |

Pearson的计算公式如下图2.1，其中E（X）为X这一独立参数的均值,其中X可以为任意参数：

(2.1)

### 依据相关系数提取参数

在有了相关系数的计算方法后，对于特征参数的提取就较为方便了许多，首先确定好要比较的参数，这里分析后发现只有速度和加速度可以作为的提取参数，其余参数如车辆ID，GPS编号等数据均易知不可作为提取的参数，那么一个可以数字特征化的参数就做寻找到了，另一个相关系数的数字特征需要进行数字化，要对速度或者加速度与安全系数的关系做相关系数化，这里需要引入交通事故风险这一特征，这一特征在澳大利亚学者研究中可以得到，车辆速度越高，事故的风险比率越大[11]，大致如表2.4所示。

表 .4 车辆速度与事故风险比率关系

|  |  |
| --- | --- |
| 车辆速度（km/h） | 事故风险比率 |
| 60 | 1.3 |
| 65 | 2.7 |
| 70 | 14.3 |
| 75 | 25.6 |
| 80 | 41.3 |

由此可见，利用事故风险比率这一数据可以很好的量化速度（加速度）与安全指数的评估。

因此可以使用matlab来计算Pearson系数，因为matlab自带Pearson相关系数计算公式，因此只需要对数据进行输入就好了，计算速度的速度系数详细过程见图2.2，加速度与事故风险比率的计算方法类似：

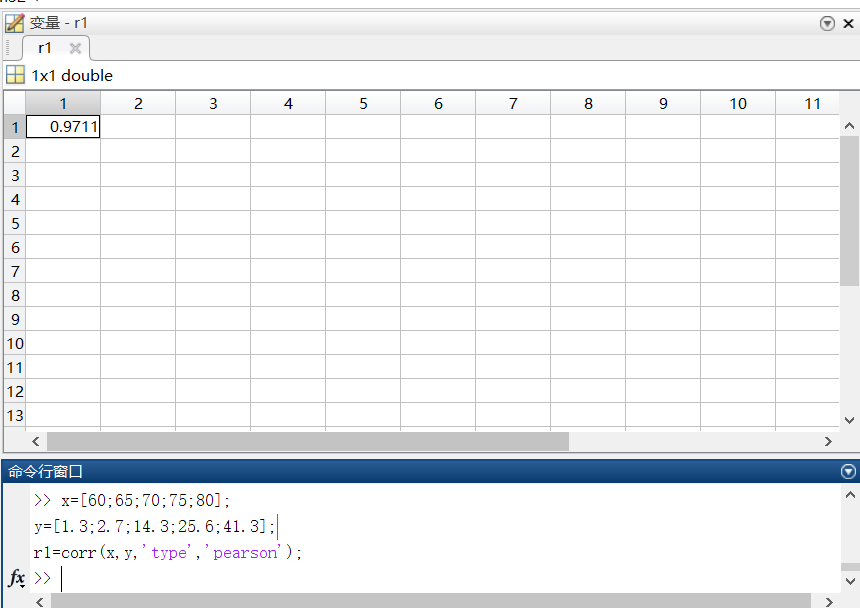


图 .2 matlab计算相关系数

利用2.1所示的相关系数计算公式和matlab自带公式即可得到加速度与速度对于安全指数的相关性，如表2.5所示：

表 .5 车辆速度与事故风险比率关系

|  |  |  |
| --- | --- | --- |
| 参数名称 | Pearson相关系数值 | 相关程度 |
| 速度 | 0.9711 | 极强相关 |
| 加速度 | 0.7846 | 强相关 |

根据表2.5可知，速度与加速度具有强相关与极强相关的特性，因此应该将速度与加速度两个参数作为样本聚类的两个特征参数。

## 聚类样本数据选取

本文所使用的样本数据均为OBD所采取的数据，由于需要部署到云计算服务器进行研究，因此采集到的数据量较多，这样才能保证数据的准确性和算法的精确度。

但是由于初始阶段聚类算法的探讨并不在云计算平台上进行，因而初始聚类样本的选取不宜过多，因此只得采用一部分数据进行初步聚类算法的探讨，由于聚类算法对数据的要求并不严格，因此采用二八方法随机选取一小部分数据作为聚类算法探讨的样本。

此样本对收集的样本进行预处理，只得到速度及加速度两个参数，并对重复的数据进行去重，最后得到的数据大约在1000条左右，此时的数据量用于谱聚类算法约为1-10s不定，为单机用户可接受的数据程度。

## 聚类方法研究

### K-Means算法

K-Means聚类算法的思想为：将N个样本数据按照一定的算法随机聚类到K个簇中，该算法将簇与簇之间的相似度尽可能的降到最低。

K-Means算法的流程如下：

1. 随机选取K个点作为质心
2. 将数据集中的每一个点分配到一个簇中，具体来讲，为每个点找距其最近的质心，并将其分配给质心所对应的簇
3. 在上一步完成后，更新质心坐标，也即重新计算簇内所有点坐标平均值，重复第二步，直到所有点的簇心不再更新或误差小于一定值

伪代码实现如下：

**表2.6 K-Means伪代码实现**

随机选取k个点作为初始质心：

当任意一个点的簇分配结果发生改变的时候:

对数据集中的每个数据点:

对每个质心:

计算质心与数据点之间的距离

将数据点分配到距其最近的簇

对每一个簇:

求出均值并将其更新为质心

K-Means算法的优点：

（1）处理数据快速，对于轻量级数据具有极佳的处理效果

（2）算法简单，易实施

（3）需要自己确定K的值

K-Means算法的缺点：

1. 无法处理大量数据

（2）容易陷入局部最优值

### BiK-Means算法

二分K均值（BiK-Means）聚类的思想与K-Means类似，不过BiK-Means克服了K-Means算法容易陷入局部最优值的缺点。

为了度量划分效果，这里引入误差平方和（SSE）这一参数，计算方法为每个点到质心的距离的平方和，SSE值越小表示聚类效果越好，一般通过合并两个簇以实现SSE值的减小，二分K均值聚类算法即是使用的上述技术以实现最优化

BiK-Means算法流程：

1. 将所有点作为一个簇，然后将该簇一分为二
2. 之后选择一个簇作继续划分，选择哪一个簇来划分需要进行决策，一般对所有簇都做一次划分，选择SSE增幅最小的那一个簇作为最终需要划分的簇
3. 上述过程不断的迭代，直到最后划分的簇的个数达到要求的个数

伪代码如下：

表 .7 BiK-Means伪代码实现

将所有数据点看成一个簇

当簇数目小于k时

对每一个簇

     计算总误差

     在给定的簇上面进行k-均值聚类（k=2）

      计算将该簇一分为二后的总误差

选择使得误差最小的那个簇进行划分操作

BiK-Means算法的优点：

（1）对于K-Means聚类算法做出了改进，使得在簇类划分的时候不会陷入局部最小值，保证聚类的成功性

（2）算法效率较快

BiK-Means算法的缺点：

1. 同K-Means算法一样，需要自己确定K的值

（2）对于非凸型数据处理起来较为困难

### 谱聚类算法

谱聚类算法是一种基于图论思想的聚类算法，该算法通过对样本数据处理，得到拉普拉斯矩阵，再提取拉普拉斯矩阵的特征向量，最后对特征向量进行聚类。

谱聚类算法简单描述：

输入：n个样本数据X={x1,x2,x3,…,xn}以及需要聚类的簇的数目k；

输出：聚类簇：A1,A2,..,AK

1. 使用2.2的公式计算N\*N矩阵的相似度Sij，相似度矩阵W由Sij组成

(2.2)

1. 使用2.3的公式计算度矩阵D，D由di组成

(2.3)

1. 计算拉普拉斯矩阵L = D-W
2. 计算拉普拉斯矩阵L的特征值，取前k个最小的特征值的特征向量u1,u2…uk
3. 将第四步产生的k个特征向量组合成特征矩阵U = {u1,u2…uk}，U∈Rn\*k
4. 令yi为U的第i行向量，其中i = 1,2…n
5. 使用K-Means算法将新样本点Y = {y1,y2,…,yn}聚类成簇C1,C2,…,CK
6. 输出簇A1,A2,…,AK，其中Ai = {j | yj∈Ci}

简言之谱聚类算法也就是利用图论的知识理论，将样本数据从离散的点处理为矩阵，将样本点从数据处理到相似矩阵，再从相似矩阵处理到拉普拉斯矩阵，随后在拉普拉斯矩阵上进行处理，利用某些固定的算法将拉普拉斯矩阵的特征向量提取出来，再对这些特征向量进行处理，利用一些传统的聚类方法（如K-Means等）将

这些向量作为新的样本点进行聚类，随后得到新的类簇，这就是该算法的思想。

谱聚类算法的优点：

（1）当需要聚类的类别个数（也即K的值）较小的时候，谱聚类的聚类效果会比较好，但是当聚类的类别个数较多的时候，则谱聚类算法的实施更为困难

（2）谱聚类算法对于高维数据的聚类更加的方便

（3）谱聚类算法在处理系数矩阵时更有优势，因为该算法只需要计算数据间的相似度数矩阵

谱聚类算法的缺点：

（1）谱聚类对于数据所构成的图的要求较高，而且对聚类的个数也有着苛刻的要求，小细节的变化都会造成该算法的失灵

（2）谱聚类算法适用于各个簇之间的数量相差不断的情况，如果一些簇的个数较少，谱聚类算法则会出现失效

## 聚类算法效果

### 聚类算法实现环境

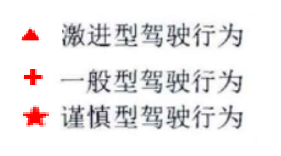
由于聚类算法的实现采用的数据量是可以直接实现的，所以选择在本机上进行实验，本次实验的实验环境部署如下表：

表 .8 实验环境资源配置

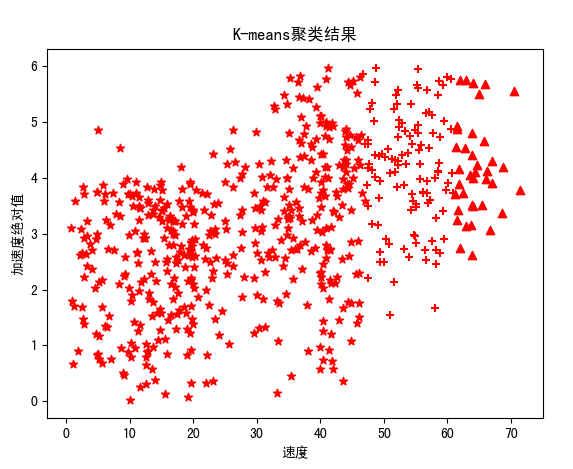
|  |  |
| --- | --- |
| 名称 | 详细配置 |
| 计算机 | Inter Core i5,8G RAM,NVIDIA 960M |
| Python | Python 3.5 |
| Matlab | Matlab 2016a |
| Ubuntu | Ubuntu 18.04LTS |
| JDK | JDK\_1.8.0 |

### 聚类效果

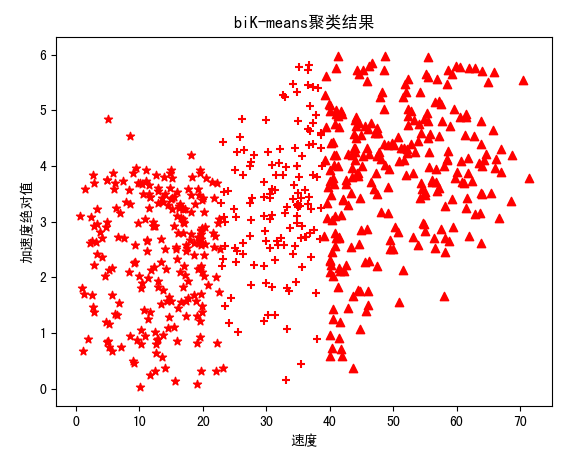
基于2.1节所提取的特征参数拟定了两个参数：加速度与速度，2.3节讨论的算法，基于提取的特征参数使用K-Means，BiK-Means以及谱聚类算法对一部分车辆数据进行实验，效果图如下：



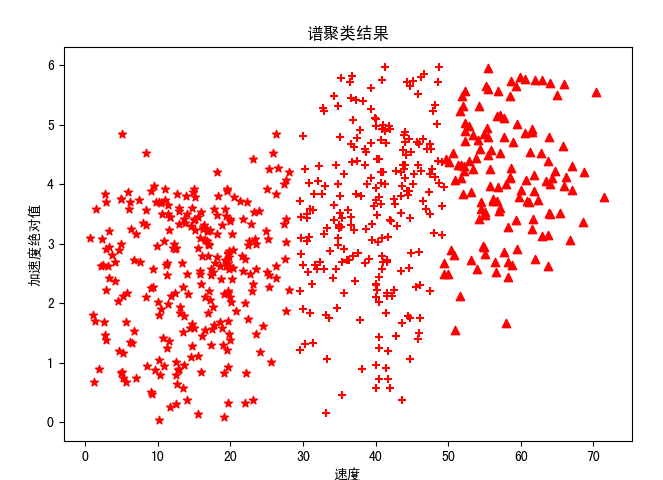
**图2.3 聚类颜色分类情况**



**图2.4 K-Means聚类结果**



**图2.5 BiK-Means聚类结果**



**图2.6 谱聚类结果**

图2.4，图2.5，图2.6分别显示了K-Means，BiK-Means以及谱聚类三种聚类算法对实际数据的处理，简单分析可知，由于K-Means算法的局限性，容易陷入如图2.4的情况，陷入一些局部最小值，而BiK-Means算法虽然解决了局部最小值的问题，但是从图2.5可以很明显的看出BiK-Means算法在聚类是虽然做出了聚类，但是它将大量的一般型行为（黄色簇）划分到了谨慎型行为中（绿色簇），而且对于一些数据的处理，BiK-Means混淆了其簇类，划分效果不明显，由此从总体上来看，谱聚类算法的聚类效果最好，BiK-Means聚类算法其次，K-Means聚类算法效果最差。

## 本章小结

本章主要研究了聚类算法的实施，先依据Pearson相关系数提取了特征参数，然后依据一定的方法选取了合适的测试数据集，最后研究了三种不同的聚类算法：分别是传统的K-Means聚类算法与BiK-Means聚类算法以及当前对于处理球形以及小部分数据极为有效的谱聚类算法，并通过可视化聚类对比三种算法的聚类效果，结果发现谱聚类算法具有更好的处理效果以及更高的精确度。

# 基于云平台的并行谱聚类研究

通过上一章的探索，认为谱聚类（Spectral Clustering）算法是一种相较于传统算法更为有效的一种聚类算法，在经过上一章测试数据的验证后，可以得出结论：谱聚类相较于传统的聚类算法是能够得到更好的聚类效果的。

虽然谱聚类算法具有非常好的聚类效果，但是有一个问题一直困扰着谱聚类的实际应用，那就是它的时间复杂度，在一般的情况下，谱聚类算法的时间复杂度达到了O（n3），其中n为收集的样本数据的个数，这样的算法复杂度导致谱聚类在处理大量数据时速度过于缓慢，相较于传统的聚类算法而言，谱聚类算法更难去处理大量的数据，因此对于此类算法而言，所要面对的问题就是如何解决大数据问题下的谱聚类算法实现。

针对目前所遇到的问题，科研人员们大部分围绕着两类方式在进行研究，一类是基于采样的谱聚类，另一种方法是基于并行的谱聚类算法，由于云计算平台的飞速发展，因此在实现基于并行（Parallel-based）的谱聚类算法上有更加简单易行的方法去实施，因此本章主要依托于，借助于分布式的计算环境，能够简单且快速的进行并行计算。

虽然基于并行的研究方法有很多，如PVM，MPI等方式，但是这些方式要么对于计算的算法编写要求过高，要么对计算的性能要求过高，因此这两种并行方法都不适宜于推广进行研究，而唯有MapReduce框架可以简单且方便的实现吗，而基于MapReduce框架的Hadoop平台就成了首要的研究对象。

## Hadoop平台研究

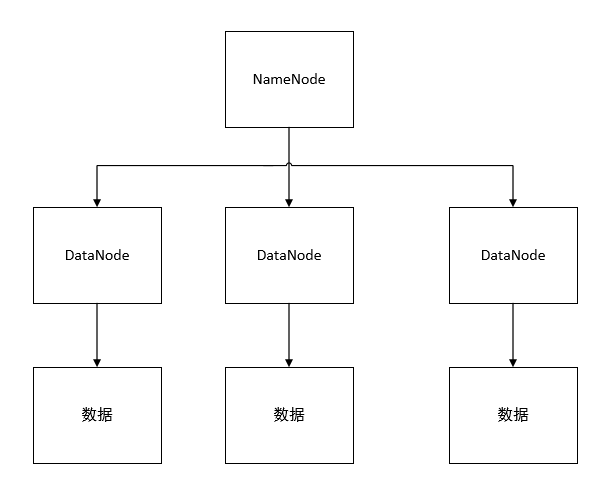
Hadoop是一个成熟的开源云计算平台，它使用的框架是基于微软研究的MapReduce框架，同时Hadoop也实现了一个分布式文件系统（Hadoop Distributed File System），简称为HDFS，在此分布式文件系统上，Hadoop也实现了一个分布式数据库HBase。

### 分布式系统HDFS

HDFS是一个分布式文件存储系统，它对于硬件的要求并不高，因此实现成本低廉，使得分布式的部署更为节约成本，而且HDFS还实现了流式访问数据的功能。

对于超大的文件来说，它们的存储和访问是一个难题，但是HDFS系统很好的解决了这个问题，HDFS将文件分割成一个个的小块，将这些小块随机的分配到集群中，运行在HDFS上的文件多是以流式的方式读取，这样的方式使得访问模型变得更加的高效。

HDFS采用了master/slave结构，集群通常由一个NameNode和多个DataNode组成，客户端通常用NameNode和DataNode交互来访问系统，上一段中提到的数据分块就是放在DataNode中的，而NameNode则是负责统筹整个HDFS系统的运行管理他们之间的关系见图3.1：



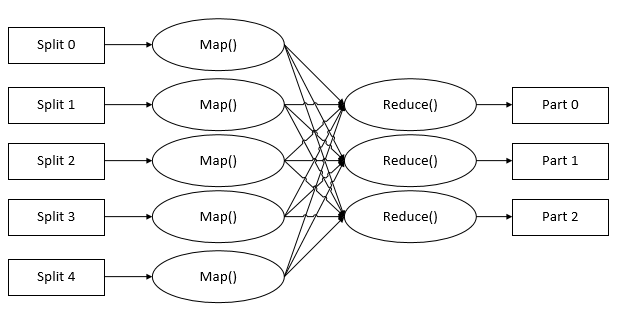
**图3.1 HDFS集群节点结构**

### MapReduce框架

MapReduce是一种处理海量数据的并行框架，MapReduce框架是运行在Google的分布式文件系统GFS（Google File System）的，Hadoop基于Google公司提出的几篇关于MapReduce的论文复现了MapReduce框架并实现了开源。

MapReduce采用的分治的办法对数据进行处理，简单来说MapReduce就是将任务先分解处理，再将处理后的小任务再聚合到一起，MapReduce简单来说其实就是Map和Reduce两个函数，map函数负责分解任务，reduce函数负责合并任务。

对于杂乱无序的数据，Map将这些数据解析出来，然后从中提取出Key和Value，也即是数据特征，经过此阶段后，Reduce所接收到的数据全部都是已处理好的Key和Value这一对键值，因此只需要对这一对键值所进行处理就好了。



**图3.2 MapReduce处理过程**

### 分布式数据库HBase

HBase的来源其实就是作为Hadoop Database的缩写而已，因此HBase实际上就是服务于Hadoop的一个数据库，HBase作为以列存储的数据库，它适用于联机事务处理，并且数据库的行和列都被视为小数目。

HBase建立在HDFS上，是一种数据库系统，而且HBase具有实时读写的功能，这样使得在处理较大规模的数据集时更加的灵活方便，HBase不同于一般的关系数据库，HBase对于非结构式的数据有着非常好的处理能力。

HBase仍然使用HDFS作为它的数据库文件存储系统，并且同样使用MapReduce框架来处理该数据库中的海量数据，HBase和HDFS是互通的，两者可以互相之间进行数据访问，并且实现实时读写。

因此，HBase更适宜用于大规模数据的存储和更改，因此，对于Hadopp这样的云计算平台而言，HBase是很有必要的。

## 并行谱聚类算法研究

在算法2.3.3中简要的提到了谱聚类算法的实现，但是由于谱聚类算法的复杂度过高，因此需要使用Hadoop这一云计算平台对谱聚类算法做出并行处理，以求加快数据的处理速度，在本章最开始部分提到了需要采用并行的方式来优化谱聚类算法，因此需要研究谱聚类算法可以加快速度的部分，然后对其进行并行操作以求加快速度。

### 谱聚类算法可并行部分研究

基于2.3.3节的算法，再次给出标准化后的谱聚类算法（简单描述）：

**表3.1 谱聚类算法**

输入：n个样本点X={x1,x2,x3,…,xn}和聚类簇的数目k；

输出：聚类簇：A1,A2,..,AK

（1）计算得到相似矩阵W

（2）构造度数矩阵D

（3）计算拉普拉斯矩阵L = D-W

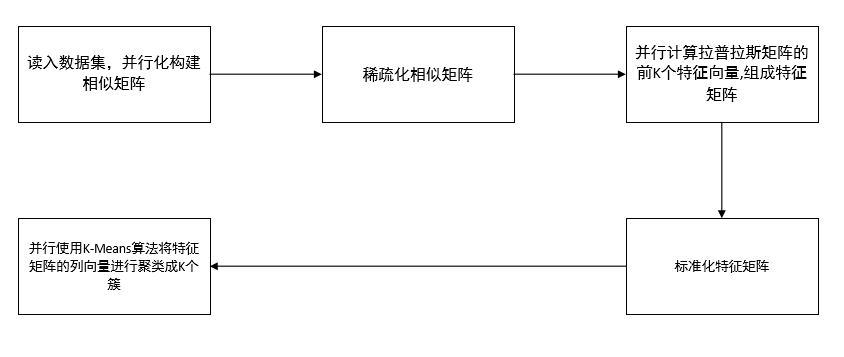
（4）计算出拉普拉斯矩阵的K个特征向量，并将这K个特征向量作为特征矩阵U的列向量

（5）标准化矩阵U为Y

（6）用K-Means算法将y­i聚类成K个簇A1,A2,..,AK

从上面的算法可以得知，在第二步的度数矩阵计算上，实际上是一个O（n）的做法，而拉普拉斯矩阵L的计算也是一个O（1）的做法，因此所需要进行优化的步骤也就三个地方：（1）相似矩阵的计算过程（2）拉普拉斯矩阵的K个特征向量的计算过程（3）谱聚类算法中K-Means算法最后对特征矩阵聚类的过程。因此在此过程中，可以将上述三个步骤中可以并行的部分提取出来放入Hadoop平台的计算过程中以实现加速。

基于朴素的谱聚类算法，从上面的分析可以提取出具体的并行谱聚类算法流程图：



**图3.3 并行谱聚类算法实现流程图**

### 并行计算相似矩阵

首先需要知道朴素的相似矩阵计算办法，相似矩阵的朴素做法便是对每一对不同的<i，j>都需要计算相似性Wij，然后才能够获得一个相似矩阵W，此做法的算法复杂度为O（n2），伪代码如表3.2：

**表3.2 朴素相似矩阵算法**

for i to n:

for j = i+1 to n:

compute Wij（计算i与j的相似性）

但是N方的复杂度显然不是所需要的，因此需要对此过程进行并行计算，通过表3.2的朴素算法可以很轻松的看出来相似矩阵的计算分为两个过程，对于每个i，计算每个j的相似度，这里只需要从j = i+1开始计算，否则会形成重复计算，因此计算次数实际为N\*(N+1)/2次，所以可以让每个Map去实现某个i为基础，j从i+1到n的计算过程，这样的话可以形成n个map过程，这样的话每个过程只需要n-i+1次计算就可以了，又因为所有的Map过程是并行计算的，所以实际上再并行计算后，这个算法的复杂度变成了O（n），所以基于HDFS的相似矩阵并行计算过程大致可以用如下的伪代码来表示：

**表3.3 并行计算相似矩阵伪代码**

Input: <Key, Value>

Output:<Key’,Value’>

index1 = key, index2 = n-key+1

for i in {index1, index2}:

i\_Data = getDataFromHDFS(i)

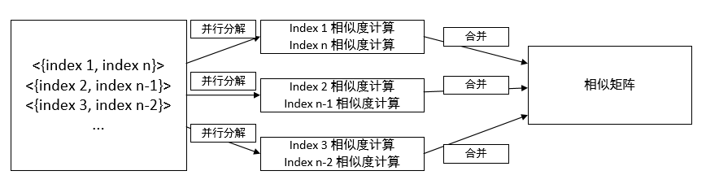
for j = i+1 to n:

j\_Data = getDataFromHDFS(j)

similarity = computeSimilarity(i,j)

storeSimilarity(i, j ,Similarity) into HBase

依据表3.3以及表3.2所示的伪代码对比可以很明显的看出比起传统的做法，基于HDFS的并行相似矩阵计算需要使用MapReduce框架来实现并行，注意到的是如果按照上一段讨论的做法来做，那么最后一个Map所计算只有第n-1个值和第n个值，而第一个Map则需要从1到n计算完，需要计算n-1次，所以为了均衡次数，将第一个Map和最后一个Map合并，第二个Map与倒数第二个Map合并，这样减少了Map的数量，并且均衡了并行负载，过程如下图3.4。



**图3.4 并行计算相似矩阵**

在计算完了相似矩阵后，将其存入HBase中，以供之后的运算使用。

### 并行计算特征向量

Lanczos算法是一种将对称矩阵转换成对称三对角矩阵的算法，Lanczos算法可以产生一组特征向量，该组特征向量是正交的，并且可以将初始矩阵转化为三对角矩阵，这样原问题就转化为了求三对角矩阵的特征向量，尤其在于处理巨大大且稀疏的矩阵时，Lanczos算法具有非常好的效果，在计算k个特征矩阵时，Lanczos算法更加的有效，它只需要迭代k次就可以得到前k小或者前k大的特征向量，因此这里使用Lanczos算法计算拉普拉斯矩阵的特征值以及特征向量。算法Lanczos实施步骤见表3.4。

**表3.4 Lanczos算法伪代码**

输入：拉普拉斯矩阵L，大小为n\*n，迭代次数一般也为n

输出：一个n\*n的矩阵V以及具有正交性的三对角实矩阵T = V\*LV

(1)首先令V1为单位向量

(2)初始迭代过程：

1、令W1’ = L\*V1

2、令α1 = W1’\*V1

3、令W1 = W1’ – α1\*V1

(3)对于j = 2,…,n

1、让βj = ||Wj-1||（计算Wj-1的模长）

2、如果βj不为0，则让Vj = Wsj-1/βj，否则让vj作为一个正交与v1到vj-1的一个向量

3、令Wj’ = A\*Vj

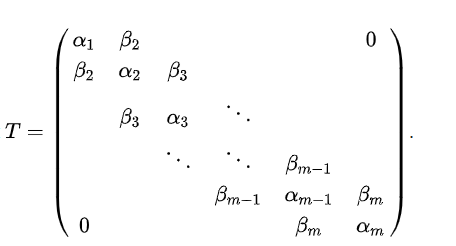
4、令αj = Wj’\*Vj

5、令Wj = Wj’ – αj\*Vj – βjVj-1

(4)将V1,…,Vm作为V矩阵的列向量拼起来，得到V矩阵

(5)得到三角实对称矩阵T = V\*LV

经过表3.4的公式计算之后可以得到如图3.5所示的T矩阵，也就是通过Lanczos算法得到了想要的三对角矩阵。

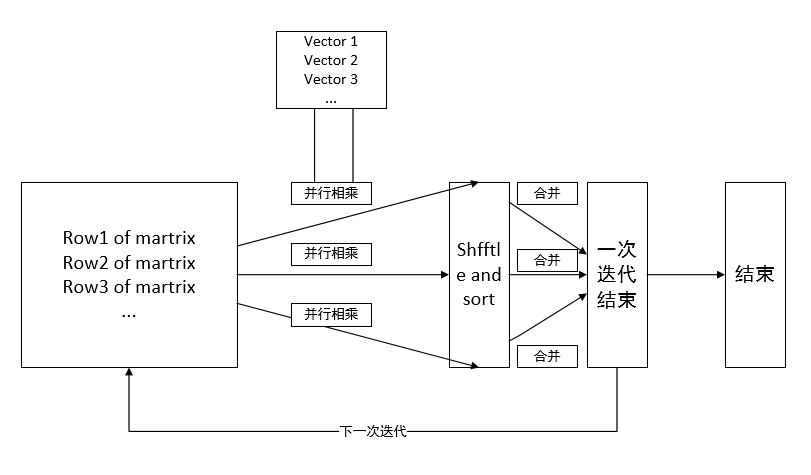


**图3.5 三对角矩阵T**

在得到了三对角矩阵T之后，需要对此三对角矩阵去求计算特征向量，朴素的特征向量做法不足以处理n过于巨大的矩阵，因为将过大的矩阵转化为线性方程的阶数过高，因此需要使用三对角矩阵来做优化，之后再使用QR算法一次次迭代，当迭代次数够多时，可以得到特征值的近似解，同时也可以得到相应的特征向量。

很明显可以从表3.4的算法中看出每次在做矩阵L以及向量Vj的乘法时需要耗费大量的时间，因为在做向量和矩阵乘法时平均每次需要对矩阵的每一行都乘上列矩阵，这就需要n次的计算，而在算法中需要很多次这样矩阵相乘的运算，因此增加了计算时间，所以需要通过并行运算来加快算法速度。

从这个方向来想：既然每次都需要计算矩阵的行和向量的乘积，为何不将矩阵的行拆开然后放到map中去，这样每次只需要计算某个map和向量的乘积即可，因此使用MapReduce框架可以实现并行计算矩阵与向量的乘法。



**图3.6 并行计算特征向量**

### 并行实现K-Means聚类

由谱聚类算法可知，在通过3.2.3节的特征向量计算后，可以得到特征矩阵，然后就可以使用这特征矩阵进行标准化，最后得到的标准化矩阵中的列向量便可以拿来聚类成k个不同的簇，这样就得到了谱聚类最后的结果，但是谱算法最后实现步骤的K-Means算法并行实现无疑可以更加提高算法的速度，因此需要研究K-Means算法的可并行算法。

首先再次提出单机情况下K-Means算法的实现步骤：

**表3.5 K-Means算法伪代码**

输入：数据集DataSet以及需要聚类的簇类数目K

输出：聚类后的不同簇

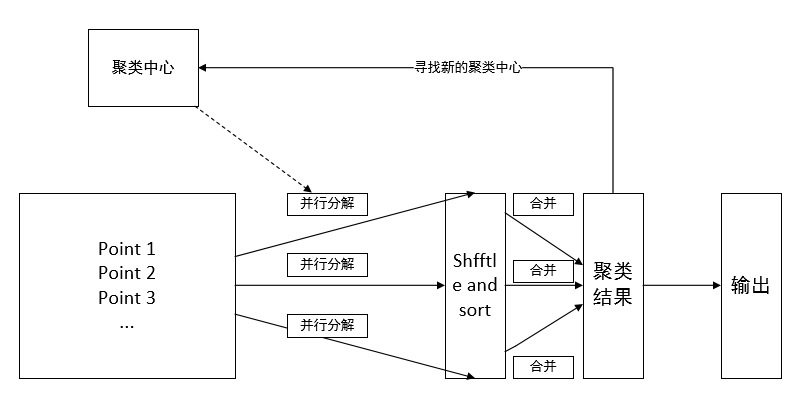
1、任意选择k个样本作为聚类的初始簇中心点

2、根据每个簇类的中心点，将数据集中的每个点分配到最近的簇

3、计算簇类中坐标均值，更新簇类的中心点

4、重复该算法的第2步和第3步，直到算法收敛

从K-Means算法中可以得知在此算法中，需要将每个样本都分配到不同的簇中，对于每个样本数据的分簇是独立的，因此需要将分簇过程进行并行化。



**图3.7 并行K-Means算法**

对于K-Means的并行算法来讲，也需要借助MapReduce框架，Map框架存储的键值对<Key，Value>代表的是距离和坐标，通过整理每个点离聚类中心的距离，然后进行suffle和sort可以得到当前数据应该分配的簇类，Reduce中存储的便是这个新的键值对，然后将相同的key合并，计算出平均值，然后得出新的聚类中心点，n这样在使用K-Means算法时得到了并行的优化。

### 单机与并行下谱聚类算法对比

为了对比单机环境和并行环境，按时间复杂度对单机和并行下的谱聚类算法进行对比。

并行化计算相似矩阵。假定每次在计算一对数据点的相似度为O（1），也就是需要算一次，从3.2.2的方法可以得出，在计算第一个数据点时需要计算n次，第二个数据点需要计算n-1次，依次类推，计算整个相似矩阵则一共需要(n2+n）/2次，而假定集群中一共有m台机器，则所需要的时间复杂度为O（(n2+n）/2m）。

并行化计算k个最小的特征向量。从前面的分析可知，非并行条件下计算k个特征向量所需要的时间复杂度为O（ks+k2n），其中s为矩阵与向量相乘所需要花费的时间，k为特征向量个数，n为矩阵行数，在并行条件下，拉普拉斯矩阵计算的过程被分布到了m台机器上，因此每次在做矩阵和向量的乘法时需要的时间复杂度变成了O（ks/m+k2n）。

并行化实现K-Means算法。因为K-Means算法每次需要与k个中心点进行距离计算，又因为每次需要计算n个点，所以单机情况下的K-Means算法的时间复杂度达到了O（nk2），而并行情况下的K-Means算法被分布到了m台机器上，因此并行情况下的时间复杂度在O（nk2/m）。

因此可以知道并行条件下的总的时间复杂度在O（（n2+n）/2m）+ O(ks/m+k2n ) + O(nk2/m)，而在本章的最初讨论了单机情况下朴素谱聚类算法的时间复杂度大约是O（n3），这是远远大于并行条件的时间复杂度的，因此可以看到在并行条件下谱聚类算法计算速度的显著提升。

## 实验仿真

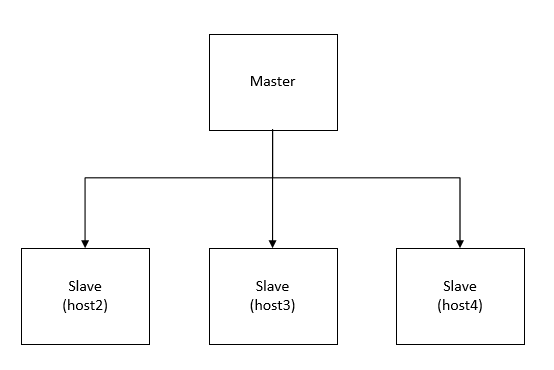
### Hadoop平台搭建

#### 实验环境

实验平台：4台云主机（I5系统，2.0GHz，2G内存），Ubuntu 16.04，Hadoop2.8.5，ssh访问器putty或者其他。

#### 服务器部署

先将4台服务器按照一台Master，三台Slave的格式来部署4台服务器具体如下图3.8:

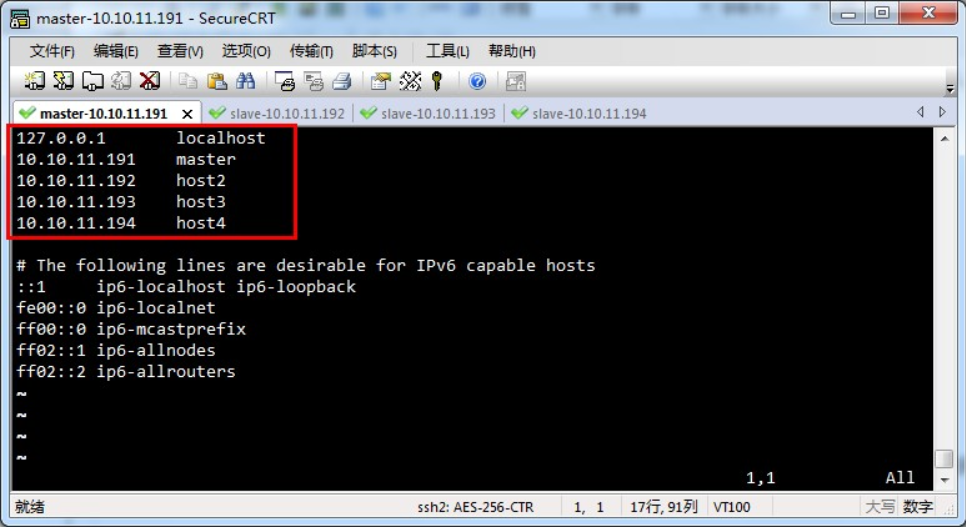


**图3.8 服务器部署办法**

Master用作NameNode，Slave用作DataNode，换言之，如图3.8所展示的一样，Master服务器作为一个控制节点，而host2到host4作为DataNode，用作数据存储和计算节点，负责处理HDFS上的数据块。

#### 配置主机名和IP

需要使用此四台服务器部署在hadoop上，为了便于实施，需要统一四台服务器的IP，因此需要重写host，如下图3.9所示：



**图3.9 修改主机名和IP映射**

#### 配置SSH

为了保证master服务器对slave服务器的控制，需要使用部署SHH以实现服务器之间的通信。

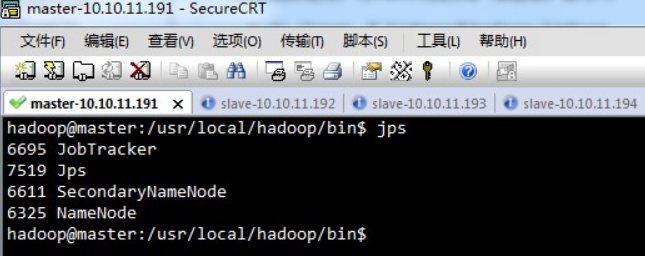
使用ssh-keygen -t rsa在master服务器上和slave服务器上设置密钥存放位置，先在master将公钥放入ssh文件夹下的authorized\_keys，再将此已认证的密钥分别拷贝到所有的slave服务器中的ssh文件夹下，随后给予权限，这样就实现了master服务器的免密登录。

#### 配置jdk和hadoop环境

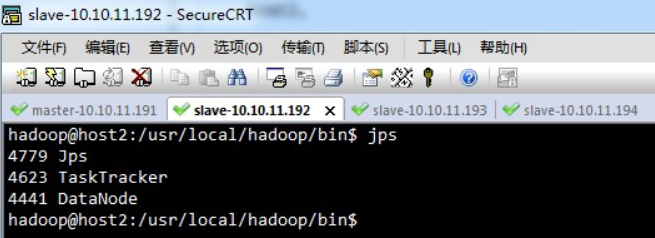
对四个linux服务器的配置JAVA和Hadoop环境变量以及配置文件。需要配置的文件有care-site.xml，hdfs-site.xml，mapred-site.xml，具体配置文件可参见网上教程，此处不再赘述。

#### 测试hadoop是否成功配置

测试之前需要初始化Namenode ,使用如下代码：hadoop namenode –format，并且让hadoop配置生效：source hadoop-env.sh，然后在master服务器与slave服务器中使用jps命令测试，效果图应如3.10与3.11：



**图3.10 master服务器测试**



**图3.11 slave服务器测试**

### 谱聚类算法实现

在有了3.2的谱聚类算法并行基础且有了Hadoop平台后便可以开始进行基于Hadoop平台的谱聚类算法实现了。

#### 数据选取

由于谱聚类算法的实施仍然需要数据的支撑，为了对比单机平台与云计算平台的计算时间差别，因此需要多准备一些数据来进行计算，数据准备的格式见表3.6

**表3.6 数据准备分组**

|  |  |
| --- | --- |
| 数据组数 | 每组数据个数 |
| 10 | 100 |
| 100 | 100 |
| 1000 | 100 |
| 10000 | 100 |
| 100000 | 100 |

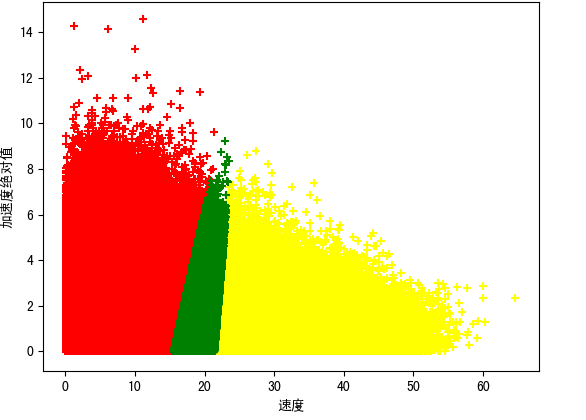
#### 算法实现

在预处理阶段，需要将得到数据进行处理，手上拿到的数据只有坐标点，所以需要对这些数据进行处理，首先对点之间进行距离的计算，然后算出点之间的邻接矩阵，所需要计算的是相似度量矩阵Rnxn，当前有邻接矩阵A以及通过邻接矩阵算出来的度矩阵D，拉普拉斯矩阵L = D-A，根据这些矩阵可以得出相似矩阵。

有了相似矩阵之后就可以开始一步步实施3.2的并行谱聚类算法了，首先需要对该相似矩阵并行计算前k个特征向量，这里使用3.2.3中所描述的Lanczos算法进行实现，最后并行计算K-Means算法即可完成全部的实验工作。

#### 效果展示

首先对一组数据量较小的数据做聚类展示，如下图3.12所示，可以看到这便是并行下的谱聚类算法实现效果图：



**图3.12 云平台下的谱聚类效果图**

从上面的效果图以及在2.4节的谱聚类效果图对比中可以看出，实际聚类效果与单机下的谱聚类算法效果并无太大差别，但是这并不是所需要重点关注的地方，因为使用的是云计算平台，所以最需要去对比的是时间，因此需要进行单机平台和云计算平台两种平台下谱聚类算法的效果，效果对比表见表3.8：

**表3.8 数据准备分组**

|  |  |  |  |
| --- | --- | --- | --- |
| 数据组数 | 每组数据个数 | 单机运行时间 | 云计算平台运行时间 |
| 10 | 100 | 1s以内 | 7s |
| 100 | 100 | 4s | 21s |
| 1000 | 100 | 35s | 14s |
| 10000 | 100 | 104s | 38s |
| 100000 | 100 | 8947s | 540s |

通过表3.8可知，在单机环境下和Hadoop平台的计算速度是不同的，当数据较小的时候，由于云计算平台需要对三部分的算法都进行并行运算，因此当数据较小时，这些部分反而会降低运行效率，且在云平台启动时也需要耗费一定的时间，所以在数据较少时，云计算平台所花费的时间反而更加多，而当数据较多时，云平台的计算效率就体现出来了，因此可以得出结论，当样本数据大大增加时，云计算平台无疑会降低谱聚类算法所需要的时间。

## 本章小结

本章主要讨论了基于云计算平台的并行谱聚类算法，首先讨论了基于并行和基于采样两种方式的实现，然后讨论了基于并行的Hadoop平台的可行性。

之后在3.1节详细研究了Hadoop平台的HDFS分布式系统，MapReduce框架以及HBase这一分布式数据库；在3.2节探讨了谱聚类算法的并行部分，包括相似矩阵的并行计算，特征向量的并行计算，K-Means算法的并行实现三个部分；最后在3.3节对Hadoop平台下的谱聚类并行实现做出了实验仿真，并比较了单机环境和云计算平台下谱聚类算法的时间消耗，得出云计算平台在数据量打的情况下更优的结论。

通过这一章的研究可以发现，在Hadoop云平台下的谱聚类算法实现后能够很明显的提升速度，尤其是在数据量较大的时候这种效果会更加的明显，因此采用云计算平台来实现谱聚类算法，并对数据集进行聚类，将聚类好的结果提取出来。

# 总结与展望

## 总结

近些年来，我国的车辆数量飞速上涨，随之而来的就是各种事故风险的发生率越来越高，为了有效的规避掉这些风险，国内外许多学者也在此方面上做出了很多的研究，有了很多种不同解决问题的办法。在研究了许多相关技术之后，本文基于云计算平台下的谱聚类算法研究驾驶行为-安全评估模型，主要做了以下工作：

1. 分析了本课题的研究意义和目的，研究了一些国内外观点与本课题相关的文献，大致确定了本文的主要研究方向和技术方法。
2. 基于相关系数方法提取特征参数，并重点分析了K-Means，Bi-KMeans，谱聚类三种算法，比较了谱聚类算法相较于其他两种算法的优势，确定了聚类算法。
3. 研究了Hadoop云平台，分析了Hadoop云平台中各层的技术，研究了谱聚类算法的并行性和优异性，提炼了谱聚类算法中的可并行部分，并详细比较了单机环境和云平台环境下谱聚类算法的时间消耗。

## 展望

虽然本文实现了基于车辆网数据的驾驶行为模型研究，但是本文的工作还有很多需要进一步的研究：

1. 本文在特征参数的选取上太过于绝对，只是单纯的选取的速度与加速度两种参数，并没有考虑其他因素，如天气，路况等，这些不同的条件下其实都需要进行特殊考虑，但是由于收集的数据的局限性无法做出此类判断，因此在今后的研究中可以考虑加入以上因素。
2. 本文采用的聚类算法之间和分类器之间虽然有所对比，但也只是粗略的对比，并没有做出极为细致的对比，希望可以在今后的研究中利用更加详实的数据进行对比。

相信进过进一步的研究之后，本文的设计和研究将会更加的深入，系统将会更加的完善，由于研究水平有限，因此文中有许多不足之处，希望大家批评指正。

# 参考文献

1. 平行之家. 想知道我国汽车保有量吗？截至2018年底，直逼2.4亿辆. <https://baijiahao.baidu.com/s?id=1622721747462873661&wfr=spider&for=pc>
2. 肖献强，任春燕，王其东. 基于隐马尔克夫模型的驾驶行为.预测方法研究[J]. 中国机械工程， 2013.21-25
3. 梁军，姚明. 基于多Agent和驾驶员行为的汽车追尾预警模型. 2009海峡两岸智能型运输系统学术研究会， 2009.3-4
4. Wang Rui, Lukic Srdjan M. Review of driving conditions prediction and driving style recognition based control algorithms for hybrid electric vehicles. Vehicle Power and Propulsion Conference(VPPC)，2011 IEEE，2011.1-7
5. Eren Haluk, Makinist, Akin Erhan, etal. Estimating driving behavior by a smartphone. Intelligent Vehicles Symposium(IV), 2012 IEEE, 2012.234-239
6. Salameh N，Challita G，Mosusset S，etal. Collaborative postioning and embedded multi-sensors fusion cooperation in advanced driver assistance system[J]. Transportation Research Part C:Emerging Technologies, 2013.197-213
7. Qi Geqi, Du Yiman, Wu Jianping, etal. Leveraging longitudinal driving behaviour with data mining techniques for driving analysis. Intelligent Transport Systems,IET, 2015.792-801
8. 王登峰，邓阳庆，刘延林，等.汽车使用诸因素对燃油经济性的影响分析和试验研究[J]. 公路交通科技，2003.155-158
9. 高振海, 段立飞. 仿人小脑神经网络结构的驾驶员方向控制行为建模[J]. 汽车工程, 2012(4):288-291.
10. 刘雁飞, 吴朝晖. 驾驶ACT-R认知行为建模[J]. 浙江大学学报(工学版), 2006, 40(10):1657-1662.
11. 孙川, 吴超仲, 褚端峰, et al. 基于车联网数据挖掘的营运车辆 驾驶速度行为聚类研究[J]. 交通运输系统工程与信息, 2015, 15(6):82-87.
12. 朱波. 驾驶行为在车队安全管理中的研究与应用[D] 2016：13-15
13. 张开. 基于云计算的驾驶行为分析[D]. 2016.
14. 李志闯. 基于Hadoop的并行谱聚类算法实现[D]. 华南理工大学.
15. 杨煜. 基于Hadoop MapReduce并行近似谱聚类算法研究与实现[D]. 2014.