文本复制检测报告单(全文对照)

№:ADBD2018R_2018050917371820180518091825428659971212　　检测时间:2018-05-18 09:18:25

检测文献:　1526567738385_陶超权_出租车营运大数据分析

作者:　陶超权

检测范围:　中国学术期刊网络出版总库

中国博士学位论文全文数据库/中国优秀硕士学位论文全文数据库

中国重要会议论文全文数据库

中国重要报纸全文数据库

中国专利全文数据库

图书资源

优先出版文献库

大学生论文联合比对库

互联网资源(包含贴吧等论坛资源)

英文数据库(涵盖期刊、博硕、会议的英文数据以及德国Springer、英国Taylor&Francis 期刊数据库等)

港澳台学术文献库

互联网文档资源

CNKI大成编客-原创作品库

个人比对库

时间范围:　1900-01-01至2018-05-18

检测结果

总文字复制比：24.8%　　跨语言检测结果：0%

去除引用文献复制比：24.2%　　去除本人已发表文献复制比：24.8%

单篇最大文字复制比：9.1%（基于Web的校园办公自动化系统设计与实现）

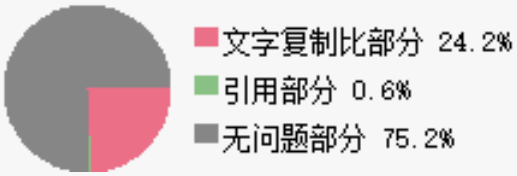
重复字数：[4842]　　总段落数：[6]

总字数：[19543]　　疑似段落数：[5]

单篇最大重复字数：[1783]　　前部重合字数：[272]

疑似段落最大重合字数：[2097]　　后部重合字数：[4570]

疑似段落最小重合字数：[142]



指　标：☐疑似剽窃观点　☑疑似剽窃文字表述　☐疑似自我剽窃　☐疑似整体剽窃　☐过度引用

表格：0　　公式：0　　疑似文字的图片：0　　脚注与尾注：0

7.7%（142）　中英文摘要等（总1856字）

8.1%（192）　第一章引言（总2385字）

44.9%（2035）　第二章大数据隐私保护及数据挖掘相关技术（总4534字）

46.6%（2097）　第三章数据获取及预处理（总4500字）

7.7%（376）　第四章基于MapReduce框架的热点区域挖掘及可视化（总4884字）

0%（0）　第五章总结与展望（总1384字）

（注释：无问题部分　文字复制比部分　引用部分）

1. 中英文摘要等　　总字数：1856

相似文献列表　文字复制比：7.7%(142)　疑似剽窃观点：(0)

1	交通与物流工程学院-交工132班-220130752-张起明-乌鲁木齐市火车南站出租车载客率调查班 -《大学生论文联合比对库》- 2017-05-11	7.5%（139） 是否引证：否
2	关于出租车载客地点序列推荐技术的研究 陈轶非;李治军;姜守旭; -《智能计算机与应用》- 2013-12-01	1.6%（30） 是否引证：否

原文内容	相似内容来源

1	此处有 31 字相似 摘要 随着国民经济的发展以及人民生活水平的提高，出租车越来越成为人们出行不可缺少的工具，无论是周内的上班工作还是周末的 出行娱乐，出租车都在扮演着越来越重要的角色。在万物联网的今天， 基本每辆出租车都安装了GPS终端，这些终端装置会定时向数据库发送出租车实时的行驶状态，营运状态等。这些数据将会成为我们用	关于出租车载客地点序列推荐技术的研究 陈轶非;李治军;姜守旭; -《智能计算机与应用》- 2013-12-01（是否引证：否） 1.0引言在人们日常的出行活动中,出租车正扮演着越来越重要的角色。如今,已经有数量众多的现代人选择出租车来作为其代步的出行方式。根据最近一份关于纽约出租车服务的调查[1],结果显示有41%的调 交通与物流工程学院-交工132班-220130752-张起明-乌鲁木齐市火车南站出租车载客率调查 班 -《大学生论文联合比对库》- 2017-05-11（是否引证：否） 1.ds: Taxi; Railway station; Carrying capacity; Urumqi在人们日常的出行活动中，出租车正扮演着越来越重要的角色。如今，已经有数量众多的现代人选择出租车来作为其代步的出行方式。而出租车载客率影响着出租车行业的发展和乘客乘坐出租车的便捷性。乌
2	此处有 111 字相似 区域，为司机和乘客提供个性化推荐。 关键词：出租车，大数据，聚类，MapReduce,MongoDB，DBscan Abstract With the development of the national economy and the improvement of people’s living standards, taxis have become an indispensable tool for people to travel. Taxi	交通与物流工程学院-交工132班-220130752-张起明-乌鲁木齐市火车南站出租车载客率调查 班 -《大学生论文联合比对库》- 2017-05-11（是否引证：否） 1.化和这一区域交通系统的协调运作和可持续发展等方面都具有非常重要的意义。关键词：出租车；火车站；载客率；乌鲁木齐Abstract: with the development of economy and the improvement of people's living standard, the taxi passengers can fully according to the need to provide flexib

指　标

疑似剽窃文字表述

1. Abstract
With the development of the national economy and the improvement of people’s living standards, taxis

2. 第一章引言

总字数：2385

相似文献列表　文字复制比：8.1%(192)　疑似剽窃观点：(0)

1	职前教师与在职教师数学教学知识的对比研究 张超(导师：李淑文) -《东北师范大学硕士论文》- 2013-05-01	2.1%（49） 是否引证：否
2	基于空间k-匿名的位置隐私保护技术研究 侯士江(导师：刘国华) -《燕山大学博士论文》- 2014-05-01	2.0%（48） 是否引证：是
3	基于位置服务中用户位置隐私保护关键技术研究 车延辙(导师：何钦铭) -《浙江大学博士论文》- 2013-01-01	2.0%（47） 是否引证：是
4	基于GIS切片和缓存技术在农电管理系统的研究与应用 程树仁(导师：马进;吉振中) -《华北电力大学硕士论文》- 2012-06-01	1.6%（37） 是否引证：否
5	基于MSF的路灯监控系统数据库同步模块的研究与设计 沈磊(导师：朱军) -《安徽大学硕士论文》- 2012-04-01	1.5%（36） 是否引证：否
6	重庆市柑橘种植户对柑橘政策性保险的支付意愿研究 牛玉珊(导师：祁春节) -《华中农业大学硕士论文》- 2012-06-01	1.4%（34） 是否引证：否
7	基于出租车轨迹数据挖掘的推荐模型研究 赵苗苗(导师：赵丹亚) -《首都经济贸易大学硕士论文》- 2015-05-05	1.3%（30） 是否引证：否

原文内容		相似内容来源
1	此处有 52 字相似 护技术，通过对用户C在某一时段内的位置信息进行匿	基于空间k-匿名的位置隐私保护技术研究 侯士江 -《燕山大学博士论文》- 2014-05-01（是否引证：是）

	名和泛化处理，使得在该时段内至少有k个用户都在C所处的位置范围内；侯士江 基于用户—匿名器—LBS架构，分别针对欧氏空间和路网环境的位置隐私保护、常见查询等问题进行了研究[4]； 车延辙围绕着改进算法效率和提高抵御攻击能力的目标，研究移动点对点体系结构下的用户位置隐私保护关键技术[5]；Gupta针	1.中首先系统评估了现有位置隐私保护方法的适用性和有效性，引入了攻击分类，并分析了现有方法的保护目标和抗攻击能力。然后基于用户—匿名器—LBS 架构，分别针对欧氏空间和路网环境的位置隐私保护、常见查询等问题进行了研究。具体内容如下。★首先，提出了空间 k-匿名共匿算法。不仅保证了共匿性要求，而且由于采用通用的空间索引技术，如
2	此处有 51 字相似 内；侯士江基于用户—匿名器—LBS架构，分别针对欧氏空间和路网环境的位置隐私保护、常见查询等问题进行了研究[4]；车延辙 围绕着改进算法效率和提高抵御攻击能力的目标，研究移动点对点体系结构下的用户位置隐私保护关键技术[5]； Gupta针对k-匿名中存在可信第三方的主要限制问题，提出了通过若干个同等级用户相互共享一些位置信息的方法来扩展可信第三	基于位置服务中用户位置隐私保护关键技术研究 车延辙 -《浙江大学博士论文》- 2013-01-01（是否引证：是） 1.第三方是整个系统中计算与通讯的瓶颈,而后者也受困于较低的匿名成功率、较大的通讯开销以及较弱的抵御攻击能力。★本文围绕着改进算法效率和提高抵御攻击能力的目标,研究移动点对点体系结构下的用户位置隐私保护关键技术,研究切入点包括：双向的位置信息收集策略、减少点对点通讯量、放宽用户信任假设条件和考虑语义位置影响等。本文的主要贡献有：
3	此处有 52 字相似 ，同时将一天划分为不同时段，通过使用不同的阈值，来找到最佳乘车热点区域；最后通过高德地图API将乘车热点区域可视化。 论文结构 论文一共分为五个章节。 第一章引言——主要介绍了论文的研究背景和意义，以及国内外研究现状， 包括大数据及其隐私保护的研究现状和出租车数据利用的研究现状，最后介绍了本文的内容和结构。 第二章大数据隐私保护及数据挖	职前教师与在职教师数学教学知识的对比研究 张超 -《东北师范大学硕士论文》- 2013-05-01（是否引证：否） 1.等师范院校培养教育人才给定一个借鉴，也为促进在职教师的成长提供一定的理论依据这就是本研究的意义所在。1.4 论文的结构本论文一共分为 7个章节，第1章节主要是引论部分主要阐述了论文研究的背景、研究的意义、研究的问题。论文的第2章主要是文献综述，阐述了学科教学知识的分类、教师知识的来源、学科教学知识的内在特征以及当前教师发展的困境 重庆市柑橘种植户对柑橘政策性保险的支付意愿研究 牛玉珊 -《华中农业大学硕士论文》- 2012-06-01（是否引证：否） 1.回归法(Back ward: Wald)进行迭代回归，踢除不显著的因素，最终留下影响显著的因素。1.3. 3论文结构本论文一共分为四个章节，第一章导论主要介绍选题的目的、意义、国内外研究进展、研究思路与方法等。第二章主要讨论重庆市相■橘生产的风险区划并以此引出在重庆市开展相橘种 基于出租车轨迹数据挖掘的推荐模型研究 赵苗苗 -《首都经济贸易大学硕士论文》- 2015-05-05（是否引证：否） 1.P和司机偏好相关度IMP,并利用BP神经网络第6页共55页集。1.4.3论文组织结构本文共分为六章,每章的具体内容如下:第一章——绪论,本文的研究背景和意义以及主要研究内容。第二章——出租车轨迹数据挖掘的基本理论和相关技术,对现有的轨迹数据挖掘的基本理论和现有方法进行研究,了解出租车轨迹数据挖
4	此处有 37 字相似 论文一共分为五个章节。 第一章引言——主要介绍了论文的研究背景和意义，以	基于MSF的路灯监控系统数据库同步模块的研究与设计 沈磊 -《安徽大学硕士论文》- 2012-04-01（是否引证：否） 1.文的内容结构安排如下：第一章：简单介绍了项目研究背景及一些开发技术的发展现状和技术背景，然后描述了一下研究的意义，最后介绍了本文的内容结构；第

	及国内外研究现状，包括大数据及其隐私保护的研究现状和出租车数据利用的研究现状，最后介绍了本文的内容和结构。 第二章 大数据隐私保护及数据挖掘相关技术——主要介绍了大数据隐私保护的常见方法，并对数据挖掘做了概述并介绍了其挖掘过程。 第三	二章：首先介绍了路灯监控系统的整体结构，然后针对本文重点研究的数据库同步在系统中的位置和需求进行了分析介绍，并提出了
		基于GIS切片和缓存技术在农电管理系统的研究与应用 程树仁 -《华北电力大学硕士论文》- 2012-06-01（是否引证：否）
		1.1.4.2论文内容结构第一章绪论。主要介绍本文的研究背景、相关技术、国内外研究现状，在此基础上提出了本文要研究的主要内容，最后介绍了本文的内容结构。第二章地理信息系统切片发布原理研究。对地理数据发布技术现状进行了客观全面的分析，在此基础上提出了基于REST架构的地理数

指　标
疑似剽窃文字表述

1. 论文结构
   论文一共分为五个章节。
   第一章引言——主要介绍了论文的研究背景和意义，以及国内外研究现状，

3. 第二章大数据隐私保护及数据挖掘相关技术　　总字数：4534

相似文献列表　文字复制比：44.9%(2035)　疑似剽窃观点：(0)

1	张周中_1120122224_校园一卡通消费异常检测的研究与实现 张周中 -《大学生论文联合比对库》- 2016-06-02	15.7%（710） 是否引证：否
2	DBSCAN聚类算法原理 - xuanyuansen的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	15.7%（710） 是否引证：否
3	2220132708-杨俊波-基于浏览器数据的文本聚类程序设计 杨俊波 -《大学生论文联合比对库》- 2017-06-10	5.9%（269） 是否引证：否
4	1_王震_一种基于密度最大值聚类算法的研究与应用 王震 -《大学生论文联合比对库》- 2017-05-11	5.6%（254） 是否引证：否
5	基于密度的聚类方法研究 陈雪儿 -《大学生论文联合比对库》- 2014-05-23	5.0%（225） 是否引证：否
6	基于密度的聚类方法研究 陈雪儿 -《大学生论文联合比对库》- 2014-05-27	5.0%（225） 是否引证：否
7	基于DBScan算法的网页聚类分析 沈逸帆 -《大学生论文联合比对库》- 2017-05-21	4.8%（218） 是否引证：否
8	探索推荐引擎内部的秘密，第3部分：深入_sofiafighting -《网络（http://blog.sina.com）》- 2012	4.5%（202） 是否引证：否
9	推荐引擎相关算法 - 聚类_W老泉 -《网络（http://blog.sina.com）》- 2013	4.5%（202） 是否引证：否
10	探索推荐引擎内部的秘密，第 3 部分: 深入推荐引擎相关算法 - 聚类_SomeThere -《网络（http://blog.sina.com）》- 2014	4.5%（202） 是否引证：否
11	3130931034-王杰-k-modes聚类算法的实现与应用 王杰 -《大学生论文联合比对库》- 2017-06-12	4.4%（200） 是否引证：否
12	数据挖掘中基于遗传算法的聚类方法应用研究 吴多比(导师：朱晓红) -《重庆大学硕士论文》- 2009-04-01	4.3%（195） 是否引证：否
13	数据挖掘中基于遗传算法的K-means聚类算法的研究及应用 赵松(导师：何熊熊) -《浙江工业大学硕士论文》- 2014-04-08	4.3%（193） 是否引证：否
14	基于划分的聚类算法研究 郑柏杰(导师：杨小帆;何国辉) -《重庆大学硕士论文》- 2005-10-01	4.3%（193） 是否引证：否
15	数字版权管理策略研究 马军军 -《大学生论文联合比对库》- 2017-04-27	4.1%（185） 是否引证：否
16	网格聚类算法的研究	4.1%（185） 是否引证：否

	程伟想(导师：孟建良) -《华北电力大学（河北）硕士论文》- 2008-12-18	
17	瞿安国_1120132189_学术研究趋势分析系统的设计与实现	4.0%（180）
	瞿安国 -《大学生论文联合比对库》- 2017-05-31	是否引证：否
18	基于密度的建筑物聚类分析	3.9%（179）
	王安东 -《大学生论文联合比对库》- 2017-05-29	是否引证：否
19	49-3_基于微博的舆情分析与热点推荐	3.5%（157）
	基于微博的舆情分析与热点推荐 -《大学生论文联合比对库》- 2017-04-06	是否引证：否
20	色谱指纹图谱的智能聚类分析在中医湿证辨别方面的研究	3.4%（156）
	胡琳(导师：黄振国) -《东华大学硕士论文》- 2003-12-01	是否引证：否
21	基于特征向量的个性化推荐算法研究	2.9%（131）
	杜定宇(导师：王茜) -《重庆大学硕士论文》- 2011-04-01	是否引证：否
22	新闻聚合系统设计	2.4%（109）
	徐宇航 -《大学生论文联合比对库》- 2017-05-06	是否引证：否
23	基于支持向量机的电信话务量预测方法	2.4%（109）
	陈电波(导师：徐福仓;吴敏) -《中南大学硕士论文》- 2008-06-30	是否引证：否
24	00343186681013071_王震_一种基于密度最大值聚类算法的研究与应用	2.2%（101）
	郑浩 -《大学生论文联合比对库》- 2017-05-12	是否引证：否
25	高考志愿填报的数据分析研究	2.0%（89）
	杨浩杰(导师：陈志国;刘刚) -《河南大学硕士论文》- 2011-05-01	是否引证：否
26	1495515502393_夏浪伟_20170523查重	1.9%（88）
	夏浪伟 -《大学生论文联合比对库》- 2017-05-23	是否引证：否
27	1495515502393_夏浪伟_20170523查重	1.9%（88）
	夏浪伟 -《大学生论文联合比对库》- 2017-05-23	是否引证：否
28	改进的模糊聚类算法在入侵检测中的应用研究	1.5%（70）
	邹翔(导师：张玉芳) -《重庆大学硕士论文》- 2015-04-01	是否引证：否
29	基于电网运行数据集的电力系统运行评估及优化研究	1.5%（69）
	刘柏林(导师：穆钢) -《华北电力大学(北京)博士论文》- 2017-06-01	是否引证：否
30	个人信息去标识化框架及标准化	1.0%（45）
	谢安明;金涛;周涛; -《大数据》- 2017-09-20	是否引证：否
31	基于深度学习的车辆检测和车牌定位	0.9%（40）
	封晶(导师：任克强) -《江西理工大学硕士论文》- 2017-05-24	是否引证：否
32	面向数据挖掘的隐私保护算法研究	0.7%（33）
	郑少飞(导师：李玲娟) -《南京邮电大学硕士论文》- 2011-03-01	是否引证：否
33	加载隐私保护的网络安全综合管理关键技术研究	0.7%（33）
	马进(导师：李建华) -《上海交通大学博士论文》- 2012-05-01	是否引证：否

	原文内容	相似内容来源
1	此处有 52 字相似 数据挖掘相关技术 大数据隐私保护 1.1.3. 大数据隐私保护概述 隐私[11]指主体没有公开的知识、信息等，根据不同主题，隐私又可分为公共隐私和个人隐私。公共隐私指群体的共同信息或者模式信息，主要针对企业和政府部门。 个人隐私指个人独立的基本资料，如医疗信息，电子档案信息等。大数据在分析过程中如果不对用户敏感信息进行保护，极有可能造成用	加载隐私保护的网络安全综合管理关键技术研究 马进 -《上海交通大学博士论文》- 2012-05-01（是否引证：否） 1.个人身份信息、财务信息、偏好信息、活动状况信息、人口普查、就诊记录、经济调查、个人数据的电子文件、电子档案等。公共隐私则以群体共同特征或模式信息为代表，主要针对企业及政府部门，如安全信息和趋势分析等。图1.2 隐私信息的分类Figure1.2 Classification of P 面向数据挖掘的隐私保护算法研究 郑少飞 -《南京邮电大学硕士论文》- 2011-03-01（是否引证：否） 1.免个人的隐私泄露，又为信息共享提供良好的信息支持是信息安全需要解决的一个重要课题。 根据隐私的主体对象的不同，隐私可分为个人隐私和公共隐私[6]。个人隐私的主体信息包括个人独立的网络资料信息、电子档案信息、电子邮件信息等，以及个人基本状况信息、人口普查信息、信用卡使用记录信息
2	此处有 50 字相似	个人信息去标识化框架及标准化 谢安明;金涛;周涛; -《大

	人隐私指个人独立的基本资料，如医疗信息，电子档案信息等。大数据在分析过程中如果不对用户敏感信息进行保护，极有可能造成用户 信息泄露。如何在不泄露用户隐私的前提下，提高大数据的利用率，是目前大数据研究领域的关键问题之一。 1.1.4. 大数据隐私保护方法 （1）数据发布匿名保护技术 为防止攻击者在数据发布时使用链接攻击获取用户敏感信息，	数据》- 2017-09-20（是否引证：否） 1.企业和其他组织收集的数据中,通常含有个人姓名、电话、证件号码等信息,如果将收集到的原始数据直接进行发布,会导致严重的个人信息泄露。如何在不泄露用户个人信息的前提下,有效开放共享数据,挖掘大数据的价值,是目前大数据研究领域的关键问题。近年来,针对个人信息的去标识化研究获得了很多的关注。所谓去标识化,就是指去除一组可识别数据和数据主体之间关联关系的过程。
3	此处有 99 字相似 导致匿名表产生信息泄露，因而又有人提出了l-多样化技术[13]，使得匿名组里敏感属性的多样性大于或等于l。 （2）数据 水印技术 数据水印技术[14]指将特定的信息嵌入数字信号中，数字信号可以是音频、图片或视频等。若要拷贝有数字水印的信号，所嵌入的信息也会一并被拷贝。数字水印可分为浮现式和隐藏式两种。一般来说，浮现 式的水印通常包含版权拥有者的名称或标志，其所包含的信息可在观看图片或视频时同时被看见。隐藏式的水印是以数字数据的方式加入	数字版权管理策略研究_马军军-《大学生论文联合比对库》- 2017-04-27（是否引证：否） 1.用人人皆知的公开密钥对发送的信息进行加密，安全地传送给商户，然后由商户用自己的私有密钥进行解密。[5]3.3.3数字水印技术数字水印，是指将特定的信息嵌入数字信号中，数字信号可能是音频、图片或是视频等。若要拷贝有数字水印的信号，所嵌入的信息也会一并被拷贝。数字水印可分为浮现式和隐藏式两种。[6]浮现式：是可被看见的水印（visible watermarking），其所包含的信息可在观看图片或视频时同时被看见。一般来说
4	此处有 156 字相似 浮现式和隐藏式两种。一般来说，浮现式的水印通常包含版权拥有者的名称或标志，其所包含的信息可在观看图片或视频时同时被看见。 隐藏式的水印是以数字数据的方式加入音频、图片或视频中，但在一般的状况下无法被看见。隐藏式水印的重要作用之一是保护版权，期望能借此避免或阻止数字媒体未经授权的复制和拷贝。 数据挖掘 1.1.5. 数据挖掘的概念 数据挖掘[15]是一个多学科交叉领域，它融合了数据库技术、人工智能、机器学习、统计学、知识工程、 面向对象方法、信息检索等最新技术的研究成果，从大量数据中提取知识和模式。数据挖掘的实际工作是对大规模数据进行自动或半自动	数字版权管理策略研究_马军军-《大学生论文联合比对库》- 2017-04-27（是否引证：否） 1.般来说，浮现式的水印通常包含版权拥有者的名称或标志。例如视频或图片上面的水印标志，当然电视台的台标也是浮现式的一种。隐藏式：是以数字数据的方式加入音频、图片或视频中，但在一般的状况下无法被看见。隐藏式水印的重要应用之一是保护版权，期望能借此避免或阻止数字媒体未经授权的复制和拷贝。针对数字水印性质的要求：安全性：水印信息应当难以篡改、难以伪造。隐蔽性：水印对感官不可知觉，水印的嵌入不能影响被 改进的模糊聚类算法在入侵检测中的应用研究_邹翔-《重庆大学硕士论文》- 2015-04-01（是否引证：否） 1.掘更为恰当。b.如何减小输入参数的影响,而参数的选择没有简单有效的通用方法。c.如何进一步提高检测率和降低误检率。2.2数据挖掘2.2.1数据挖掘的概念数据挖掘是一个多学科交叉的领域,它融合了统计学、并行计算、机器学习、模式识别、数据库、人工智能等多种学科的理论技术方法[28]。作为一个涵盖多学科的领域,数据挖掘的定义方法有多种。一部分人将数据挖掘看作是“

5	此处有 98 字相似 据库技术、人工智能、机器学习、统计学、知识工程、面向对象方法、信息检索等最新技术的研究成果，从大量数据中提取知识和模式。 数据挖掘的实际工作是对大规模数据进行自动或半自动的分析，以提取过去未知的有价值的潜在信息，例如数据的分组（通过聚类分析）、数据的异常记录（通过异常检测）和数据之间的关系（通过关联式规则挖掘）。 1.1.1. 数据挖掘的过程 （1）数据预处理 在使用挖掘算法计算之前，必须收集数据集。由于数据挖掘只能发现存在于数	瞿安国_1120132189_学术研究趋势分析系统的设计与实现 瞿安国 -《大学生论文联合比对库》- 2017-05-31（是否引证：否） 1.和数据管理方面、数据预处理、模型与推断方面考量、兴趣度度量、复杂度的考虑，以及发现结果、可视化及在线更新等后续处理。数据挖掘的实际工作是对大规模数据进行自动或半自动的分析，以提取过去未知的有价值的潜在信息，例如数据的分组（通过聚类分析）、数据的异常记录（通过异常检测）和数据之间的关系（通过关联式规则挖掘）。这通常会涉及到一部分数据库的技术，例如构建树图、对于信息的索引等。这些潜在信息可通过对输入数据处理之后的总结来呈现，之后
6	此处有 50 字相似 集中发现具有某些相同特征的数据子集；分类，将已知的分类或者结构应用于未知的数据，对其进行分类；回归，寻找用最小错误对数据 建模的函数；汇总，提供更紧凑的数据表示方法，包括生成可视化和报表。 （3）结果验证 如果数据挖掘 方法被误用，可能会产生一个看似很重要实则在其他新的数据集上不能复现的结果，这种结果通常是使用了太多假设或者不符合假设统计	瞿安国_1120132189_学术研究趋势分析系统的设计与实现 瞿安国 -《大学生论文联合比对库》- 2017-05-31（是否引证：否） 1.系统都包含的自动将用户收到的邮件根据特定的特征划分为“垃圾邮件”和“合法邮件”。回归：试图找到能够以最小误差对该数据建模的函数。汇总：提供了一个更紧凑的数据集表示，包括生成可视化和报表。2.1.3数据挖掘的过程数据挖掘主要包括以下几个部分的内容：数据选择、数据预处理、数据缩减、确定KDD的目标、确定知识发现算法、数据挖
7	此处有 56 字相似 聚合。 （2）分割聚类算法 首先将数据集分割为k个划分，然后通过迭代某特定算法，使得某个指标最优以达到最终结果。这种 聚类算法又可详细划分为基于图论的聚类、基于网格的聚类、基于平方差的聚类和基于密度的聚类。 基于图论的聚类将聚类 转换为组合优化问题，然后利用图论相关知识，并结合启发式算法来求解原问题。其一般做法是，先构造数据集的最小生成树，然后逐步	网格聚类算法的研究 程伟想 -《华北电力大学（河北）硕士论文》- 2008-12-18（是否引证：否） 1.已成为数据挖掘研究中一个非常活跃的研究课题。 迄今为止，人们已经提出了许多聚类算法,如基于划分的聚类算法、基于层次的聚类算法、基于密度的聚类算法、基于网格的聚类算法和基于模型的聚类算法等 。所有这些算法都试图通过不同的途径实现对大规模数据库的有效聚类，但总的来说，都没有取得理想的效果。可以说，对大 基于密度的建筑物聚类分析 王安东 -《大学生论文联合比对库》- 2017-05-29（是否引证：否） 1.物上节点si在线的左侧。2.1.4建筑物聚类方法及意义在对建筑物的聚类时，常常利用空间聚类的方法，包括基于密度的聚类、基于划分的聚类、基于网格的聚类、基于模型的聚类、基于层次的聚类和基于约束的聚类。建筑物聚类的最终结果是在空间聚类的的指导下，根据人的视觉感知的规律和约束，在地图比例尺由大变小时，发现表示居民地符号的演 数据挖掘中基于遗传算法的K-means聚类算法的研究及应用 赵松 -《浙江工业大学硕士论文》- 2014-04-08（是否引

		证：否） 1.构多种多样，所以没有一种聚类算法可以适用所有的多维数据集。根据数据在聚类中的应用规则，聚类算法大致分为：基于层次的聚类算法、基于划分的聚类算法、基于密度旳聚类算法、基于网格的聚类算法和其他聚类算法丨、1)基于层次的聚类算法基于层次的聚类算法又称为树聚类算法。它通过数据之间的关联规则，反复分裂 2220132708-杨俊波-基于浏览器数据的文本聚类程序设计杨俊波-《大学生论文联合比对库》- 2017-06-10（是否引证：否） 1.高的相似性。2.2 常见的聚类算法聚类分析是一个很热门的研究领域,目前已经出现了很多种聚类方法。总体上可分为层次聚类法、划分聚类法、基于密度的聚类方法、基于模型的聚类方法和基于网格的聚类方法等。2.2.1 层次聚类法层次聚类法（Hierarchical Methods）对给定的数据集进行层次的分解，直到满足某
8	此处有 261 字相似 识，并结合启发式算法来求解原问题。其一般做法是，先构造数据集的最小生成树，然后逐步删除最小生成树中最大长度的边，形成更多聚类。 基于网格的聚类算法[17]采用空间驱动的方法，把嵌入空间划分成独立于输入对象分布的单元。这种方法使用一种多分辨率的网络数据结构。它将对象空间量化成有限数目的单元，这些网格形成了网格结构，所有的聚类结构都在该结构上进行。这种方法的主要优点是处理速度快，其处理时间独立于数据对象数，仅依赖于量化空间中的每一维的单元数。基本思想就是将每个属性的可能值分割成许多相邻的区间，创建网格单元的集合（我们假设属性值是连续的，序数的，区间的）。每个对象落入一个网格单元，网格单元对应的属性空间包含该对象的值。 基于平方差的聚类 算法的主要思想是逐步优化聚类结果，通过迭代想目标数据集重复向聚类中心进行聚类以得到最优解。该聚类方法又可详细分为最邻近聚	1_王震_一种基于密度最大值聚类算法的研究与应用 王震-《大学生论文联合比对库》- 2017-05-11（是否引证：否） 1.，但需要密度参数作为终止条件主要的聚类算法有：DBSCAN，OPTICS，DENCLUE等。(4)基于网格的方法基于网格的聚类方法采用空间驱动的方法，把嵌入空间划分成独立于输入对象分布的单元。基于网格的聚类方法使用一种多分辨率的网络数据结构。它将对象空间量化成有限数目的单元，这些网格形成了网格结构，所有的聚类结构都在该结构上进行。这种方法的主要优点是处理速度快，其处理时间独立于数据对象数，而仅依赖于量化空间中的每一维的单元数。总结一下就是：将对象空间量化为有限数目的单元，形成一个网状结构，所有聚类都在这个网状结构上进行。基本思想就是将每个属性的可能值分割成许多 2.元数。总结一下就是：将对象空间量化为有限数目的单元，形成一个网状结构，所有聚类都在这个网状结构上进行。基本思想就是将每个属性的可能值分割成许多相邻的区间，创建网格单元的集合（我们假设属性值是连续的，序数的，区间的）。每个对象落入一个网格单元，网格单元对应的属性空间包含该对象的值。(5)基于模型的方法我们之所以在数据集上进行聚类分析，是因为我们假定数据集中的对象属于不同的固有类别，即聚类分析的目的就是发现隐藏的 2220132708-杨俊波-基于浏览器数据的文本聚类程序设计杨俊波-《大学生论文联合比对库》- 2017-06-10（是否引证：否） 1. 基于网格的聚类方法基于网格的聚类方法（Grid-based method）：基于网格的聚类算法，使用一个网格结构，把空间划分成独立于输入对象分布的单元。基于网格的聚类方法使用一种多分辨率的网络数据结构。它将对象空间量化成有限数目的单元，这些网格形成了网格结构，所有的聚类结构都在该结构上进行。这种方法的主要优点是处理速度快，其处理时间独立于数据对象

数，而仅依赖于量化空间中的每一维的单元数。简单来说基于网格的聚类方法就是将对象空间量化为有限数目的单元，形成一个网状结构，所有聚类都在这个网状结构上进行。基本

2.数。简单来说基于网格的聚类方法就是将对象空间量化为有限数目的单元，形成一个网状结构，所有聚类都在这个网状结构上进行。基本思想就是将每个属性的可能值分割成许多相邻的区间，创建网格单元的集合（我们假设属性值是区间的，连续的，序数的）。每个对象落入一个网格单元，网格单元对应的属性空间包含该对象的值。基于网格的聚类方法有较高的数据处理速度，因此适用于大数据集和高维数据集。2.3 K-means聚类算法由2.2.2节可知K-m

3130931034-王杰-k-modes聚类算法的实现与应用_王杰-《大学生论文联合比对库》- 2017-06-12（是否引证：否）

1.方法不同，它将簇定义为密度相连的点的最大集合，能够把具有足够高密度的区域划分为簇，并可在噪声的空间数据库中发现任意形状的聚类。基于网格的聚类方法采用空间驱动的方法，把嵌入空间划分成独立于输入对象分布的单元。基于网格的聚类方法使用一种多分辨率的网络数据结构。它将对象空间量化成有限数目的单元，这些网格形成了网格结构，所有的聚类结构都在该结构上进行。这种方法的主要优点是处理速度快，其处理时间独立于数据对象数，而仅依赖于量化空间中的每一维的单元数。将对象空间化为有限数目单元，形成一个网状结构，所有聚类都在这个网状结构上进行。基本思想就是将每个属性的可能值分割成许多相邻

2.成一个网状结构，所有聚类都在这个网状结构上进行。基本思想就是将每个属性的可能值分割成许多相邻的区间，创建网格单元的集合。每个对象落入一个网格单元，网格单元对应的属性空间包含该对象的值。3.2.3 评估度量DBIDavies-Bouldin指数，简称DBI。它是一种评估度量的聚类算法。是计算类内距离之

数据挖掘中基于遗传算法的K-means聚类算法的研究及应用_赵松-《浙江工业大学硕士论文》- 2014-04-08（是否引证：否）

1.利用密度估计进行聚类的算法。4)基于网格的聚类算法3浙江工业大学硕士学位论文 基于网格的聚类算法采用空间驱动的方法，把嵌入空间划分成独立于输入对象分布的单元。这类算法采用的是一种多分辨率的网格数据结构。基于网格聚类算法的主要思想是：首先将空间划分成独立的类似网格的单元，然

2.次的结点或簇的数目。15浙江工业大学硕士学位论文 2.6.5基于网格方法基于网格的聚类[37】方法使用一种多分辨率的网格数据结构。它将对象空间量化成有限数目的单元，这些单元形成了网格结构，所有的聚类操作都在该结构上进行。这种方法的主要优点是处理速度快，其处理时间独立于数据对象数，而仅依赖于量化空间中每一个维上的单元数。基于网格聚类方法的经典算法

为STING算法。STINGPSI是基于网格方法聚类的一种经典聚类算法，其采用了网格的多

49-3_基于微博的舆情分析与热点推荐 基于微博的舆情分析与热点推荐 -《大学生论文联合比对库》- 2017-04-06（是否引证：否）

1.密度的聚类方法主要有三种代表性的方法，分别是DBSAN、OPTICS和DENCLUE。3.0.4 基于网格的方法基于网格的聚类方法采用空间驱动的方法，把嵌入空间划分成独立于输入对象分布的单元。这种方法使用一种多分辨率的网格数据结构，将对象空间量化成有限数目的单元，形成网格结构，所有的聚类都在该结构上进行。这种方法处理速度快，处理时间独立于数据对象数，而仅依赖于量化空间中每一维上的单元数。3.0.5 增量式的聚类方法互联网的发展导致数据爆炸式地增长，这对大量数据的存储提出了巨大的挑战。如何解决有限空间下

基于密度的建筑物聚类分析 王安东 -《大学生论文联合比对库》- 2017-05-29（是否引证：否）

1.聚类算法将空间量化为有限数目的单元，形成一个网格结构，所有聚类都在网格上进行。基于网格的聚类方法采用空间驱动的方法，把嵌入空间划分成独立于输入对象分布的单元。基于网格的聚类方法使用一种多分辨率的网络数据结构。它将对象空间量化成有限数目的单元，这些网格形成了网格结构，所有的聚类结构都在该结构上进行。这种方法的主要优点是处理速度快，其处理时间独立于数据对象数，而仅依赖于量化空间中的每一维的单元数。总结一下就是：将对象空间量化为有限数目的单元，形成一个网状结构，所有聚类都在这个网状结构上进行。其典型算法有STING算法以及CLIQU

色谱指纹图谱的智能聚类分析在中医湿证辨别方面的研究 胡琳 -《东华大学硕士论文》- 2003-12-01（是否引证：否）

1.密度函数的局部最大。2 .4基于网格的方法 基于网格的聚类方法采用一个多分辨率的网格数据结构。它将空间量化为有限数目的单元，这些单元形成了网格结构，所有的聚类操作都在网格上进行，这种方法的主要优点是处理速度快，其处理时间独立于数据对象的数目，仅依赖于量化空间中每一维上的单元数目。 基于网格方法的有代表性的例子包括:STING，它利用存储在网格单元中的统计信息;Wav

基于划分的聚类算法研究 郑柏杰 -《重庆大学硕士论文》- 2005-10-01（是否引证：否）

1.影响。 3.4.4 基于网格的方法 基于网格的聚类方法采用多分辨率的网格数据结构。是把对象空间量化为有限数目的单元，形成一个网格结构，所有操作都在这个网格结构上进行。这种方法的主要优点是处理速度快，处理时间独立于数据对象的数目，只与量化空间中每一维的单元数目有关[10]。这类算法主要包括 S TING 算法和 CLIQUE 算法。

基于支持向量机的电信话务量预测方法 陈电波 -《中南大学硕士论文》- 2008-06-30（是否引证：否）

1.状的簇。 (4)基于网格的方法(grid－based method) 基于网格的聚类方法采用一个多分辨率的网格数据结构，把对象空间量化为有限数目的单元，形成了一个网格结构。所有的聚类操作都在这个网格结构上进行。这种方法的主要优点是它的处理速度很快，其处理时间独立于数据对象的数目，只与量化空间中每一维的单元数目有关。 (5)基于模型的方法(model－based method)中南大学硕士学位论文

网格聚类算法的研究 程伟想 -《华北电力大学（河北）硕士论文》- 2008-12-18（是否引证：否）

1.响聚类结果的质量。 152.3 基于网格的聚类算法 基于网格的聚类方法采用了网格的数据结构，它将空间量化为有限数目的单元，这些单元形成了网格结构，所有的聚类操作都在网格上进行。这种方法的主要优点是处理速度快，其处理时间独立于数据对象的数目，仅依赖于量化空间中每一维上的单元数目。 基于网格方法包括：STING，它利用存储在网格单元中的统计

数据挖掘中基于遗传算法的聚类方法应用研究 吴多比 -《重庆大学硕士论文》- 2009-04-01（是否引证：否）

1.离，基于产生的次序信息，OPTICS 来抽取聚类。 3.5.4 基于网格的方法 基于网格的聚类方法采用多分辨率的网格数据结构，把对象空间量化为有限数目的单元，形成一个网格结构，所有操作都在这个网格结构上进行。这种方法的主要优点是处理速度快，处理时间独立于数据对象的数目，只与量化空间中每一维的单元数目有关。代表性的算法是 STING 算法和 CLIQUE 算法。 ①STING(Statistical Inform

新闻聚合系统设计 徐宇航 -《大学生论文联合比对库》- 2017-05-06（是否引证：否）

1.组，直到分组达到优化条件为止。代表性算法为：K-MEANS 算法、K-MEDOIDS 算法。（3）基于网格的方法基于网格的聚类方法采用空间驱动的方法，把嵌入空间划分成独立于输入对象分布的单元。基于网格的聚类方法使用一种多分辨率的网络数据结构。它将对象空间量化成有限数目的单元，这些网格形成了网格结构，所有的聚类结构都在该结构上进行。代表性算法有：WAVE-CLUSTER 算法、STING 算法。（4）基于密度的方法该方法将基于密度对数据进行划

00343186681013071_王震_一种基于密度最大值聚类算法的研究与应用 郑浩 -《大学生论文联合比对库》- 2017-05-12（是否引证：否）

1.基于网格的方法顾名思义，基于网格的聚类方法是在网状结构上进行的聚类算法，这种网状结构是通过吧对象空间量化为有限数目的单元形成的。基本思想就是将每个属性的可能值分割成许多相邻的区间，创建网格单元的集合（假设属性值是连续的、序数的和区间的）。每个对象落入一个网格单元，网格单元对应的属性空间包含该对象的值。(5)基于模型的方法我们之所以在数据集上进行聚类分析，是因为我们假定数据集中的对象属于不同的固有类别，即聚类分析的目的就是发现隐藏的

高考志愿填报的数据分析研究_杨浩杰 -《河南大学硕士论文》- 2011-05-01（是否引证：否）

1. 基于网格的聚类方法常用的主要算法有：STING，CLIQUE，Wave Cluster 等。这种算法把对象空间量化为有限数目的单元，形成了一个网格结构。所有的聚类操作都在这个网格结构上进行。这种方法的主要优点是它的处理速度很快，其处理时间独立于数据对象的数目，只与量化空间中每一维的单元数目有关[41]。 25 3.6 群集智能算法 人工智能在经历了 20 世纪 80

1495515502393_夏浪伟_20170523查重_夏浪伟 -《大学生论文联合比对库》- 2017-05-23（是否引证：否）

1.速度快，处理时间短。总结一下就是：将对象空间量化为有限数目的单元，形成一个网状结构，所有聚类都在这个网状结构上进行。基本思想就是将每个属性的可能值分割成许多相邻的区间，创建网格单元的集合（这里假设属性值是连续的，序数的，区间的）。每个对象落入一个网格单元，网格单元对应的属性空间包含该对象的值。将每个点赋予其所在的网格的值，然后统计出每个网格有多少个点在里面，然后划分成一定的等级，来得出点的分布特征。本文使用

1495515502393_夏浪伟_20170523查重_夏浪伟 -《大学生论文联合比对库》- 2017-05-23（是否引证：否）

1.速度快，处理时间短。总结一下就是：将对象空间量化为有限数目的单元，形成一个网状结构，所有聚类都在这个网状结构上进行。基本思想就是将每个属性的可能值分割成许多相邻的区间，创建网格单元的集合（这里假设属性值是连续的，序数的，区间的）。每个对象落入一个网格单元，网格单元对应的属性空间包含该对象的值。将每个点赋予其所在的网格的值，然后统计出每个网格有多少个点在里面，然后划分成一定的等级，来得出点的分布特征。本文使用

基于电网运行数据集的电力系统运行评估及优化研究_刘柏林 -《华北电力大学(北京)博士论文》- 2017-06-01（是否引证：否）

1.主要优点是处理速度快，且计算的复杂度不依赖于数据集的数据对象个数，而只和每个维度网格的个数相关[82]。基本思想就是将每个属性的可能值分割成许多相邻的区间，创建网格单元的集合，每个对象落入一个网格单元，网格单元对应的属性空间包含该对象的值。统计信息网格 (Statistical Info

2.每个维度网格的个数相关[82]。基本思想就是将每个属性的可能值分割成许多相邻的区间，创建网格单元的集合，每个对象落入一个网格单元，网格单元对应的属性空间包含该对象的值。统计信息网格 (Statistical Information Grid，STING)算法是一种基于网格的多分辨

瞿安国_1120132189_学术研究趋势分析系统的设计与实现瞿安国 -《大学生论文联合比对库》- 2017-05-31（是否引证：否）

		1.2.5.4基于网格的聚类算法经典的基于网格的聚类算法有：STING算法、WaveCluster算法等，它们的特点是速度快，处理速度独立于数据对象，只与量化空间中每一单元的数目有关。基于网格的聚类算法主要思想是把对象空间化为有限数目的单元，从而得到一个网格结构，然后在这个网格上进行聚类。
9	此处有 207 字相似 法概率聚类算法、K-medoids[18]算法和K-means算法。其中K-means是目前应用最多的一种聚类算法，给定划分数量 k并创建一个初始划分，从数据集中随机地选择 k 个对象，每个对象初始地代表了一个簇中心（Cluster Centroid）。对于其他对象，计算其与各个簇中心的距离，将它们划入距离最近的簇。采用迭代的重定位技术，尝试通过对象在划分间移动来改进划分。所谓重定位技术，就是当有新的对象加入簇或者已有对象离开簇的时候，重新计算簇的平均值，然后对对象进行重新分配。这个过程不断重复，直到各簇中对象不再变化为止。 基于密度的聚类算法根据数据的分布密度，合并密度超过某个阈值的相邻区域为一个区域，可以在有噪音的数据中发现各种形状和各种大	探索推荐引擎内部的秘密，第3部分：深入_sofiafighting - 《网络（http://blog.sina.com）》- （是否引证：否） 1.足两个要求： 每个组至少包含一个对象 每个对象必须属于且仅属于一个组。 K 均值的基本原理是这样的，给定 k，即要构建的划分的数目， 首先创建一个初始划分，随机地选择 k 个对象，每个对象初始地代表了一个簇中心。对于其他的对象，根据其与各个簇中心的距离，将它们赋给最近的簇。 然后采用一种迭代的重定位技术，尝试通过对象在划分间移动来改进划分。所谓重定位技术，就是当有新的对象加入簇或者已有对象离开簇的时候，重新计算簇的平均值，然后对对象进行重新分配。这个过程不断重复，直到没有簇中对象的变化。当结果簇是密集的，而且簇和簇之间的区别比较明显时，K 均值的效果比较好。对于处理大数据集，这个算法是相对可伸缩的和高效 推荐引擎相关算法 - 聚类_W老泉 - 《网络（http://blog.sina.com）》- （是否引证：否） 1.足两个要求： 每个组至少包含一个对象 每个对象必须属于且仅属于一个组。 K 均值的基本原理是这样的，给定 k，即要构建的划分的数目， 首先创建一个初始划分，随机地选择 k 个对象，每个对象初始地代表了一个簇中心。对于其他的对象，根据其与各个簇中心的距离，将它们赋给最近的簇。 然后采用一种迭代的重定位技术，尝试通过对象在划分间移动来改进划分。所谓重定位技术，就是当有新的对象加入簇或者已有对象离开簇的时候，重新计算簇的平均值，然后对对象进行重新分配。这个过程不断重复，直到没有簇中对象的变化。当结果簇是密集的，而且簇和簇之间的区别比较明显时，K 均值的效果比较好。对于处理大数据集，这个算法是相对可伸缩的和高效 探索推荐引擎内部的秘密，第 3 部分: 深入推荐引擎相关算法 - 聚类_SomeThere - 《网络（http://blog.sina.com）》- （是否引证：否） 1.足两个要求： 每个组至少包含一个对象 每个对象必须属于且仅属于一个组。 K 均值的基本原理是这样的，给定 k，即要构建的划分的数目， 首先创建一个初始划分，随机地选择 k 个对象，每个对象初始地代表了一个簇中心。对于其他的对象，根据其与各个簇中心的距离，将它们赋给最近的簇。 然后采用一种迭代的重定位技术，尝试通过对象在划分间移动来改进划分。所谓重定位技术，就是当有新的对象加入簇或者已有对象离开簇的时候，重新计算簇的平均值，然后对对象进行重新分配。这个过程不断重复，直到没有簇中对象的变化。当结果簇是密集的，而且簇和簇之间的区别比较明显时，K 均值的效果比较好。对于处理大数据集，这个算法

		是相对可伸缩的和高效 基于特征向量的个性化推荐算法研究 杜定宇 -《重庆大学硕士论文》- 2011-04-01（是否引证：否） 1.①首先创建初始划分，算法创建 k 个簇，随机地选择 k 个对象，每个对象代表了一个簇中心。计算剩余的对象与各个簇中心的距离，将它们赋给最近的簇。 ②然后采用一种迭代的重定位技术，尝试通过对象在划分间移动来改进划分。所谓重定位技术，就是当有新的对象加入簇或者已有对象离开簇的时候，重新计算簇的平均值，然后对对象进行重新分配。这个过程不断重复，直到簇中对象不再发生变化。 当结果簇是密集的，而且簇和簇之间的区别比较明显时，K 均值的效果比较好。对于处理大数据集，这个算法是 基于划分的聚类算法研究 郑柏杰 -《重庆大学硕士论文》- 2005-10-01（是否引证：否） 1.，基于划分的方法首先创建一个初始划分，通常采用的方法是随机选取 k 个数据对象作为初始点，然后采用一种迭代的重定位技术，尝试通过对象在划分间移动来改进划分，采用的准则是：在同一个聚簇中的数据对象尽可能相近或相关，不同的聚簇中的数据对象尽可能相异或无关 数据挖掘中基于遗传算法的聚类方法应用研究 吴多比 -《重庆大学硕士论文》- 2009-04-01（是否引证：否） 1.划分的方法首先创建一个初始划分，通常采用的方法是随机选取 k 个数据对象作为初始聚类中心点，然后采用一种迭代的重定位技术，尝试通过对象在划分间移动来改进划分，采用的准则是：在同一个簇中的数据对象尽可能相似，不同的簇中的数据对象尽可能相似。 根据对象在划分
10	此处有 30 字相似 或者已有对象离开簇的时候，重新计算簇的平均值，然后对对象进行重新分配。这个过程不断重复，直到各簇中对象不再变化为止。 基于密度的聚类算法根据数据的分布密度，合并密度超过某个阈值的 相邻区域为一个区域，可以在有噪音的数据中发现各种形状和各种大小的簇，常用于对空间数据进行聚类。DBSCAN算法[19]就	网格聚类算法的研究 程伟想 -《华北电力大学（河北）硕士论文》- 2008-12-18（是否引证：否） 1.定数目的点。这样的方法可以用来过虑“噪音”孤立点数据，发现任意形状的簇。 DBSCAN 是一个有代表性的基于密度的方法，它根据一个密度阈值来控制簇的增长。OPTICS 是另一个基于密度的方法，它为自动的和交互的聚类分析提供一个聚类顺序。 (4)基于网 1_王震_一种基于密度最大值聚类算法的研究与应用 王震 -《大学生论文联合比对库》- 2017-05-11（是否引证：否） 1.将继续扩展围绕所找到的核心点周围的点，并且最终可以在没有集群中心的情况下识别集群。1.3本文的研究内容本文通过对基于密度的聚类算法的原理分析，采纳其中的合理成分，在此基础上研究快速密度峰搜索算法（CSFFDP）原理并提出一种考虑邻居交集和密度差的基于密度最大值的聚类算法（ 00343186681013071_王震_一种基于密度最大值聚类算法的研究与应用 郑浩 -《大学生论文联合比对库》- 2017-05-12（是否引证：否） 1.将继续扩展围绕所找到的核心点周围的点，并且最终可以在没有集群中心的情况下识别集群。1.3本文的研究内容本文通过对基于密度的聚类算法的原理分析，采纳其中的合理成分，在此基础上研究快速密度峰搜索算

		法（CSFFDP）原理并提出一种考虑邻居交集和密度差的基于密度最大值的聚类算法（
11	此处有 40 字相似 域包含样本集D中与xj的距离不大于的子样本集，即,这个样本集的个数记为 2）核心对象：对于任一样本xj∈D，如果其邻域对应的至少包含MinPts个样本，即如果，则xj是核心对象。 3）密度直达 ：如果且,则称p由q密度直达。当p和q都是核心对象时，密度直达满足对称性。 图2.1 图2.1中，假设MinPts=	基于深度学习的车辆检测和车牌定位 封晶 -《江西理工大学硕士论文》- 2017-05-24（是否引证：否） 1.iijNxxxdist Dx，；（2.13）核心对象：若jx 的? -邻域至少包含 Min Pts 个样本，即 (7) (8)Min Ptsx Nj??，则 jx 是一个核心对象；密度直达：若jx 位于ix 的？-邻域中，且ix 作为核心成员，那么定义jx 由ix 密度直达。密度可达：对ix 与j
12	此处有 225 字相似 b) :p,q密度相连 7）噪声点：设C1,C2,…,Ck是样本集D中的簇，则定义噪声点为不属于任何簇的数据集，即 算法流程： 输入：——半径 MinPts——给定点在邻域内成为核心对象的最小邻域点数。 D——集合。 输出：目标类簇集合 方法： Repeat 1）判断输入点是否为核心对象 2）找出核心对象的E邻域中的所有直接密度可达点。 Until 所有输入点都判断完毕 Repeat 针对所有核心对象的E邻域内所有直接密度可达点找到最大密度相连对象集合，中间涉及到一些密度可达对象的合并。 Until 所有核心对象的E领域都遍历完毕	基于密度的聚类方法研究 陈雪儿 -《大学生论文联合比对库》- 2014-05-23（是否引证：否） 1.点, DBSCAN 将访问数据库中的下一个点。（6）继续这一过程, 直到数据库中的所有点都被处理。算法描述：算法： DBSCAN输入： E——半径MinPts——给定点在E邻域内成为核心对象的最小邻域点数。D——集合。输出： 目标类簇集合方法： Repeat1） 判断输入点是否为核心对象2） 找出核心对象的E邻域中的所有直接密度可达点。Until 所有输入点都判断完毕 Repeat针对所有核心对象的E邻域内所有直接密度可达点找到最大密度相连对象集合，中间涉及到一些密度可达对象的合并。Until 所有核心对象的E领域都遍历完毕 算法流程如下：图 2.3 算法流程图根据DBSCAN算法列出对算法的实现目标a) 可以直接调整半径和最小阈值 基于密度的聚类方法研究 陈雪儿 -《大学生论文联合比对库》- 2014-05-27（是否引证：否） 1.点, DBSCAN 将访问数据库中的下一个点。（6）继续这一过程, 直到数据库中的所有点都被处理。算法描述：算法： DBSCAN输入： E——半径MinPts——给定点在E邻域内成为核心对象的最小邻域点数。D——集合。输出： 目标类簇集合方法： Repeat1） 判断输入点是否为核心对象2） 找出核心对象的E邻域中的所有直接密度可达点。Until 所有输入点都判断完毕 Repeat针对所有核心对象的E邻域内所有直接密度可达点找到最大密度相连对象集合，中间涉及到一些密度可达对象的合并。Until 所有核心对象的E领域都遍历完毕 算法流程如下：图 2.3 算法流程图根据DBSCAN算法列出对算法的实现目标a) 可以直接调整半径和最小阈值 基于DBScan算法的网页聚类分析 沈逸帆 -《大学生论文联合比对库》- 2017-05-21（是否引证：否） 1.集中不包含在任何集群中的数据点形成异常点。如图（5）。图（5）DBScan算法运行示意图而它的伪代码大概是：输入： E——半径MinPts——给定点在E邻域

	伪码描述： DBSCAN(D, eps, MinPts) { C = 0 for each point P in	内成为核心对象的最小邻域点数。D——集合。输出：目标类簇集合方法： Repeat1）判断输入点是否为核心对象2）找出核心对象的E邻域中的所有直接密度可达点。Until 所有输入点都判断完毕Repeat针对所有核心对象的E邻域内所有直接密度可达点找到最大密度相连对象集合，中间涉及到一些密度可达对象的合并。Until 所有核心对象的E领域都遍历完毕而其流程图简单可以表示为如图（6）：图（6）DBScan算法流程示意图3.3 DBScan算法优势DBSca 色谱指纹图谱的智能聚类分析在中医湿证辨别方面的研究 胡琳 -《东华大学硕士论文》- 2003-12-01（是否引证：否） 1.包含多于MinPts个点，则创建一个以p作为核心对象的新簇。然后，DBSCAN反复地寻找从这些核心对象直接密度可达的对象，这个过程可能涉及一些密度可达簇的合并。当没有新的点可以被添加到任何簇时，该过程结束。 如果采用空间索 基于划分的聚类算法研究 郑柏杰 -《重庆大学硕士论文》- 2005-10-01（是否引证：否） 1.多于 MinPts 个其他数据对象，则创建一个以 P作为核心对象的新簇。然后，反复地寻找从这些核心对象直接密度可达的对象。这个过程可能涉及一些密度可达簇的合并。当没有新的点可以被添加到任何簇时，该过程结束。这样算法得到的聚簇是是基于密度可达性的最大的密度 数据挖掘中基于遗传算法的聚类方法应用研究 吴多比 -《重庆大学硕士论文》- 2009-04-01（是否引证：否） 1.ε-邻域包含多于 MinPts 个其他数据对象，则创建一个以 P 作为核心对象的新簇。然后，反复地寻找从这些核心对象直接密度可达的对象。这个过程可能涉及一些密度可达簇的合并。当没有新的点可以被添加到任何簇时，该过程结束。这样算法得到的簇是是基于密度可达性的最大的密度相连对象的集合，

13	此处有 711 字相似
	最大密度相连对象集合，中间涉及到一些密度可达对象的合并。
	
	Until 所有核心对象的E领域都遍历完毕
	
	伪码描述：
	DBSCAN(D, eps, MinPts) {
	
	C = 0
	
	for each point P in dataset D {
	
	if P is visited
	
	continue next point
	
	mark P as visited
		DBSCAN聚类算法原理 - xuanyuansen的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》-（是否引证：否）
		
		1.同的是DBSCAND算法中有样本与样本直间分为直接（密度）可达与（密度）可达两种情况，定义分别如下：算法过程 伪代码 DBSCAN(D,eps,MinPts) { C = 0 for each point P in dataset D { if P is visited continue next point mark P as visited NeighborPts = regionQuery(P,eps) if sizeof(NeighborPts) < MinPts mark P as NOISE else { C = next cluster expandCluster(P,NeighborPts,C,eps,MinPts) } } } expandCluster(P,NeighborPts,C,eps,MinPts) { add P to cluster C for each point P' in NeighborPts { if P' is not visited { mark P' as visited NeighborPts' = regionQuery(P',eps) if sizeof(NeighborPts') >= MinPts NeighborPts = NeighborPts joined with NeighborPts' } if P' is not yet member of any cluster add P' to cluster C } } regionQuery(P,eps) return all points within P's eps-neighborhood (including P)

```
NeighborPts = regionQuery(P, eps)

if sizeof(NeighborPts) < MinPts

mark P as NOISE

else {

C = next cluster

expandCluster(P, NeighborPts, C, eps, MinPts)

}

}

}

expandCluster(P, NeighborPts, C, eps, MinPts) {

add P to cluster C

for each point P' in NeighborPts {

if P' is not visited {

mark P' as visited

NeighborPts' = regionQuery(P', eps)

if sizeof(NeighborPts') >= MinPts

NeighborPts = NeighborPts joined with NeighborPts'

}

if P' is not yet member of any cluster

add P' to cluster C

}

}

regionQuery(P, eps)

return all points within P's eps-neighborhood (including P)
```

图2.2

在图2.2中，MinPts=5，红色点为核心对象。所有核心

1.流程图对应的伪代码如算法3-1所示。算法3-1输入：数据对象集合D，半径Eps，密度阈值MinPts输出：聚类CDBSCAN(D, eps, MinPts) {C = 0for each point P in dataset D {if P is visitedcontinue next pointmark P as visitedNeighborPts = regionQuery(P, eps)if sizeof(NeighborPts) < MinPtsmark P as NOISEelse {C = next clusterexpandCluster(P, NeighborPts, C, eps, MinPts)}}}expandCluster(P, NeighborPts, C, eps, MinPts) {add P to cluster Cfor each point P' in NeighborPts {if P' is not visited {mark P' as visitedNeighborPts' = regionQuery(P', eps)if sizeof(NeighborPts') >= MinPtsNeighborPts = NeighborPts joined with NeighborPts'}if P' is not yet member of any clusteradd P' to cluster C}}regionQuery(P, eps){return all points within P's eps-neighborhood (including P)}3.4 基于局部异常因子的异常检测通过dbscan全局密度阈值的过滤，历史记录中的极其异常的消费点会被筛选出来，一些密度较低

	对象密度直达的样本点在以核心对象为圆心的圆	

指　标

疑似剽窃文字表述

---

1. 信息泄露。如何在不泄露用户隐私的前提下，提高大数据的利用率，是目前大数据研究领域的关键问题之一。
2. 若要拷贝有数字水印的信号，所嵌入的信息也会一并被拷贝。数字水印可分为浮现式和隐藏式两种。一般来说，浮现
3. 隐藏式的水印是以数字数据的方式加入音频、图片或视频中，但在一般的状况下无法被看见。隐藏式水印的重要作用之一是保护版权，期望能借此避免或阻止数字媒体未经授权的复制和拷贝。
4. 聚类算法又可详细划分为基于图论的聚类、基于网格的聚类、基于平方差的聚类和基于密度的聚类。
   基于图论的聚类将聚类
5. 这种方法使用一种多分辨率的网络数据结构。它将对象空间量化成有限数目的单元，这些网格形成了网格结构，所有的聚类结构都在该结构上进行。这种方法的主要优点是处理速度快，其处理时间独立于数据对象数，仅依赖于量化空间中的每一维的单元数。
6. 每个对象落入一个网格单元，网格单元对应的属性空间包含该对象的值。
   基于平方差的聚类
7. 对于其他对象，计算其与各个簇中心的距离，将它们划入距离最近的簇。采用迭代的重定位技术，尝试通过对象在划分间移动来改进划分。所谓重定位技术，就是当有新的对象加入簇或者已有对象离开簇的时候，重新计算簇的平均值，然后对对象进行重新分配。这个过程不断重复，直到各簇中对象不再变化为止。
   基于密度的聚类算法根据数据的分布密度，合并密度超过某个阈值的
8. 算法流程：
   输入：——半径
   MinPts——给定点在邻域内成为核心对象的最小邻域点数。
   D——集合。
   输出：目标类簇集合
   方法： Repeat
   1）判断输入点是否为核心对象
   2）找出核心对象的E邻域中的所有直接密度可达点。
   Until 所有输入点都判断完毕
   Repeat
   针对所有核心对象的E邻域内所有直接密度可达点找到最大密度相连对象集合，中间涉及到一些密度可达对象的合并。

4. 第三章数据获取及预处理

总字数：4500

相似文献列表　文字复制比：46.6%(2097)　疑似剽窃观点：(0)

1	基于Web的校园办公自动化系统设计与实现 靖雯(导师：陆鑫;卢长军) -《电子科技大学硕士论文》- 2012-03-01	39.6%（1783） 是否引证：否
2	[gb18030] 10055101徐浩程 毕设论文 徐浩程 -《大学生论文联合比对库》- 2014-06-13	38.5%（1731） 是否引证：否
3	[gb18030] 10055101徐浩程 毕设论文 徐浩程 -《大学生论文联合比对库》- 2014-06-13	38.5%（1731） 是否引证：否
4	HMAC-MD5的分析与实现 康四雯 -《大学生论文联合比对库》- 2015-06-01	38.5%（1731） 是否引证：否
5	基于Android端的新闻阅读器的设计与实现 张雨杰 -《大学生论文联合比对库》- 2014-05-02	38.4%（1730） 是否引证：否
6	韩笑_大学生在线求职招聘网站的设计与实施 20142174 -《大学生论文联合比对库》- 2014-10-28	36.6%（1649） 是否引证：否
7	基于WEB的人才招聘系统的设计与实现 王璐瑶(导师：王连平) -《吉林大学硕士论文》- 2013-06-01	36.6%（1649） 是否引证：否
8	现在才知道其实中国对美国军方最具威胁的不是核武也不是打卫星技术.原来是中国的密码破译能力.中国数字家已将西方认为不可能破译的MD5和SHA-1成功破译_标点 - 韶韶 - 记者版_媒体 -《网络（http://www.xici.net/）》- 2013	36.6%（1649） 是否引证：否
9	网络121_2012122631_聂睿 聂睿 -《大学生论文联合比对库》- 2016-06-08	36.6%（1648） 是否引证：否
10	Message-Digest Algorithm 5 - 走马观花 - CSDN博客	36.3%（1632）

	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
11	超市后台管理系统设计与实现	35.3%（1590）
	吴波(导师：张平健;潘勇) - 《华南理工大学硕士论文》- 2010-11-10	是否引证：否
12	基于SHA-256算法的嵌入式软件保护技术研究	34.3%（1543）
	郑佳敏(导师：沈建华) - 《华东师范大学硕士论文》- 2014-03-22	是否引证：否
13	MD5 - yjg428的专栏 - 博客频道 - CSDN.NET	32.7%（1471）
	- 《网络（http://blog.csdn.net）》- 2013	是否引证：否
14	简要分析用MD5加密算法加密信息（精编版） - Auspicious Cloud's View - 博客频道 - CSDN.NET	32.4%（1456）
	- 《网络（http://blog.csdn.net）》- 2013	是否引证：否
15	简要分析用MD5加密算法加密信息（精编版） - Auspicious Cloud's View - CSDN博客	32.4%（1456）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
16	MD5_百度百科	32.2%（1451）
	- 《网络（http://baike.baidu.c）》- 2010	是否引证：否
17	MD5算法Qt实现 - tandesir的专栏 - CSDN博客	32.2%（1451）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
18	什么是MD5 - 豆丁网	32.2%（1451）
	- 《互联网文档资源（http://www.docin.com）》- 2017	是否引证：否
19	常数ti是4294967296*abs_caoaugt	31.4%（1415）
	- 《网络（http://blog.sina.com）》- 2014	是否引证：否
20	md5算法及其C语言的实现 - 网络与安全,mayu8758,拥抱开源，把握未来！	31.4%（1411）
	- 《网络（http://blog.opendige）》- 2011	是否引证：否
21	电子印章的设计与实现	31.0%（1393）
	龙宇 - 《大学生论文联合比对库》- 2015-05-27	是否引证：否
22	MD5加密算法详细分析	4.0%（181）
	- 《网络（http://blog.csdn.net）》- 2017	是否引证：否
23	基于C/S与B/S混合模式下教务系统的研究和分析	3.9%（175）
	刘立(导师：王长波) - 《华东师范大学硕士论文》- 2011-04-01	是否引证：否
24	13刘佳文--2014年度结项报告_移动支付安全问题研究—以微信支付为例	3.9%（175）
	- 《大学生论文联合比对库》- 2015-10-20	是否引证：否
25	13刘佳文--2014年度结项报告_移动支付安全问题研究—以微信支付为例-新	3.9%（175）
	- 《大学生论文联合比对库》- 2015-10-21	是否引证：否
26	20123264从海云_温蜜_毕设论文	3.6%（160）
	从海云 - 《大学生论文联合比对库》- 2016-06-07	是否引证：否

	原文内容	相似内容来源
1	此处有 72 字相似 密码学家罗纳德·李维斯特于1992年公开了MD5[20]摘要加密算法，输入任意长度的信息，经过处理，输出128位的信息。 经过程序流程，生成四个32位数据，最后联合起来成为一个128-bits散列。基本方式为，求余，取余，调整长度，与链接变量进行循环运算，得出结果。 图3.1展示了其运算过程，一个MD5运算—由类似的64次循环构成，分成4组16次。F 一个非线性函数；一个函数运算一次。	MD5加密算法详细分析 - 《网络（http://blog.csdn.net）》-（是否引证：否） 1.表示一个 32-bits 常数，用来完成每次不同的计算。MD5是输入不定长度信息，输出固定长度128-bits的算法。经过程序流程，生成四个32位数据，最后联合起来成为一个128-bits散列。基本方式为，求余、取余、调整长度、与链接变量进行循环运算。得出结果。{\displaystyle F(X,Y,Z)=(X\wedge {Y})\vee (\neg {X}\wedge { Message-Digest Algorithm 5 - 走马观花 - CSDN博客 - 《网络（http://blog.csdn.net）》-（是否引证：否） 1.示一个 32-bits 常数，用来完成每次不同的计算。MD5是輸入不定長度信息，輸出固定長度128-bits的演算法。经过程序流程，生成四个32位数据，最后联合起来成为一个128-bits散列。基本方式为，求余、取余、调整长度、与链接变量进行循环运算。得出结果。 是XOR,AND,OR ,NOT 的符号。 伪代码 //Note: All variables are unsign

电子印章的设计与实现 龙宇 -《大学生论文联合比对库》- 2015-05-27（是否引证：否）

1.28位（16字节）的散列值，用于确保信息传输完整一致。MD5是输入不定长度信息，输出固定长度128-bits的算法。经过算法程序流程，生成四个32位数据，最后联合起来成为一个128-bits散列。基本方式为，求余、取余、调整长度、与链接变量进行循环运算，最后得出结果。一般128位的MD5散列被表示为32位十六进制数字。下面将给出几个MD5散列具体示例：（1）MD5("")=d41d

基于C/S与B/S混合模式下教务系统的研究和分析 刘立 -《华东师范大学硕士论文》- 2011-04-01（是否引证：否）

1.途下对比用户操作的合法性[27】。 MDS是输入不定长度信息，输出固定长度128一bits的算法。经过程序流程，生成四个32位数据，最后联合起来成为一个128一bits散列。基本方式为，求余、取余、调整长度、与链接变量进行循环运算。得出结果。f(x，y，z)=(x八y)V(一x八z)g(x，y，z)=(x八z)V(y八一z)h(x，y，z)=x

13刘佳文--2014年度结项报告_移动支付安全问题研究—以微信支付为例 -《大学生论文联合比对库》- 2015-10-20（是否引证：否）

1.产生碰撞的概率非常之小。以下是这个算法的实现过程：MD5是输入不定长度信息，输出固定长度128-bits的算法。经过程序流程，生成四个32位数据，最后联合起来成为一个128-bits散列。基本方式为：求余、取余、调整长度、与链接变量进行循环运算，得出结果。1、填充编码。在MD5算法中，首先需要对信息进行填充，使其位长对512求余的结果等于448。因此，信息的位长（Bit

13刘佳文--2014年度结项报告_移动支付安全问题研究—以微信支付为例-新 -《大学生论文联合比对库》- 2015-10-21（是否引证：否）

1.产生碰撞的概率非常之小。以下是这个算法的实现过程：MD5是输入不定长度信息，输出固定长度128-bits的算法。经过程序流程，生成四个32位数据，最后联合起来成为一个128-bits散列。基本方式为：求余、取余、调整长度、与链接变量进行循环运算，得出结果。1、填充编码。在MD5算法中，首先需要对信息进行填充。因此，信息的位长（Bits Length）将被扩展至N*512

现在才知道其实中国对美国军方最具威胁的不是核武也不是打卫星技术.原来是中国的密码破译能力.中国数字家已将西方认为不可能破译的MD5和SHA-1成功破译_标点 - 韶韶 - 记者版_媒体 -《网络（http://www.xici.net/）》- （是否引证：否）

1.i 表示一个 32-bit 常数,用来完成每次不同的计算. MD5算法以16个32位子分组即512位分组来提供数据杂凑，经过程序流程，生成四个32位数据，最后联合起来成为一个128位散列。基本方式为，求余、取余、调整长度、与链接变量进行循环运算。得出结果。 是 XOR,AND,OR ,NOT 的符号。 伪代码 //Note: All variables are unsign

2	此处有 125 字相似 合起来成为一个128-bits散列。基本方式为，求余，取余，调整长度，与链接变量进行循环运算，得出结果。图3.1展示了其 运算过程，一个MD5运算—由类似的64次循环构成，分成4组16次。F 一个非线性函数；一个函数运算一次。Mi 表示一个 32-bits 的输入数据，Ki 表示一个 32-bits 常数，用来完成每次不同的计算。 图3.1 MD5运算流程 摘要算法 流程如下： 1）填充：如果信息长度（以bit为单位）对512求余的结果不等于448，则对信息进行填充，在信息尾部添加一	现在才知道其实中国对美国军方最具威胁的不是核武也不是打卫星技术.原来是中国的密码破译能力.中国数字家已将西方认为不可能破译的MD5和SHA-1成功破译_标点 - 韶韶 - 记者版_媒体 -《网络（http://www.xici.net/）》-（是否引证：否） 1.楼 ☆金牌小卧底★ 的话：如果是这样，那哈希算法 …… ...哈希算法不就是MD5! 算法 Figure 1. 一个MD5运算— 由类似的64次循环构成,分成4组16次. F 一个非线性函数; 一个函数运算一次. Mi 表示一个 32-bit 字节的输入数据,Ki 表示一个 32-bit 常数,用来完成每次不同的计算. MD5算法以16个32位子分组即512位分组来提供数据杂凑，... 还真像那么回事，只是看的头晕k MD5加密算法详细分析 -《网络（http://blog.csdn.net）》-（是否引证：否） 1. 1321 中被加以规范。 将数据（如一段文字）运算变为另一固定长度，是散列算法的基础原理。 算法 Figure 1. 一个MD5运算— 由类的64次循环构成，分成4组16次。F 一个非线性函数；一个函数运算一次。Mi 表示一个 32-bits 的输入数据，Ki 表示一个 32-bits 常数，用来完成每次不同的计算。 MD5是输入不定长度信息，输出固定长度128-bits的算法。经过程序流程，生成四个32位数据，最后联合起来成为一个128-b Message-Digest Algorithm 5 - 走马观花 - CSDN博客 -《网络（http://blog.csdn.net）》-（是否引证：否） 1.法计算下载文件的MD5校验和，然后通过检查这两个校验和是否一致，就能判断下载的文件是否出错。 算法 Figure 1. 一个MD5运算— 由类的64次循环构成，分成4组16次。F 一个非线性函数；一个函数运算一次。Mi 表示一个 32-bits 的输入数据，Ki 表示一个 32-bits 常数，用来完成每次不同的计算。 MD5是輸入不定長度信息，輸出固定長度128-bits的演算法。经过程序流程，生成四个32位数据，最后联合起来成为一个128- 基于C/S与B/S混合模式下教务系统的研究和分析 刘立 -《华东师范大学硕士论文》- 2011-04-01（是否引证：否） 1.)=y。(xV一z)。符号代表XOR;八符号代表AND;V符号代表OR，一代表NOT。 一个MDS运算由类似的64次循环构成，分成4组16次。F一个非线性函数;一个函数运算一次。麟表示一个32一bits的输入数据，龙表示一个32一bits常数，用来完成每次不同的计算。如图5一7:基于C/S与B/S混合模式卜教务系统的研究和分析K IAA A 13刘佳文--2014年度结项报告_移动支付安全问题研究—以微信支付为例 -《大学生论文联合比对库》- 2015-10-20（是否引证：否） 1.=(N+1）*512，即长度恰好是512的整数倍。这样做的原因是为满足后面处理中对信息长度的要求。如右图，一个MD5运算由类似的64次循环构成，分成4组16次。F 一个非线性函数；一个函数运算一次。Mi 表示一个32-bits 的输入数据，Ki表示一个 32-bits 常数，用来完成每次不同的计算。主循环有四轮（MD4只有三轮），每轮循环都很相似。第一轮进行16次操作。每次操作对a、b、c和d中的其中三个作一次非线性 13刘佳文--2014年度结项报告_移动支付安全问题研究—以

		微信支付为例-新 -《大学生论文联合比对库》- 2015-10-21（是否引证：否）
		1.）*512，即长度恰好是512的整数倍。这样做是必须的，MD5需要满足后面处理中对信息长度的要求。如右图，一个MD5运算由类似的64次循环构成，分成4组16次。F 一个非线性函数；一个函数运算一次。Mi 表示一个 32-bits 的输入数据，Ki表示一个 32-bits 常数，用来完成每次不同的计算。主循环有四轮（MD4只有三轮），每轮循环都很相似。第一轮进行16次操作。每次操作对a、b、c和d中的其中三个作一次非线性
3	此处有 109 字相似 法流程如下： 1）填充：如果信息长度（以bit为单位）对512求余的结果不等于448，则对信息进行填充，在信息尾部添加一个1和若干个0，使得填充后信息长度为N*512+448位。 2）记录信息长度：用64位来存储填充前的信息长度。这64位追加在第一步所得结果后面，这样信息长度就变成N*512+448+64=(N+1)*512位。 3）初始化MD数组：A=0X67452301L，B=0XEFCDAB89L，C=0X98BADCFEL，D=0X1032	20123264从海云_温蜜_毕设论文 从海云 -《大学生论文联合比对库》- 2016-06-07（是否引证：否）
		1.—填充：如果输入信息的长度对512，求余的结果不等于448，就需要填充使得对512求余的结果等于448。填充的方法为填充一个1和n个0。填充完后，信息的长度就为N*512+448(bit)；第二步——记录信息长度：用64位来存储填充前信息长度。这64位加在第一步结果的后面，这样信息长度就变成了N*512+448+64=(N+1)*512位。第三步——装入标准的幻数（四个整数）：标准的幻数（物理顺序）是（A=(01234567)16，B=(89ABCDEF)1
4	此处有 55 字相似 64位追加在第一步所得结果后面，这样信息长度就变成N*512+448+64=(N+1)*512位。 3）初始化MD数组：A=0X67452301L，B=0XEFCDAB89L，C=0X98BADCFEL，D=0X10325476 L 4）进行循环运算 伪码描述： //Note: All variables are unsigned 32 bi	基于Web的校园办公自动化系统设计与实现 靖雯 -《电子科技大学硕士论文》- 2012-03-01（是否引证：否）
		1.的要求。MD5 中有四个 32 位被称作链接变量（Chaining Variable）的整数参数，他们分别为：A=0x67452301，B=0xefcdab89，C=0x98badcfe，D=0x10325476。35当设置好这四个链接变量后，就开始进入算法的四轮循环运算。循环的次数是信息中 512 位信息分组的数目。
		基于SHA-256算法的嵌入式软件保护技术研究 郑佳敏 -《华东师范大学硕士论文》- 2014-03-22（是否引证：否）
		1.分组。算法的输出由四个32比特的分组组成，它们级联形成一个128比特的Hash值。4个32比特向量的初始值为：A=0x67452301, B=0xefcdab89, C=0x98badcfe, D=0xl0325476它们称为链接变量（ChainingVairable)。接着进行算法的主循环，循环的次数是消息中512比特消息分组的
		MD5 - yjg428的专栏 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》-（是否引证：否）
		1.处理中对信息长度的要求。 MD5中有四个32位被称作链接变量（Chaining Variable）的整数参数，他们分别为：A=0x67452301，B=0xefcdab89，C=0x98badcfe，D=0x10325476。 当设置好这四个链接变量后，就开始进入算法的四轮循环运算。循环的次数是信息中512位信

		息分组的数目。 将上面四个链接变量 20123264从海云_温蜜_毕设论文 从海云 -《大学生论文联合比对库》- 2016-06-07（是否引证：否） 1.6，B=(89ABCDEF)16，C=(FEDCBA98)16，D=(76543210)16）。如果在程序中定义应该是（A=0X67452301L，B=0XEFCDAB89L，C=0X98BADCFEL，D=0X10325476L）。第四步——装四轮循环运算：循环的次数是分组的个数（N+1）1）将每一512字节细分成16个小组，每个小组64位（8个

5	此处有 1736 字相似
	301L，B=0XEFCDAB89L，C=0X98BADCFEL，D=0X10325476L
	
	4）进行循环运算
	
	伪码
	描述：
	
	//Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating
	
	var int[64] r, k
	
	//r specifies the per-round shift amounts
	
	r[0..15] : = {7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22}
	
	r[16..31] : = {5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20}
	
	r[32..47] : = {4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23}
	
	r[48..63] : = {6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21}
	
	//Use binary integer part of the sines of integers as constants:
	
	for i from 0 to 63
	
	k[i] := floor(abs(sin(i + 1)) × 2^32)
	
	//Initialize variables:
	
	var int h0 := 0x67452301
	
	var int h1 := 0xEFCDAB89
	
	var int h2 := 0x98BADCFE
		基于Android端的新闻阅读器的设计与实现 张雨杰 -《大学生论文联合比对库》- 2014-05-02（是否引证：否）
		1.加上a、b、c、d。然后用下一分组数据继续运行算法，最后的输出是A、B、C和D的级联。2.2.3.1MD5伪代码/Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculatingvar int[64] r, k//r specifies the per-round shift amountsr[0..15] : = {7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22}r[16..31] : = {5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20}r[32..47] : = {4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23}r[48..63] : = {6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21}//Use binary integer part of the sines of integers as constants:for i from 0 to 63k[i] := floor(abs(sin(i + 1)) × 2^32)//Initialize variables:var int h0 := 0x67452301var int h1 := 0xEFCDAB89var int h2 := 0x98BADCFEvar int h3 := 0x10325476//Pre-processing:append "1" bit to messageappend "0" bits until message length in bits ≡ 448 (mod 512)append bit length of message as 64-bit little-endian integer to message//Process the message in successive 512-bit chunks:for each 512-bit chunk of messagebreak chunk into sixteen 32-bit little-endian words w[i], 0 ≤ i ≤ 15//Initialize hash value for this chunk:var int a := h0var int b := h1var int c := h2var int d := h3//Main loop:for i from 0 to 63if 0 ≤ i ≤ 15 thenf := (b and c) or ((not b) and d)g := ielse if 16 ≤ i ≤ 31f := (d and b) or ((not d) and c)g := (5×i + 1) mod 16else if 32 ≤ i ≤ 47f := b xor c xor dg := (3×i + 5) mod 16else if 48 ≤ i ≤ 63f := c xor (b or (not d))g := (7×i) mod 16temp := dd := cc := bb := leftrotate((a + f + k[i] + w[g]),r[i]) + ba := tempNext i//Add this chunk's hash to result so far:h0 := h0 + ah1 := h1 + bh2 := h2 + ch3 := h3 + dEnd ForEachvar int digest := h0 append h1 append h2 append h3 //(expressed as little-endian)2.3XML解析XML（eXtensible Markup Language）是数据表示的一个开放标准，是跨平台的，依赖
		
		[gb18030] 10055101徐浩程 毕设论文 徐浩程 -《大学生论文联合比对库》- 2014-06-13（是否引证：否）
		1.-bits的算法[12]。伪代码如下：== 伪代码 ==<source lang="pascal">//Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculatingvar int[64] r, k//r specifies the per-round shift amountsr[0..15] : = {7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17,

var int h3 := 0x10325476

//Pre-processing:

append "1" bit to message

append "0" bits until message length in bits ≡ 448 (mod 512)

append bit length of message as 64-bit little-endian integer to message

//Process the message in successive 512-bit chunks:

for each 512-bit chunk of message

break chunk into sixteen 32-bit little-endian words w[i], 0 ≤ i ≤ 15

//Initialize hash value for this chunk:

var int a := h0

var int b := h1

var int c := h2

var int d := h3

//Main loop:

for i from 0 to 63

if 0 ≤ i ≤ 15 then

f := (b and c) or ((not b) and d)

g := i

else if 16 ≤ i ≤ 31

f := (d and b) or ((not d) and c)

g := (5×i + 1) mod 16

else if 32 ≤ i ≤ 47

f := b xor c xor d

g := (3×i + 5) mod 16

else if 48 ≤ i ≤ 63

22}r[16..31] : = {5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20}r[32..47] : = {4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23}r[48..63] : = {6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21}//Use binary integer part of the sines of integers as constants:for i from 0 to 63k[i] := floor(abs(sin(i + 1)) × 2^32)//Initialize variables:var int h0 := 0x67452301var int h1 := 0xEFCDAB89var int h2 := 0x98BADCFEvar int h3 := 0x10325476//Pre-processing:append "1" bit to messageappend "0" bits until message length in bits ≡ 448 (mod 512)append bit length of message as 64-bit little-endian integer to message//Process the message in successive 512-bit chunks:for each 512-bit chunk of messagebreak chunk into sixteen 32-bit little-endian words w[i], 0 ≤ i ≤ 15//Initialize hash value for this chunk:var int a := h0var int b := h1var int c := h2var int d := h3//Main loop:for i from 0 to 63if 0 ≤ i ≤ 15 thenf := (b and c) or ((not b) and d)g := ielse if 16 ≤ i ≤ 31f := (d and b) or ((not d) and c)g := (5×i + 1) mod 16else if 32 ≤ i ≤ 47f := b xor c xor dg := (3×i + 5) mod 16else if 48 ≤ i ≤ 63f := c xor (b or (not d))g := (7×i) mod 16temp := dd := cc := bb := leftrotate((a + f + k[i] + w[g]),r[i]) + ba := tempNext i//Add this chunk's hash to result so far:h0 := h0 + ah1 := h1 + bh2 := h2 + ch3 := h3 + dEnd ForEachvar int digest := h0 append h1 append h2 append h3 //(expressed as little-endian)</source>在python的hashlib库中内置了MD5算法的实现（如图5-3所示）：图5-3 M

[gb18030] 10055101徐浩程 毕设论文 徐浩程 -《大学生论文联合比对库》- 2014-06-13（是否引证：否）

1.-bits的算法[12]。伪代码如下：== 伪代码 ==<source lang="pascal">//Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculatingvar int[64] r, k//r specifies the per-round shift amountsr[0..15] : = {7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22}r[16..31] : = {5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20}r[32..47] : = {4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23}r[48..63] : = {6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21}//Use binary integer part of the sines of integers as constants:for i from 0 to 63k[i] := floor(abs(sin(i + 1)) × 2^32)//Initialize variables:var int h0 := 0x67452301var int h1 := 0xEFCDAB89var int h2 := 0x98BADCFEvar int h3 := 0x10325476//Pre-processing:append "1" bit to messageappend "0" bits until message length in bits ≡ 448 (mod 512)append bit length of message as 64-bit little-endian integer to message//Process the message in successive 512-bit chunks:for each 512-bit chunk of messagebreak chunk into sixteen 32-bit little-endian words w[i], 0 ≤ i ≤ 15//Initialize hash value for this chunk:var int a := h0var int b := h1var int c := h2var int d := h3//Main loop:for i from 0 to 63if 0 ≤ i ≤ 15 thenf :=

f := c xor (b or (not d))

g := (7×i) mod 16

temp := d

d := c

c := b

b := leftrotate((a + f + k[i] + w[g]),r[i]) + b

a := temp

Next i

//Add this chunk's hash to result so far:

h0 := h0 + a

h1 := h1 + b

h2 := h2 + c

h3 := h3 + d

End ForEach

var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian
)

为充分保护用户隐私，本文在进行数据导出时对车牌号码这一关键字段采用MD5算法进行映射。Oracle提供了DBMS_

(b and c) or ((not b) and d)g := ielse if 16 ≤ i ≤ 31f := (d and b) or ((not d) and c)g := (5×i + 1) mod 16else if 32 ≤ i ≤ 47f := b xor c xor dg := (3×i + 5) mod 16else if 48 ≤ i ≤ 63f := c xor (b or (not d))g := (7×i) mod 16temp := dd := cc := bb := leftrotate((a + f + k[i] + w[g]),r[i]) + ba := tempNext i//Add this chunk's hash to result so far:h0 := h0 + ah1 := h1 + bh2 := h2 + ch3 := h3 + dEnd ForEachvar int digest := h0 append h1 append h2 append h3 //(expressed as little-endian)</source>在python的hashlib库中内置了MD5算法的实现（如图5-3所示）：图5-3 M

基于Web的校园办公自动化系统设计与实现 靖雯 -《电子科技大学硕士论文》- 2012-03-01（是否引证：否）

1.qrstuvwxyz")=
f29939a25efabaef3b87e2cbfe641315伪代码实现
//Note: All variables are unsigned 32 bits and wrap modulo 2^32 whencalculatingvar int[64] r, k //r specifies the per-round shift amountsr[0..15] : = {7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22}r[16..31] : = {5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20}r[32..47] : = {4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23}r[48..63] : = {6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21}//Use binary integer part of the sines of integers as constants:for i from 0 to 63k[i] := floor(abs(sin(i + 1)) × 2^32)//Initialize variables:var int h0 := 0x67452301var int h1 := 0xEFCDAB89var int h2 := 0x98BADCFEvar int h3 := 0x10325476//Pre-processing:append "1" bit to messageappend "0" bits until message length in bits ≡ 448 (mod 512)append bit length of message as 64-bit little-endian integer to message//Process the message in successive 512-bit chunks:for each 512-bit chunk of messagebreak chunk into sixteen 32-bit little-endian words w[i], 0 ≤ i ≤ 15//Initialize hash value for this chunk:var int a := h039var int b := h1var int c := h2var int d := h3//Main loop:for i from 0 to 63if 0 ≤ i ≤ 15 thenf := (b and c) or ((not b) and d)g := ielse if 16 ≤ i ≤ 31f := (d and b) or ((not d) and c)g := (5×i + 1) mod 16else if 32 ≤ i ≤ 47f := b xor c xor dg := (3×i + 5) mod 16else if 48 ≤ i ≤ 63f := c xor (b or (not d))g := (7×i) mod 16temp := dd := cc := bb := ((a + f + k[i] + w[g]) leftrotate r[i]) + ba := temp//Add this chunk's hash to result so far:h0 := h0 + ah1 := h1 + bh2 := h2 + ch3 := h3 + dvar int digest := h0 append h1 append h2 append h3//(expressed as little-endian)404.6.2 Sha1 算法安全哈希算法（Secure Hash Algorithm）主要适用于数字签名标准（Digit

HMAC-MD5的分析与实现 康四雯 -《大学生论文联合比对库》- 2015-06-01（是否引证：否）

1.了。最重要的是，这样可以随时替换掉安全性低的Hash算法，保证了HMAC的安全性。3.3 实现MD5的伪代码//Note: All variables are unsigned 32 bits and wrap

modulo 2^32 when calculatingvar int[64] r, k//r specifies the per-round shift amountsr[0..15] := {7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22} r[16..31] := {5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20}r[32..47] := {4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23}r[48..63] := {6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21}//Use binary integer part of the sines of integers as constants:for i from 0 to 63 k[i] := floor(abs(sin(i + 1)) × 2^32)//Initialize variables:var int A:= 0x67452301var int B := 0xEFCDAB89var int C := 0x98BADCFEvar int D:= 0x10325476//Pre-processing:append "1" bit to messageappend "0" bits until message length in bits ≡ 448 (mod 512)append bit length of message as 64-bit little-endian integer to message//Process the message in successive 512-bit chunks:for each 512-bit chunk of message break chunk into sixteen 32-bit little-endian words w[i], 0 ≤ i ≤ 15 //Initialize hash value for this chunk: var int a := A var int b := B var int c := C var int d := D //Main loop: for i from 0 to 63 if 0 ≤ i ≤ 15 then f := (b and c) or ((not b) and d) g := i else if 16 ≤ i ≤ 31 f := (d and b) or ((not d) and c) g := (5×i + 1) mod 16 else if 32 ≤ i ≤ 47 f := b xor c xor d g := (3×i + 5) mod 16 else if 48 ≤ i ≤ 63 f := c xor (b or (not d)) g := (7×i) mod 16 temp := d d := c c := b b := leftrotate((a + f + k[i] + w[g]),r[i]) + b a := temp Next i //Add this chunk's hash to result so far: A:= A + a B:= B + b C:= C + c D:= D + dEnd ForEach 512-bit chunksvar int digest := h0 append h1 append h2 append h3 //(expressed as little-endian)HMAC的伪代码：function hmac (key, message)if (length(key)

韩笑_大学生在线求职招聘网站的设计与实施_20142174 - 《大学生论文联合比对库》- 2014-10-28（是否引证：否）

1.2位子分组，然后进行相关处理，最后形成四个32位分组输出，并将它们连接形成一个128位散列值。基本算法如下：var int[64] r, k//r specifies the per-round shift amounts r[0..15] : = {7, 12, 17, 22,7, 12, 17, 22,7, 12, 17, 22,7, 12, 17, 22}r[16..31] : = {5,9, 14, 20,5,9, 14, 20,5,9, 14, 20,5,9, 14, 20} r[32..47] : = {4, 11, 16, 23,4, 11, 16, 23,4, 11, 16, 23,4, 11, 16, 23} r[48..63] : = {6, 10, 15, 21,6, 10, 15, 21,6, 10, 15, 21,6, 10, 15, 21}//Use binary integer part of the sines of integers as constants: for i from 0 to 63k[i] := floor(abs(sin(i + 1)) × 2^32)//Initialize variables: var int h0 := 0x67452301 var int h1 := 0xEFCDAB89 var int h2 := 0x98BADCFE var int h3 := 0x10325476//Pre-processing: append "1" bit to message append "0" bits until message length in bits ≡ 448 (mod 512) append bit length of message as 64-bit little-endian integer to message//Process the message in successive 512-bit chunks: for each 512-bit chunk of messagebreak chunk into sixteen 32-bit little-endian words w[i], 0 ≤ i ≤

15//Initialize hash value for this chunk:var int a := h0var int b := h1var int c := h2var int d := h3//Main loop:for i from 0 to 63if 0 ≤ i ≤ 15 thenf := (b and c) or ((not b) and d)g := ielse if 16 ≤ i ≤ 31f := (d and b) or ((not d) and c)g := (5×i + 1) mod 16else if 32 ≤ i ≤ 47f := b xor c xor dg := (3×i + 5) mod 16else if 48 ≤ i ≤ 63f := c xor (b or (not d))g := (7×i) mod 16temp := dd := cc := bb := leftrotate((a + f + k[i] + w[g]),r[i]) + ba := tempNext i//Add this chunk's hash to result so far:h0 := h0 + ah1 := h1 + bh2 := h2 + ch3 := h3 + d End ForEach var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian)2.2 Web Services介绍 Web Service是基于网络的、分布式的模块化组件，它执行特定的任务，遵守具体的

基于WEB的人才招聘系统的设计与实现 王璐瑶 -《吉林大学硕士论文》- 2013-06-01（是否引证：否）

1.进行相关处理，最后形成四个 32 位分组输出，并将它们连接形成一个 128 位散列值。基本算法如下：var int[64] r, k//r specifies the per-round shift amountsr[0..15] : = {7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22}r[16..31] : = {5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20}r[32..47] : = {4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23}r[48..63] : = {6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21}//Use binary integer part of the sines of integers as constants:for i from 0 to 63k[i] := floor(abs(sin(i + 1)) × 2^32)//Initialize variables:var int h0 := 0x67452301var int h1 := 0xEFCDAB89var int h2 := 0x98BADCFEvar int h3 := 0x10325476//Pre-processing:append "1" bit to messageappend "0" bits until message length in bits ≡ 448 (mod 512)append bit length of message as 64-bit little-endian integer to message//Process the message in successive 512-bit chunks:for each 512-bit chunk of messagebreak chunk into sixteen 32-bit little-endian words w[i], 0 ≤ i ≤ 1512//Initialize hash value for this chunk:var int a := h0var int b := h1var int c := h2var int d := h3//Main loop:for i from 0 to 63if 0 ≤ i ≤ 15 thenf := (b and c) or ((not b) and d)g := ielse if 16 ≤ i ≤ 31f := (d and b) or ((not d) and c)g := (5×i + 1) mod 16else if 32 ≤ i ≤ 47f := b xor c xor dg := (3×i + 5) mod 16else if 48 ≤ i ≤ 63f := c xor (b or (not d))g := (7×i) mod 16temp := dd := cc := bb := leftrotate((a + f + k[i] + w[g]),r[i]) + ba := tempNext i//Add this chunk's hash to result so far:h0 := h0 + ah1 := h1 + bh2 := h2 + ch3 := h3 + dEnd ForEachvar int digest := h0 append h1 append h2 append h3 //(expressed as little-endian)2.5 本章小结本章介绍了与本课题相关的软件开发技术及理论，包括：数据库设计的相关理论；SQL Server 数

网络121_2012122631_聂睿 聂睿 -《大学生论文联合比对库》- 2016-06-08（是否引证：否）

1.逐位运算的函数。即，如果X，那么Y，否则Z。函数H是逐位奇偶操作符。5.3.2 MD5的代码及伪代码表示

var int[64] r, k //r specifies the per-round shift amountsr[0..15] : = {7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22}r[16..31] : = {5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20}r[32..47] : = {4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23}r[48..63] : = {6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21}//Use binary integer part of the sines of integers as constants:for i from 0 to 63k[i] := floor(abs(sin(i + 1)) × 2^32)//Initialize variables:var int h0 := 0x67452301var int h1 := 0xEFCDAB89var int h2 := 0x98BADCFEvar int h3 := 0x10325476//Pre-processing:append "1" bit to messageappend "0" bits until message length in bits ≡ 448 (mod 512)append bit length of message as 64-bit little-endian integer to message//Process the message in successive 512-bit chunks:for each 512-bit chunk of messagebreak chunk into sixteen 32-bit little-endian words w[i], 0 ≤ i ≤ 15//Initialize hash value for this chunk:var int a := h0var int b := h1var int c := h2var int d := h3//Main loop:for i from 0 to 63if 0 ≤ i ≤ 15 thenf := (b and c) or ((not b) and d)g := ielse if 16 ≤ i ≤ 31f := (d and b) or ((not d) and c)g := (5×i + 1) mod 16else if 32 ≤ i ≤ 47f := b xor c xor dg := (3×i + 5) mod 16else if 48 ≤ i ≤ 63f := c xor (b or (not d))g := (7×i) mod 16temp := dd := cc := bb := ((a + f + k[i] + w[g]) leftrotate r[i]) + ba := temp//Add this chunk's hash to result so far:h0 := h0 + ah1 := h1 + bh2 := h2 + ch3 := h3 + dvar int digest := h0 append h1 append h2 append h3//(expressed as little-endian)5.3.3 MD5的OpenSSL加密实例图5-2中所示，使用了OpenSSL中的md5命令对文本yuan.txt进行了

超市后台管理系统设计与实现 吴波 -《华南理工大学硕士论文》- 2010-11-10（是否引证：否）

1.密结束，输出的结果就是我们看不清楚的 16 位加密密文了，程序实现的伪代码如下： var int[64] r, k //r specifies the per-round shift amounts r[0..15] : = {7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22} r[16..31] : = {5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20} r[32..47] : = {4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23} r[48..63] : = {6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21} //Use binary integer part of the sines of integers as constants: for i from 0 to 63 k[i] := floor (abs (sin (i + 1)) × 2^32) //Initialize variables: var int h0 := 0x67452301 var int h1 := 0xEFCDAB89 var int h2 := 0x98BADCFE var int h3 := 0x10325476 //Pre-processing: append "1" bit to message append "0" bits until message length in bits ≡ 448 （mod 512) append bit length of message as 64-bit little-endian integer to message //Process the message in successive 512-bit chunks: for each 512-bit chunk of message break chunk into sixteen 32-bit little-endian words w[i], 0 ≤ i ≤ 15 35华南理工大学硕士学位论文 //Ini

2. i ≤ 15 35华南理工大学硕士学位论文 //Initialize hash value for this chunk: var int a := h0 var int b := h1 var int c := h2 var int d := h3 //Main loop: for i from 0 to 63 if 0 ≤ i ≤ 15 then f :=（b and c）or（（not b）and d) g := i else if 16 ≤ i ≤ 31 f :=（d and b）or（（not d）and c) g :=（5×i + 1）mod 16 else if 32 ≤ i ≤ 47 f := b xor c xor d g :=（3×i + 5）mod 16 else if 48 ≤ i ≤ 63 f := c xor（b or（not d)) g :=（7×i）mod 16 temp := d d := c c := b b :=（（a + f + k[i] + w[g]）leftrota

3.+ f + k[i] + w[g]）leftrotate r[i]）+ b a := temp //Add this chunk's hash to result so far: h0 := h0 + a h1 := h1 + b h2 := h2 + c h3 := h3 + d var int digest := h0 append h1 append h2 append h3 //（expressed as little-endian) 4.5.2 报表多元化查询 在销售管理的功能模块中，我专门新增了报表查询的模块，可以方便经营管理者通

基于SHA-256算法的嵌入式软件保护技术研究 郑佳敏 -《华东师范大学硕士论文》- 2014-03-22（是否引证：否）

1.代码实现。该伪代码中常数、是这样选择的：在第i步，A,是23S<|sin(/)|的整数部分，i的单位是弧度。//Note: All vairables are unsigned 32 bits and wrap modulo 2^32 when calculatingvar int[64] r, k//r specifies the per-round shitf amountsr[0.. 15]：-{7，12，17, 22，7，12，17, 22, 7，12，17,22，7，12, 17, 22}r[16.

2.untsr[0.. 15]：-{7，12，17, 22，7，12，17, 22, 7，12，17,22，7，12, 17, 22}r[16..3】]:-{5, 9, 14,20, 5，9, 14,20, 5，（），14，20, 5, 9, 14,20}r[32..47]：-{4,11,16,23, 4，11，16，23，4，11，16，23，4, 11, 16,23}r[48..63

3.，11，16，23，4, 11, 16,23}r[48..63]：-{6,10,15,21, 6,10, 15,21, 6, 10, 15,21, 6, 10, 15,21}//Use binary integer part of the sines of integers as constants:for i from 0 to 63k[i] := lfoor(abs(sin(i +1)) X 2八32)//Initialize vairables:var int hO 0x67452301var int hi := 0xEFCDAB89var int h2 :- 0x98BADCFEvar int h3 :- 0x10325476//Pre-processing:append "1” bit to messageappend "0" bits until message length in bits 三 448 (mod 512)append bit length of message as 64-bit little-endian integer to message//Process the message in successive 512-bit chunks:for each 512-bit chunk of messagebreak chunk into sixteen 32-bit little-endian words w[i], 0 ^ i ^ 15//Initialize hash value for this chunk:var inta := hOvar int b := hivar int c := h2var int d := h311平东师范大学硕士学位论文基于SHA-256算法的嵌入式软件保护技术研究 //Main loop

4.r int d := h311平东师范大学硕士学位论文基于SHA-

256算法的嵌入式软件保护技术研究 //Main loop:for i from 0 to 63ifO ^ i ^ 15 thenf := (b and c) or ((not b) and d)g:=ielse if 16 ^ i ^ 31f := (d and b) or ((not d) and c)g := (5Xi + 1) mod 16else if 32 ^ i ^ 47f := b xor c xor dg := (3 X i + 5) mod 16else if 48《i ^ 63f :== c xor (b or (not d)) ■:::g:=(7Xi)m。dl6，，:t/ . ^ 〃 =?牙emJpT := d '.. .

5.r (not d)) ■:::g:=(7Xi)m。dl6，，:t/ . ^ 〃 =?牙emJpT := d '.. .d := cc := bb := letfrotate((a + f + k[i] + w[g])，r[i]) + ba := tempNext i//Add this chunk's hash to result so far:hO := hO + ahi := hi + bh2 := h2 + ch3 ：二 h3 + dEnd ForEachvar int digest ;= hO append hi append h2 append h3 //(expressed as little-endian)2.2.2. MD5算法的安全性分析对于MD5相对MD4所作的改进，Ri|ves(t做了概述[)]。a.在原有三

Message-Digest Algorithm 5 - 走马观花 - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证：否）

1.方式为，求余、取余、调整长度、与链接变量进行循环运算。得出结果。 是 XOR,AND,OR ,NOT 的符号。伪代码 //Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating var int[64] r,k //r specifies the per-round shift amounts r[0..15] : = {7,12,17,22,7,12,17,22,7,12,17,22,7,12,17,22} r[16..31] : = {5

2.} r[48..63] : = {6,10,15,21,6,10,15,21,6,10,15,21,6,10,15,21} //Use binary integer part of the sines of integers as constants: for i from 0 to 63 k[i] := floor(abs(sin(i 1)) × 2^32) //Initialize variables: var int h0 := 0x67452301 var int h1 := 0xEFCDAB89 var int h2 := 0x98BADCFE var int h3 := 0x10325476 //Pre-processing: append "1" bit to message append "0" bits until message length in bits ≡ 448 (mod 512) append bit length of message as 64-bit little-endian integer to message //Process the message in successive 512-bit chunks: for each 512-bit chunk of message break chunk into sixteen 32-bit little-endian words w[i],0 ≤ i ≤ 15 //Initialize hash value for this chunk: var int a := h0 var int b := h1 var int c := h2 var int d := h3 //Main loop: for i from 0 to 63 if 0 ≤ i ≤ 15 then f := (b and c) or ((not b) and d) g := i else if 16 ≤ i ≤ 31 f := (d and b) or ((not d) and c) g := (5×i 1) mod 16 else if 32 ≤ i ≤ 47 f := b xor c xor d g := (3×i 5) mod 16 else if 48 ≤ i ≤ 63 f := c xor (b or (not d)) g := (7×i) mod 16 temp := d d := c c := b b := leftrotate((a f k[i] w[g]),r[i]) b a := temp Next i //Add this chunk's hash to result so far: h0 := h0 a h1 := h1 b h2 := h2 c h3 := h3 d End ForEach var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian) MD5散列一般128位的MD5散列被表示为32位十六进制数字。以下是一个43位长的仅ASCII字母列的MD5散列： M

MD5_百度百科 -《网络（http://baike.baidu.c）》-（是否引证：否）

1.74ab98d277d9f5a5611c2c9f419d9f[编辑本段]算法的伪代码与标准C语言实现MD5算法的伪代码 //Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating var int64 r,k //r specifies the per-round shift amounts r 0..15 : = {7,12,17,22,7,12,17,22,7,12,17,22,7,12,17,22} r16..31 : = {5,9

2.23} r48..63 : = {6,10,15,21,6,10,15,21,6,10,15,21,6,10,15,21} //Use binary integer part of the sines of integers as constants: for i from 0 to 63 ki := floor(abs(sin(i + 1)) × 2^32) //Initialize variables: var int h0 := 0x67452301 var int h1 := 0xEFCDAB89 var int h2 := 0x98BADCFE var int h3 := 0x10325476 //Pre-processing: append "1" bit to message append "0" bits until message length in bits ≡ 448 (mod 512) append bit length of message as 64-bit little-endian integer to message //Process the message in successive 512-bit chunks: for each 512-bit chunk of message break chunk into sixteen 32-bit little-endian words wi,0 ≤ i ≤ 15 //Initialize hash value for this chunk: var int a := h0 var int b := h1 var int c := h2 var int d := h3 //Main loop: for i from 0 to 63 if 0 ≤ i ≤ 15 then f := (b and c) or ((not b) and d) g := i else if 16 ≤ i ≤ 31 f := (d and b) or ((not d) and c) g := (5×i + 1) mod 16 else if 32 ≤ i ≤ 47 f := b xor c xor d g := (3×i + 5) mod 16 else if 48 ≤ i ≤ 63 f := c xor (b or (not d)) g := (7×i) mod 16 temp := d d := c c := b b := ((a + f + ki + wg) leftrotate ri) + b a := temp //Add this chunk's hash to result so far: h0 := h0 + a h1 := h1 + b h2 := h2 + c h3 := h3 + d var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian)标准C语言实现具体的一个MD5实现 /* * md5 -- compute and check MD5 messag

简要分析用MD5加密算法加密信息（精编版）- Auspicious Cloud's View - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》-（是否引证：否）

1.m_lpszBuffer[nIndex],&Input[i],nInputLen-i); } 八、MD5加密算法（伪代码描述） //Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating var int[64] r,k //r specifies the per-round shift amounts r[0..15] : = {7,12,17,22,7,12,17,22,7,12,17,22,7,12,17,22} r[16..31] : = {5

2.} r[48..63] : = {6,10,15,21,6,10,15,21,6,10,15,21,6,10,15,21} //Use binary integer part of the sines of integers as constants: for i from 0 to 63 k[i] := floor(abs(sin(i + 1)) × 2^32) //Initialize variables: var int h0 := 0x67452301

var int h1 := 0xEFCDAB89 var int h2 := 0x98BADCFE var int h3 := 0x10325476 //Pre-processing: append "1" bit to message append "0" bits until message length in bits ≡ 448 (mod 512) append bit length of message as 64-bit little-endian integer to message //Process the message in successive 512-bit chunks: for each 512-bit chunk of message break chunk into sixteen 32-bit little-endian words w[i],0 ≤ i ≤ 15 //Initialize hash value for this chunk: var int a := h0 var int b := h1 var int c := h2 var int d := h3 //Main loop: for i from 0 to 63 if 0 ≤ i ≤ 15 then f := (b and c) or ((not b) and d) g := i else if 16 ≤ i ≤ 31 f := (d and b) or ((not d) and c) g := (5×i + 1) mod 16 else if 32 ≤ i ≤ 47 f := b xor c xor d g := (3×i + 5) mod 16 else if 48 ≤ i ≤ 63 f := c xor (b or (not d)) g := (7×i) mod 16 temp := d d := c c := b b := ((a + f + k[i] + w[g]) leftrotate r[i]) + b a := temp //Add this chunk's hash to result so far: h0 := h0 + a h1 := h1 + b h2 := h2 + c h3 := h3 + d var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian) 本篇博文的版权信息如下：

现在才知道其实中国对美国军方最具威胁的不是核武也不是打卫星技术.原来是中国的密码破译能力.中国数字家已将西方认为不可能破译的MD5和SHA-1成功破译_标点 - 韶韶 - 记者版_媒体 - 《网络（http://www.xici.net/）》- （是否引证：否）

1.方式为，求余、取余、调整长度、与链接变量进行循环运算。得出结果。 是 XOR,AND,OR ,NOT 的符号。伪代码 //Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating var int[64] r,k //r specifies the per-round shift amounts r[0..15] := {7,12,17,22,7,12,17,22,7,12,17,22,7,12,17,22} r[16..31] := {5,9

2. r[48..63] := {6,10,15,21,6,10,15,21,6,10,15,21,6,10,15,21} //Use binary integer part of the sines of integers as constants: for i from 0 to 63 k[i] := floor(abs(sin(i + 1)) × 2^32) //Initialize variables: var int h0 := 0x67452301 var int h1 := 0xEFCDAB89 var int h2 := 0x98BADCFE var int h3 := 0x10325476 //Pre-processing: append "1" bit to message append "0" bits until message length in bits ≡ 448 (mod 512) append bit length of message as 64-bit little-endian integer to message //Process the message in successive 512-bit chunks: for each 512-bit chunk of message break chunk into sixteen 32-bit little-endian words w[i],0 ≤ i ≤ 15 //Initialize hash value for this chunk: var int a := h0 var int b := h1 var int c := h2 var int d := h3 //Main loop: for i from 0 to 63 if 0 ≤ i ≤ 15 then f := (b and c) or ((not b) and d) g := i else if 16 ≤ i ≤ 31 f := (d and b) or ((not d) and c) g := (5×i + 1) mod 16 else if 32 ≤ i ≤ 47 f := b xor c xor d g := (3×i + 5) mod 16 else if 48 ≤ i ≤ 63 f := c xor (b or (not d)) g := (7×i) mod 16 temp := d d := c c := b b := ((a + f + k[i] +

w[g]) leftrotate r[i]) + b a := temp //Add this chunk's hash to result so far: h0 := h0 + a h1 := h1 + b h2 := h2 + c h3 := h3 + d var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian) MD5散列一般128位的MD5散列被表示为32位十六进制数字。以下是一个43位长ASCII字母列的MD5散列： MD5

MD5算法Qt实现 - tandesir的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》-（是否引证：否）

1.位字节A开始，高位字节D结束。 *下述摘自维基百科 http://zh.wikipedia.org/zh/MD5 伪代码 //Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating var int[64] r,k //r specifies the per-round shift amounts r[0..15] : = {7,12,17,22,7,12,17,22,7,12,17,22,7,12,17,22} r[16..31] : = {5

2.} r[48..63] : = {6,10,15,21,6,10,15,21,6,10,15,21,6,10,15,21} //Use binary integer part of the sines of integers as constants: for i from 0 to 63 k[i] := floor(abs(sin(i + 1)) × 2^32) //Initialize variables: var int h0 := 0x67452301 var int h1 := 0xEFCDAB89 var int h2 := 0x98BADCFE var int h3 := 0x10325476 //Pre-processing: append "1" bit to message append "0" bits until message length in bits ≡ 448 (mod 512) append bit length of message as 64-bit little-endian integer to message //Process the message in successive 512-bit chunks: for each 512-bit chunk of message break chunk into sixteen 32-bit little-endian words w[i],0 ≤ i ≤ 15 //Initialize hash value for this chunk: var int a := h0 var int b := h1 var int c := h2 var int d := h3 //Main loop: for i from 0 to 63 if 0 ≤ i ≤ 15 then f := (b and c) or ((not b) and d) g := i else if 16 ≤ i ≤ 31 f := (d and b) or ((not d) and c) g := (5×i + 1) mod 16 else if 32 ≤ i ≤ 47 f := b xor c xor d g := (3×i + 5) mod 16 else if 48 ≤ i ≤ 63 f := c xor (b or (not d)) g := (7×i) mod 16 temp := d d := c c := b b := leftrotate((a + f + k[i] + w[g]),r[i]) + b a := temp Next i //Add this chunk's hash to result so far: h0 := h0 + a h1 := h1 + b h2 := h2 + c h3 := h3 + d End ForEach var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian) 【代码解释】 1 步骤3的实现 _state[0] = 0x67452301; _state[1] = 0xefcdab

简要分析用MD5加密算法加密信息（精编版） - Auspicious Cloud's View - CSDN博客 -《网络（http://blog.csdn.net）》-（是否引证：否）

1.m_lpszBuffer[nIndex],&Input[i],nInputLen-i); } 八、MD5加密算法（伪代码描述） //Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating var int[64] r,k //r specifies the per-round shift amounts r[0..15] : = {7,12,17,22,7,12,17,22,7,12,17,22,7,12,17,22} r[16..31] : = {5

2.} r[48..63] : = {6,10,15,21,6,10,15,21,6,10,15,21,6,10,15,21} //Use binary integer part of the sines of integers as constants: for i from 0 to 63 k[i] := floor(abs(sin(i + 1)) × 2^32) //Initialize variables: var int h0 := 0x67452301 var int h1 := 0xEFCDAB89 var int h2 := 0x98BADCFE var int h3 := 0x10325476 //Pre-processing: append "1" bit to message append "0" bits until message length in bits ≡ 448 (mod 512) append bit length of message as 64-bit little-endian integer to message //Process the message in successive 512-bit chunks: for each 512-bit chunk of message break chunk into sixteen 32-bit little-endian words w[i],0 ≤ i ≤ 15 //Initialize hash value for this chunk: var int a := h0 var int b := h1 var int c := h2 var int d := h3 //Main loop: for i from 0 to 63 if 0 ≤ i ≤ 15 then f := (b and c) or ((not b) and d) g := i else if 16 ≤ i ≤ 31 f := (d and b) or ((not d) and c) g := (5×i + 1) mod 16 else if 32 ≤ i ≤ 47 f := b xor c xor d g := (3×i + 5) mod 16 else if 48 ≤ i ≤ 63 f := c xor (b or (not d)) g := (7×i) mod 16 temp := d d := c c := b b := ((a + f + k[i] + w[g]) leftrotate r[i]) + b a := temp //Add this chunk's hash to result so far: h0 := h0 + a h1 := h1 + b h2 := h2 + c h3 := h3 + d var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian) 本篇博文的版权信息如下：

什么是MD5 - 豆丁网 - 《互联网文档资源（http://www.docin.com）》- 2017-4-28 2:18:33（是否引证：否）

1.ab98d277d9f5a5611c2c9f419d9f [编辑本段]算法的伪代码与标准C语言实现 MD5算法的伪代码 //Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating var int[64] r,k //r specifies the per-round shift amounts r[0..15] : = {7,12,17,22,7,12,17,22,7,12,17,22,7,12,17,22} r[16..31] : = {5

2.} r[48..63] : = {6,10,15,21,6,10,15,21,6,10,15,21,6,10,15,21} //Use binary integer part of the sines of integers as constants: for i from 0 to 63 k[i] := floor(abs(sin(i + 1)) × 2^32) //Initialize variables: var int h0 := 0x67452301 var int h1 := 0xEFCDAB89 var int h2 := 0x98BADCFE var int h3 := 0x10325476 //Pre-processing: append "1" bit to message append "0" bits until message length in bits ≡ 448 (mod 512) append bit length of message as 64-bit little-endian integer to message //Process the message in successive 512-bit chunks: for each 512-bit chunk of message break chunk into sixteen 32-bit little-endian words w[i],0 ? i ? 15 //Initialize hash value for this chunk: var int a := h0 var int b := h1 var int c := h2 var int d := h3 //Main loop: for i from 0 to 63 if 0 ? i ? 15 then f := (b and c) or ((not b) and d) g := i else if 16

? i ? 31 f := (d and b) or ((not d) and c) g := (5×i + 1) mod 16 else if 32 ? i ? 47 f := b xor c xor d g := (3×i + 5) mod 16 else if 48 ? i ? 63 f := c xor (b or (not d)) g := (7×i) mod 16 temp := d d := c c := b b := ((a + f + k[i] + w[g]) leftrotate r[i]) + b a := temp //Add this chunk's hash to result so far: h0 := h0 + a h1 := h1 + b h2 := h2 + c h3 := h3 + d var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian) 标准C语言实现 具体的一个MD5实现 /* * md5 -- compute and check MD5 message

MD5 - yjg428的专栏 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否）

1.9939a25efabaef3b87e2cbfe641315 编辑本段算法的伪代码与标准C语言实现 MD5算法的伪代码 //Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating var int[64] r,k //r specifies the per-round shift amounts r[0..15] : = {7,12,17,22,7,12,17,22,7,12,17,22,7,12,17,22} r[16..31] : = {5

2.} r[48..63] : = {6,10,15,21,6,10,15,21,6,10,15,21,6,10,15,21} //Use binary integer part of the sines of integers as constants: for i from 0 to 63 k[i] := floor(abs(sin(i 1)) × 2^32) //Initialize variables: var int h0 := 0x67452301 var int h1 := 0xEFCDAB89 var int h2 := 0x98BADCFE var int h3 := 0x10325476 //Pre-processing: append "1" bit to message append "0" bits until message length in bits ≡ 448 (mod 512) append bit length of message as 64-bit little-endian integer to message //Process the message in successive 512-bit chunks: for each 512-bit chunk of message break chunk into sixteen 32-bit little-endian words w[i],0 ≤ i ≤ 15 //Initialize hash value for this chunk: var int a := h0 var int b := h1 var int c := h2 var int d := h3 //Main loop: for i from 0 to 63 if 0 ≤ i ≤ 15 then f := (b and c) or ((not b) and d) g := i else if 16 ≤ i ≤ 31 f := (d and b) or ((not d) and c) g := (5×i 1) mod 16 else if 32 ≤ i ≤ 47 f := b xor c xor d g := (3×i 5) mod 16 else if 48 ≤ i ≤ 63 f := c xor (b or (not d)) g := (7×i) mod 16 temp := d d := c c := b b := ((a f k[i] w[g]) leftrotate r[i]) b a := temp //Add this chunk's hash to result so far: h0 := h0 a h1 := h1 b h2 := h2 c h3 := h3 d var int digest := h0 append h1 append h2 append

3. result so far: h0 := h0 a h1 := h1 b h2 := h2 c h3 := h3 d var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian) 标准C语言实现 具体的一个MD5实现 /* * md5 -- compute and check MD5 message

md5算法及其C语言的实现 - 网络与安全,mayu8758,拥抱开源，把握未来！ -《网络（http://blog.opendige）》- （是否引证：否）

1.= d174ab98d277d9f5a5611c2c9f419d9f 算法的伪代码与标准C语言实现 MD5算法的伪代码 //Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating var int[64] r,k //r specifies the per-round shift amounts r[0..15] : = {7,12,17,22,7,12,17,22,7,12,17,22,7,12,17,22} r[16..31] : = {5

2.} r[48..63] : = {6,10,15,21,6,10,15,21,6,10,15,21,6,10,15,21} //Use binary integer part of the sines of integers as constants: for i from 0 to 63 k[i] := floor(abs(sin(i + 1)) × 2^32) //Initialize variables: var int h0 := 0x67452301 var int h1 := 0xEFCDAB89 var int h2 := 0x98BADCFE var int h3 := 0x10325476 //Pre-processing: append "1" bit to message append "0" bits until message length in bits ≡ 448 (mod 512) append bit length of message as 64-bit little-endian integer to message //Process the message in successive 512-bit chunks: for each 512-bit chunk of message break chunk into sixteen 32-bit little-endian words w[i],0 ≤ i ≤ 15 //Initialize hash value for this chunk: var int a := h0 var int b := h1 var int c := h2 var int d := h3 //Main loop: for i from 0 to 63 if 0 ≤ i ≤ 15 then f := (b and c) or ((not b) and d) g := i else if 16 ≤ i ≤ 31 f := (d and b) or ((not d) and c) g := (5×i + 1) mod 16 else if 32 ≤ i ≤ 47 f := b xor

3.nd c) g := (5×i + 1) mod 16 else if 32 ≤ i ≤ 47 f := b xor c xor d g := (3×i + 5) mod 16 else if 48 ≤ i ≤ 63 f := c xor (b or (not d)) g := (7×i) mod 16 temp := d d := c c := b b := ((a + f + k[i] + w[g]) leftrotate r[i]) + b a := temp //Add this chunk's hash to result so far: h0 := h0 + a h1 := h1 + b h2 := h2 + c h3 := h3 + d var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian) 标准C语言实现 具体的一个MD5实现 /* * md5 -- compute and check MD5 message

常数ti是4294967296*abs_caoaugt - 《网络（http://blog.sina.com）》- （是否引证：否）

1.d076287532e86365e841e92bfc50d8c. ' 验证哈希... ' 哈希值是相同的。 伪代码实现 //Note: All variables are unsigned 32 bits and wrap modulo 2^32 when calculating var int64 r,k //r specifies the per-round shift amounts r 0..15 := {7,12,17,22,7,12,17,22,7,12,17,22,7,12,17,22} r16..31 := {5,9,1

2.3} r48..63 := {6,10,15,21,6,10,15,21,6,10,15,21,6,10,15,21} //Use binary integer part of the sines of integers as

constants: for i from 0 to 63 ki := floor(abs(sin(i + 1)) × 2^32) //Initialize variables: var int h0 := 0x67452301 var int h1 := 0xEFCDAB89 var int h2 := 0x98BADCFE var int h3 := 0x10325476 //Pre-processing: append "1" bit to message append "0" bits until message length in bits ≡ 448 (mod 512) append bit length of message as 64-bit little-endian integer to message //Process the message in successive 512-bit chunks: for each 512-bit chunk of message break chunk into sixteen 32-bit little-endian words wi,0 ≤ i ≤ 15 //Initialize hash value for this chunk: var int a := h0 var int b := h1 var int c := h2 var int d := h3 //Main loop: for i from 0 to 63 if 0 ≤ i ≤ 15 then f := (b and c) or ((not b) and d) g := i else if 16 ≤ i ≤ 31 f := (d and b) or ((not d) and c) g := （ 5×i + 1 ） mod 16 else if 32 ≤ i ≤ 47 f := b xor c xor d g := （ 3×i + 5 ） mod 16 else if 48 ≤ i ≤ 63 f := c xor (b or (not d)) g := （ 7×i) mod 16 temp := d d := c c := b b := ((a + f + ki + wg) leftrotate ri) + b a := temp //Add this c

3.p := d d := c c := b b := ((a + f + ki + wg) leftrotate ri) + b a := temp //Add this chunk's hash to result so far: h0 := h0 + a h1 := h1 + b h2 := h2 + c h3 := h3 + d var int digest := h0 append h1 append h2 append h3 //(expressed as little-endian) 编辑本段MD5优势Van oorschot和Wiener曾经考虑过一个在散列中暴力搜寻冲突的函数（brute-force

电子印章的设计与实现 龙宇 -《大学生论文联合比对库》-2015-05-27（是否引证：否）

1.seal client business.Key word:MD5;RSA;PKI附录AMD5算法伪代码//Note:All variables are unsigned 32 bits and wrap modulo 2^32 when calculatingvar int[64]r,k//r specifies the per-round shift amountsr[0..15]：={7,12,17,22,7,12,17,22,7,12,17,22,7,12,17,22}r[16..31]：={5,9,

2.6,23}r[48..63]：={6,10,15,21,6,10,15,21,6,10,15,21,6,10,15,21}//Use binary integer part of the sines of integers as constants:for i from 0 to 63k[i]:=floor(abs(sin(i+1))×2^32)//Initialize variables:var int h0:=0x67452301var int h1:=0xEFCDAB89var int h2:=0x98BADCFEvar int h3:=0x10325476//Pre-processing:append"1"bit to messageappend"0"bits until message length in bits≡448(mod 512)append bit length of message as 64-bit little-endian integer to message//Process the message in successive 512-bit chunks:for each 512-bit chunk of messagebreak chunk into sixteen 32-bit little-endian words w[i],0≤i≤15//Initialize hash value for this chunk:var int a:=h0var int b:=h1var int c:=h2var int d:=h3//Main loop:for i from 0 to 63if 0≤i≤15 thenf:=(b and c)or((not b)and d)g:=ielse if 16≤i≤31f:=(d and b)or((not d)and

		c)g:=(5×i+1)mod 16else if 32≤i≤47f:=b xor c xor dg:=(3×i+5)mod 16else if 48≤i≤63 3.c)g:=(5×i+1)mod 16else if 32≤i≤47f:=b xor c xor dg:=(3×i+5)mod 16else if 48≤i≤63f:=c xor(b or(not d))g:=(7×i)mod 16temp:=dd:=cc:=bb:=leftrotate((a+f+k[i]+w[g]),r[i]) 4. xor(b or(not d))g:=(7×i)mod 16temp:=dd:=cc:=bb:=leftrotate((a+f+k[i]+w[g]),r[i])+ba:=tempNext i//Add this chunk's hash to result so far:h0:=h0+ah1:=h1+bh2:=h2+ch3:=h3+dEnd ForEachvar int di 5.ash to result so far:h0:=h0+ah1:=h1+bh2:=h2+ch3:=h3+dEnd ForEachvar int digest:=h0 append h1 append h2 append h3//(expressed as little-endian)附录B定理：对于自然数a,b,c,d,e，如果a%e=b,c%e=d，则ac%e=bd%e。证明过程：

指　标
疑似剽窃文字表述

1. 经过程序流程，生成四个32位数据，最后联合起来成为一个128-bits散列。基本方式为，求余，取余，调整长度，与链接变量进行循环运算，得出结果。
2. 运算过程，一个MD5运算—由类似的64次循环构成，分成4组16次。F 一个非线性函数；一个函数运算一次。Mi 表示一个 32-bits 的输入数据，Ki 表示一个 32-bits 常数，用来完成每次不同的计算。

5. 第四章基于MapReduce框架的热点区域挖掘及可视化

总字数：4884

相似文献列表　文字复制比：7.7%(376)　疑似剽窃观点：(0)

1	聚类之层次聚类、基于划分的聚类（… - bluewater的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	5.8%（281） 是否引证：否
2	基于大数据分析的集装箱箱修点合理性分析 汪子君 -《大学生论文联合比对库》- 2017-05-26	5.7%（278） 是否引证：否
3	201224060201_李丹宁_基于出租车GPS数据的居民出行热点方法研究 李丹宁 -《大学生论文联合比对库》- 2016-06-06	5.7%（278） 是否引证：否
4	201224060201_李丹宁_基于出租车GPS数据的居民出行热点方法研究 李丹宁 -《大学生论文联合比对库》- 2016-06-13	5.7%（278） 是否引证：否
5	基于集装箱维修数据的集装箱箱修点合理性分析 汪子君 -《大学生论文联合比对库》- 2017-05-31	5.7%（278） 是否引证：否
6	数据挖掘技术中聚类算法的研究 许晨俊 -《大学生论文联合比对库》- 2017-05-31	5.7%（278） 是否引证：否
7	【数据挖掘】聚类算法总结 -- 数据科学自媒体 -- 传送门 -《网络（http://chuansong.me/）》- 2016	5.6%（274） 是否引证：否
8	聚类算法总结 - 南山牧笛的博客 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	5.6%（274） 是否引证：否
9	基于密度的聚类 - suichen1的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	5.6%（274） 是否引证：否
10	聚类算法 - 皮皮blog - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	5.6%（274） 是否引证：否
11	11668144_马晨阳_基于密度的数据挖掘聚类方法研究 马晨阳 -《大学生论文联合比对库》- 2017-06-10	5.3%（260） 是否引证：否
12	基于云计算分布式技术的海量AIS数据挖掘系统设计与实现	2.6%（129）

	原文内容	相似内容来源
1	此处有 95 字相似 . 化简：reduce这一阶段接收由map阶段传过来的key和values变量，如果map阶段一个key对应的valu es条数很多，可能会多次调用reduce方法，即前一次reduce的结果可能被包含在values中再次传递给reduce方法，这也要求，reduce的返回结果需要和value的结构保持一致。 因此本文不在reduce中进行聚类分析，只是将values递归原样返回。 聚类：finalize方法在reduce函数	Mongodb中Mapreduce特性与原理 - shift_alt - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.”中的变量；如果一个key只有一个value，那么mongodb就不会调用reduce方法。可能一个key对应的values条数很多，将会调用多次reduce，即前一次reduce的结果可能被包含在values中再次传递给reduce方法，这也要求，reduce返回的结果需要value的结构保持一致。同样，reduce返回的数据尺寸不能大于8M（document最大尺寸的一半，因为reduce的结果可能会作为input
2	此处有 281 字相似 选择阈值，使聚类结果最优，是决定聚类结果性能的关键因素。下面描述一种确定两个阈值的启发式算法。 1) k-dist: 给定数据集P={p(i); i=0,1,…n}，对于任意点P(i)，计算点P(i)到集合P的子集S={p(1), p(2), …, p(i-1), p(i+1), …, p(n)}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D={d(1), d(2), …, d(k-1), d(k), d(k+1), …,d(n)}，则d(k)就被称为k-距离。即点p(i)到所有点（除了p(i)点）之间距离第k近的距离。对聚类集合中每个点p(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1), e(2), …, e(n)}。 2) 对于任意一点p，如果令=k-dist，MinPts=k，那么所有k-dist小于等于p的点都将是核心点。如果能在集合D	【数据挖掘】聚类算法总结 -- 数据科学自媒体 -- 传送门 - 《网络（http://chuansong.me/）》- （是否引证：否） 1.ps的邻域内的点的个数不少于MinPts，则称点P为核心点。 ③DBSCAN聚类使用到一个k-距离的概念，k-距离是指：给定数据集P={p(i);i=0,1,…n}，对于任意点P(i)，计算点P(i)到集合D的子集S={p(1),p(2),…,p(i-1),p(i+1),…,p(n)}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D={d(1),d(2),…,d(k-1),d(k),d(k+1),…,d(n)}，则d(k)就被称为k-距离。也就是说，k-距离是点p(i)到所有点（除了p(i)点）之间距离第k近的距离。对待聚类集合中每个点p(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1),e(2),…,e(n)}。 ④根据经验计算半径Eps：根据得到的所有点的k-距离集合E，对集合E进行升序排序后得到k-距离集合E’，需要拟合一条排 聚类算法总结 - 南山牧笛的博客 - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.ps的邻域内的点的个数不少于MinPts，则称点P为核心点。 ③DBSCAN聚类使用到一个k-距离的概念，k-距离是指：给定数据集P={p(i);i=0,1,…n}，对于任意点P(i)，计算点P(i)到集合D的子集S={p(1),p(2),…,p(i-1),p(i1),…,p(n)}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D={d(1),d(2),…,d(k-1),d(k),d(k1),…,d(n)}，则d(k)就被称为k-距离。也就是说，k-距离是点p(i)到所有点（除了p(i)点）之间距离第k近的距离。对待聚类集合中每个点p(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1),e(2),…,e(n)}。 ④根据经验计算半径Eps：根据得到的所有点的k-距离集合E，对集合E进行升序排序后得到k-距离集合E’，需要拟合一条排 基于密度的聚类 - suichen1的专栏 - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.ps的邻域内的点的个数不少于MinPts，则称点P为核

心点。 ③DBSCAN聚类使用到一个k-距离的概念，k-距离是指：给定数据集P={p(i);i=0,1,…n}，对于任意点P(i)，计算点P(i)到集合D的子集S={p(1),p(2),…,p(i-1),p(i1),…,p(n)}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D={d(1),d(2),…,d(k-1),d(k),d(k1),…,d(n)}，则d(k)就被称为k-距离。也就是说，k-距离是点p(i)到所有点（除了p(i)点）之间距离第k近的距离。对待聚类集合中每个点p(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1),e(2),…,e(n)}。 ④根据经验计算半径Eps：根据得到的所有点的k-距离集合E，对集合E进行升序排序后得到k-距离集合E'，需要拟合一条排

聚类算法 - 皮皮blog - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证：否）

1.ps的邻域内的点的个数不少于MinPts，则称点P为核心点。 ③DBSCAN聚类使用到一个k-距离的概念，k-距离是指：给定数据集P={p(i);i=0,1,…n}，对于任意点P(i)，计算点P(i)到集合D的子集S={p(1),p(2),…,p(i-1),p(i+1),…,p(n)}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D={d(1),d(2),…,d(k-1),d(k),d(k+1),…,d(n)}，则d(k)就被称为k-距离。也就是说，k-距离是点p(i)到所有点（除了p(i)点）之间距离第k近的距离。对待聚类集合中每个点p(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1),e(2),…,e(n)}。 ④根据经验计算半径Eps：根据得到的所有点的k-距离集合E，对集合E进行升序排序后得到k-距离集合E'，需要拟合一条排

聚类之层次聚类、基于划分的聚类（… - bluewater的专栏 - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证：否）

1.Eps的邻域内的点的个数不少于MinPts，则称点P为核心点。 DBSCAN聚类使用到一个k-距离的概念，k-距离是指：给定数据集P={p(i);i=0,1,…n}，对于任意点P(i)，计算点P(i)到集合D的子集S={p(1),p(2),…,p(i-1),p(i+1),…,p(n)}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D={d(1),d(2),…,d(k-1),d(k),d(k+1),…,d(n)}，则d(k)就被称为k-距离。也就是说，k-距离是点p(i)到所有点（除了p(i)点）之间距离第k近的距离。对待聚类集合中每个点p(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1),e(2),…,e(n)}。 根据经验计算半径Eps：根据得到的所有点的k-距离集合E，对集合E进行升序排序后得到k-距离集合E'，需要拟合一条排序后

201224060201_李丹宁_基于出租车GPS数据的居民出行热点方法研究 李丹宁 -《大学生论文联合比对库》- 2016-06-06（是否引证：否）

1.ps的邻域内的点的个数不少于MinPts，则称点P为核心点。DBSCAN聚类使用到一个k-距离的概念，k-距离是指：给定数据集P={p(i);i = 0,1 ...n}，对于任意点P(i)，计算点P(i)到集合D的子集S={p(1), p(2), …, p(i-1), p(i+1), …, p(n)}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D = {d(1),

d(2), …, d(k-1), d(k), d(k+1), …,d(n)}，则d(k)就被称为k-距离。也就是说，k-距离是点p(i)到所有点（除了p(i)点）之间距离第k近的距离。对待聚类集合中每个点p(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1), e(2), …, e(n)}。DBSCAN算法是一种基于密度的聚类方法。采用这种方法对热点区域进行提取时，同样也是以所有出租车的上下车的位置进行聚类，

201224060201_李丹宁_基于出租车GPS数据的居民出行热点方法研究_李丹宁 -《大学生论文联合比对库》- 2016-06-13（是否引证：否）

1.ps的邻域内的点的个数不少于MinPts，则称点P为核心点。DBSCAN聚类使用到一个k-距离的概念，k-距离是指：给定数据集P={p(i);i = 0,1 ...n}，对于任意点P(i)，计算点P(i)到集合D的子集S={p(1), p(2), …, p(i-1), p(i+1), …, p(n)}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D = {d(1), d(2), …, d(k-1), d(k), d(k+1), …,d(n)}，则d(k)就被称为k-距离。也就是说，k-距离是点p(i)到所有点（除了p(i)点）之间距离第k近的距离。对待聚类集合中每个点p(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1), e(2), …, e(n)}。DBSCAN算法是一种基于密度的聚类方法。采用这种方法对热点区域进行提取时，同样也是以所有出租车的上下车的位置进行聚类，

基于大数据分析的集装箱箱修点合理性分析_汪子君 -《大学生论文联合比对库》- 2017-05-26（是否引证：否）

1.以点P为中心的邻域内最少点的数量（MinPts）。即k值。③DBSCAN聚类使用到一个k-距离的概念。k-距离是指：给定数据集P={p(i);i=0,1,…n}，对于任意点P(i)，计算点P(i)到集合D的子集S={p(1), p(2), …, p(i-1), p(i+1), …, p(n)}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D={d(1), d(2), …, d(k-1), d(k), d(k+1),…,d(n)}，则d(k)就被称为k-距离。也就是说，k-距离是点p(i)到所有点（除了p(i)点）之间距离第k近的距离。对待聚类集合中每个点p(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1), e(2), …, e(n)}。p(i)到所有点（除了p(i)点）之间距离第k近的距离。④根据经验计算半径Eps，急剧变化位置所对应的k-距离的值，

基于集装箱维修数据的集装箱箱修点合理性分析_汪子君 -《大学生论文联合比对库》- 2017-05-31（是否引证：否）

1.以点M为中心的邻域内最少点的数量（MinMts）。即k值。③DBSCAN聚类使用到一个k-距离的概念。k-距离是指：给定数据集M={m(i);i=0,1,…n}，对于任意点M(i)，计算点M(i)到集合D的子集S={m(1), m(2), …, m(i-1), m(i+1), …, m(n)}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D={d(1), d(2), …, d(k-1), d(k), d(k+1),…,d(n)}，则d(k)就被称为k-距离。也就是说，k-距离是点m(i)到所有点（除了m(i)点）之间距离第k近的距离。对待聚类集合中每个点m(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1), e(2), …, e(n)}。④根据经验计算半径Eps，急剧变化位置所对应的k-距离的值，确定为半径Eps的值

。（2）连通核心点生成簇①从核心

数据挖掘技术中聚类算法的研究_许晨俊 -《大学生论文联合比对库》- 2017-05-31（是否引证：否）

1.ps的邻域内的点的个数不少于MinPts，则称点P为核心点。DBSCAN聚类使用到一个k-距离的概念，k-距离是指：给定数据集P={p(i); i=0,1,…n}，对于任意点P(i)，计算点P(i)到集合D的子集S={p(1), p(2), …, p(i-1), p(i+1), …, p(n)}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D={d(1), d(2), …, d(k-1), d(k), d(k+1), …,d(n)}，则d(k)就被称为k-距离。也就是说，k-距离是点p(i)到所有点（除了p(i)点）之间距离第k近的距离。对待聚类集合中每个点p(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1), e(2), …, e(n)}。根据经验计算半径Eps：根据得到的所有点的k-距离集合E，对集合E进行升序排序后得到k-距离集合E'，需要拟合一条排序后

11668144_马晨阳_基于密度的数据挖掘聚类方法研究_马晨阳 -《大学生论文联合比对库》- 2017-06-10（是否引证：否）

1.CAN聚类算法会使用到一个K-距离的概念，我们先来解释一下K-距离：给定数据集M{m(1);i=1,2,3,4,...,n}，对于任意点M（i）计算点M(i)到集合D的子集S={m(1), m(2), …, m(i-1), m(i+1), …, m(n)}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D={d(1), d(2), …, d(k-1), d(k), d(k+1), …,d(n)}，则d(k)就被称为k-距离。也就是说，k-距离是点p(i)到所有点（除了p(i)点）之间距离第k近的距离。对待聚类集合中每个点M(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1), e(2), …, e(n)}。4、根据经验计算Eps、MinPts：根据前面得到的所有点的K-距离的集合E，对集合E进行升序排列后得到K-距离的集合E`，

基于云计算分布式技术的海量AIS数据挖掘系统设计与实现 尚斯年 -《大连海事大学硕士论文》- 2017-04-01（是否引证：否）

1.定距离公式之后，计算每个点之间的欧几里得距离，再计算每个点的灸距离值进行排序输出排序后的A距离值。其中A：距离为给定数据集P ={p(i); Z = 0,1, ...,n}，对于任一点p(i)，计算点p(〇到集合P的子集5 ={p⑴,p⑵,...4〇-1),?〇 + 1),...4〇1),}中所有点之间的距离，距离按照从小到大

2.任一点p(i)，计算点p(〇到集合P的子集5 ={p⑴,p⑵,...4〇-1),?〇 + 1),...4〇1),}中所有点之间的距离，距离按照从小到大的顺序排序，假设排序后的距离集合为D = {d(l),d(2),d(k-l),d(/c),...,d(n)}则離>就称之为A：距离，也就是说，A：距离

3.,d(/c),...,d(n)}则離>就称之为A：距离，也就是说，A：距离是点p(i)到除p(〇点自身以外所有点之间第A:近的距离。对集合P每个点计算A距离，最后得到所有点的灸距离集合£ = {e(l),e(2)e(n)}。图3.10为集合五的

		概率密度分布图。-29- 第3章系统设计

指　标
疑似剽窃文字表述

1. es条数很多，可能会多次调用reduce方法，即前一次reduce的结果可能被包含在values中再次传递给reduce方法，这也要求，reduce的返回结果需要和value的结构保持一致。
2. 对聚类集合中每个点p(i)都计算k-距离，最后得到所有点的k-距离集合E={e(1), e(2), …, e(n)}。
   2)

6. 第五章总结与展望

总字数：1384

相似文献列表　文字复制比：0%(0)　疑似剽窃观点：(0)

说明：1.总文字复制比：被检测论文总重合字数在总字数中所占的比例

2.去除引用文献复制比：去除系统识别为引用的文献后，计算出来的重合字数在总字数中所占的比例

3.去除本人已发表文献复制比：去除作者本人已发表文献后，计算出来的重合字数在总字数中所占的比例

4.单篇最大文字复制比：被检测文献与所有相似文献比对后，重合字数占总字数的比例最大的那一篇文献的文字复制比

5.指标是由系统根据《学术论文不端行为的界定标准》自动生成的

6.红色文字表示文字复制部分;绿色文字表示引用部分

7.本报告单仅对您所选择比对资源范围内检测结果负责

amlc@cnki.net

http://check.cnki.net/

http://e.weibo.com/u/3194559873/