聚类算法综述

伍育红
(重庆邮电大学移通学院 重庆 401539)

摘 要 数据挖掘技术可以从大量数据中发现潜在的、有价值的知识,它给人们在信息时代所积累的海量数据赋予了新的意义。随着数据挖掘技术的迅速发展,作为其重要的组成部分,网格聚类技术已经被广泛应用于数据分析、图像处理、市场研究等许多领域。网格聚类算法研究已经成为数据挖掘研究领域中非常活跃的一个研究课题。介绍了数据挖掘理论,对网格聚类算法进行了深入的分析研究。在研究了传统网格聚类算法的基础上,提出了一些改进的网格聚类算法,这些算法相比传统网格聚类算法有更好的聚类质量和效率。在分析了传统的多密度聚类算法的基础上,提出了基于网格的多密度聚类算法(Grid-based Clustering Algorithm for Multi-density)[1],该算法主要采用密度阈值递减的多阶段聚类技术提取不同密度的聚类,同时对聚类结果进行了人工干预。研究结果表明,基于网格的多密度聚类算法不仅能够对数据集进行正确的聚类,同时还能有效地弥补孤立点检测,有效地解决了传统多密度聚类算法不能有效识别孤立点和噪声的缺陷。基于网格的多密度聚类算法比传统的共享近邻 SNN 算法精度高,适合于均匀密度数据集、大部分多密度数据集,并且可以发现任意形状的聚类,对噪声数据和数据输入顺序不敏感,但对小部分多密度数据集的聚类结果不理想[1]。

关键词 网格聚类,密度阈值递减,多阶段聚类
中图法分类号 TP301 **文献标识码** A

General Overview on Clustering Algorithms

WU Yu-hong
(College of Mobile Telecommunications, Chongqing University of Posts and Telecommunications, Chongqing 401539, China)

Abstract Data mining techniques can be used to find out potential and useful knowledge from the vast amount of data, and it plays a new significant role to the stored data in the info-times. With the rapid development of the data mining techniques, the technique of grid clustering, as important parts of data mining, are widely applied to the fields such as pattern recognition, data analysis, image processing, and market research. Research on grid clustering algorithms has become a highly active topic in the data mining research. In this thesis, the author presented the theory of data mining, and deeply analyzes the algorithms of grid clustering. Based on the analysis of traditional grid clustering algorithms, we advanced some improved grid clustering algorithms that can enhance the quality and efficiency of grid clustering compared with the traditional grid clustering algorithms. Based on the analysis of traditional algorithms for multi-density, we advanced a grid-based clustering algorithm for multi-density(GDD). The GDD is a kind of the multi-stage clustering that integrates grid-based clustering, the technique of density threshold descending and border points extraction. As shown in the research, GDD algorithm can not only clusters correctly but find outliers in the dataset, and it effectively solves the problem that traditional grid algorithms can cluster only or find outliers only. The precision of GDD algorithm is better than that of SNN. The GDD algorithm works well for even density dataset and lots of multi-density datasets; it can discover clusters of arbitrary shapes; it isn't sensitive to the input order of noises and outliers data, but it is imperfect to cluster on some multi-density datasets.

Keywords Grid clustering, Density threshold descending, Multi-stage clustering

1 引言

聚类已经被广泛地研究了许多年,迄今为止,研究人员已经提出了很多聚类算法,大体上这些算法可以分为基于划分的方法、基于层次的方法、基于密度的方法、基于网格的方法、基于模型的方法。

基于网格的聚类技术首先通过将数据空间的每一维平均分割成等长的区间段,来将数据空间分成不相交的网格单元,同一单元中的点属于同一类的可能性比较大,所以落入同一网格中的点可被看作一个对象进行处理,以后所有的聚类操作都是在网格单元上进行。因此,基于网格的聚类过程与数据集中数据点的个数无关,只与网格的个数有关系,聚类的效率得到很大的提高,但网格聚类算法对参数的依赖性较大。

聚类分析所使用的数据集中,各个类的密集程度往往不

伍育红(1981—),硕士生,讲师,主要研究方向为数据挖掘、网络控制与信息安全、电子商务与现代物流。

尽相同，甚至差别很大。大多数现有的聚类算法都是致力于如何发现任意形状和大小的类，但很难有效地处理密度差别较大的数据集。多密度聚类是数据挖掘中的一项重要技术，其目标是发现多密度数据集中的不同密度的数据对象，将其分离出来进行数据的处理和分析。因此，多密度聚类算法的研究为将复杂的海量数据转换为需要的信息提供了强有力的分析手段。

能够处理多密度数据集的聚类算法有 Chameleon、共享近邻 SNN 算法、多阶段等密度线算法等[2]。Chameleon 算法可以用来处理多密度的数据集，但当数据较大时算法的时间复杂度太高。SNN 算法的优点是可以对不同密度和形状的数据集进行聚类，缺点是在多密度聚类和处理孤立点或噪声方面精度都不高。多阶段等密度线算法采用多阶段的方式，利用等密度线的思想对数据集进行聚类，它的缺点是不能有效地分离出多个类。

所以，对网格聚类算法和多密度聚类算法的研究将具有广阔的发展空间，它们今后将会在更多的领域中发挥作用。

2 数据挖掘概述

2.1 简述数据挖掘

随着数据库技术的飞速发展以及人们获取数据手段的多样化，人类所拥有的数据急剧增加，可是用于对这些数据进行分析处理的工具却很少。目前数据库系统所能做到的只是对数据库中已有的数据进行存取和简单的操作，人们通过这些数据所获得的信息量仅仅是整个数据库所包含的信息量的很少一部分，隐藏在这些数据之后的更重要的信息是关于这些数据的整体特征的描述及对其发展趋势的预测，这些信息在决策生成的过程中具有重要的参考价值。这就引起了对强有力的数据分析工具的急切需求。快速增长的海量数据收集、存放在大型和大量数据库中，没有强有力的工具，理解它们已经远远超出了人的能力。

面对这种挑战，数据库中的知识发现（Knowledge Discovery in Databases，KDD）技术[2]逐渐发展起来。KDD 就是从大量、不完全、有噪声的异质模糊数据中挖掘隐含的潜在的有用知识的过程，它不但能够学习已有的知识，而且可以发现未知的规律。同时，KDD 也是一门新兴的交叉学科，汇聚了数据库、人工智能、统计、可视化和并行计算等不同领域的研究人员。

一般将 KDD 中进行知识学习的阶段称为数据挖掘（Data Mining），它是整个数据库中的知识发现过程中的一个非常重要的处理环节，所以两者往往混用。一般而言，在数据库、统计和数据分析等工程应用领域称为数据挖掘，而在 AI 和机器学习界等研究领域人们则称之为数据库中的知识发现。本文将不加区分地使用两者。

2.2 数据挖掘方法

数据挖掘方法研究旨在提供数据挖掘的方法论，制定实现知识发现目标的宏观策略，几种方法可以只解决一个数据挖掘问题。数据挖掘方法主要包括决策树方法、遗传算法、神经网络方法、贝叶斯网络方法、粗糙集方法、规则归纳方法、数据库方法和可视化方法等，另外还有模糊论方法和信息论方法等[3]。其实，随着数据挖掘研究的更加深入和应用的日益广泛，各种数据挖掘方法会相互融合，全新的数据挖掘方法也会出现。因此，和其它许多领域一样，数据挖掘方法无法穷尽。

3 网格技术的特点及意义

3.1 网格的基本概念

设 $A_1,A_2,\cdots,A_i$ 是数据集 $O=\{O_1,O_2,\cdots,O_n\}$ 中数据对象的 r 个属性的有界定义域，那么 $W=A_1\times A_2\times\cdots\times A_i$ 就是一个 r 维空间，将 $A_1,A_2,\cdots,A_i$ 看成是 W 的维（属性、字段），则对于一个包含 n 个数据点的 r 维空间中的数据集 $O=\{O_1,O_2,\cdots,O_n\}$，其中 $O_i=\{O_{i1},O_{i2},\cdots,O_{ir}\}(v=1,2,\cdots,n)$，$O_i$ 的第 j 个分量 O_{ij} 将 W 分割成 r 个网格单元[3]。

基于网格的聚类算法首先要划分网格结构，按搜索子空间策略的不同，主要有基于由底向上网格划分方法的算法和基于自顶向下网格划分方法的算法。

3.2 网格的起源

智能网格（Grid）的兴起有两个先决因素，一是存储器的发展速度远远高于处理器的发展速度；二是网络的大规模普及和其带宽的飞速扩展。衡量技术进步快慢的一个实用标准是：速度或容量翻倍的平均周期，或价格减半的平均周期。对存储、网络和计算能力而言，周期分别是大约 12、9 和 18 个月。按每单位面积的比特数衡量的数据存储容量每年都在成倍增长，现在 1TB 的特大容量磁盘的成本已降到 1 千元以下。存储专家预计，这种趋势还将继续，因此他们正在规划 PB 规模的大容量外存储器，一些建立高分辨能力仿真系统的科学家们也在规划 PB 级的大容量外存储器。如此大的数据量将对我们目前的分析能力提出更高的要求。现在，微处理器性能的显著提高使普通的台式电脑或膝上型电脑变成了功能强大的计算引擎，但计算机能力的发展速度仍然落后于存储容量。因此，在单一地点整合大规模分析所需的计算资源变得越来越不现实。相对于计算机能力每 18 个月提高一倍的速度，网络性能提高的速度要快一倍。这一点改变了我们对协作本身和如何实施协作的认识。如果像所预料的那样，网络以这样的速度发展，那么通信将基本上是免费的。为充分利用丰富的带宽资源，必须设想通信密集型的新型工作方式。

3.3 网格技术的特点

智能网格是近年来兴起的一种前沿信息技术，是互联网信息技术发展的新趋势。它的思想来源于电力智能网格，目的是将计算能力和信息资源像电力网一样通过网络形式方便地传送到用户中。智能网格是高性能计算机、数据资源、因特网 3 种技术的有机组合和发展，它把分布在各地的各种计算机连接起来，进行资源共享。美国智能网格项目的负责人之一伊安·福斯特在他所主编的题为《智能网格：21 世纪信息技术基础设施的蓝图》一书中认为“智能网格就是构筑在互联网上的一组新兴技术。它将高速互联网、高性能计算机、大型数据库、传感器、远程设备等融为一体，为科技人员和普通用户提供更多的资源、功能和交互性[3]。互联网主要为人们提供电子邮件、网页浏览等通信功能，而智能网格的功能则更多和更强，能让人们透明地使用计算、存储等其他资源。”因此，智能网格是一个一致、开放、标准的计算环境的信息基础设施，支持聚合地理上广泛分布的高性能计算资源、大容量数据和信息存储资源、软件和应用系统、高速测试和获取系统以及

人力等各种资源的合作问题求解系统的构造。

智能网格的根本特征是资源共享。它把整个网络整合成一台巨大的超级虚拟计算机，实现各种资源的全面共享。目前因特网上各种信息资源由于分散在不同的地方，要进行资源共享十分困难，并且利用效率比较低。智能网格则可以实现互联网上所有资源包括硬软件资源、计算资源、存储资源、通信资源、信息资源、知识资源等的全面连通，通过智能网格系统进行利用，使网络信息资源能被充分利用，从而发挥网络信息资源的价值。

智能网格的特点可以从不同的角度进行描述。在规模上，它具有可扩展性。智能网格可以从最初的包含少数的资源发展到具有成千上万资源的大智能网格，可以从个人智能网格逐步扩展到学校智能网格、企业智能网格甚至是全球智能网格。在结构上，它具有异构性。构成智能网格计算系统的不同类型的超级计算机在体系结构、操作系统等多个层次上可能具有不同的结构。在性能上，它具有动态性。智能网格计算系统中的资源共享造成系统行动和系统性能经常变化。从管理的角度来看，智能网格具有多级管理域。由于构成智能网格计算系统的各个计算机资源可能属于不同的机构，因此，需要各个机构共同参与管理。根据计算角度和应用特性，智能网格又可分为计算智能网格、存储智能网格、数据智能网格、信息智能网格、知识智能网格等。

智能网格将带来一场互联网的革命。互联网的作用是将各种计算机连结起来，而智能网格是将各种信息资源连结起来。互联网实现了计算机硬件的连通，Web 实现了网页的连通，而智能网格试图实现互联网上所有资源的全面连通，包括计算资源、存储资源、通信资源、软件资源、信息资源、知识资源等。智能网格的应用将会遍及各个领域，从而给各行各业带来巨大的效益。正如 IBM 深度计算研究所所长比尔·普里布兰所说，智能网格和高性能计算机等信息技术的根本目的就是辅佐人类实现人与机器共生，从而解放人的大脑，提高社会的生产力。有人认为，美国 70 年代对因特网的研究导致了今天网络经济的繁荣，而现在对智能网格的研究可与当年的因特网研究相提并论，可以预料 10 年后的智能网格将如同今天的因特网一样，普及到国民经济和社会的各个领域，从而起到重大的作用。

智能网格将成为信息产业的新热点，从而带来许多机会和巨大的经济效益。据美国《福布斯》杂志的预测，智能网格技术将在 2005 年达到高峰。智能网格技术如果能按预期的 17％年增长率持续发展，那么，在 2020 年将会形成一个年产值 20 万亿美元的大产业，将对世界社会经济产生巨大的影响。

3.4 网格的划分方法

3.4.1 由底向上的划分方法

由底向上的网格划分方法按照用户输入的划分参数（即每段维数 k_i，$1<i<d$），将数据间均匀划分为相等大小的网格单元。假设落入同一网格单元内的所有数据点都属于同一网，每个网格单元保存落入其内数据的统计信息，比如数据点个数、数据点之和。包含一定数据点的网格单元被称为高密度网格单元。

Wave Cluster 与 CLIQUE 是采用由底向上网格划分方法的代表算法。低维空间数据的性能超越了 BIRCH、CLARANS 与 DBSCAN 等优秀的聚类算法。CLIQUE 考虑了高维子空间聚类，但它的时间复杂度较高，需要用户指定全局密度阈值 MAFIA 对 CLIQUE 进行改进。为了减少聚类算法需要处理的网格单元数目，MAFIA 均匀划分网格中每一维上数据分布密度相似的相邻段进行合并，由此得到一个不均匀划分的网格，这个网格在数据分布均匀的区域划分粒度大，在数据分布不均匀的区域划分粒度小，这种均匀划分网格的方法能够提高聚类的质量，被后续的许多算法所采用。

采用由底向上的网格划分方法的优点在于，它能通过对数据的一遍扫描，将数据压缩到一个网格数据结构内，那么得到的聚簇的精度较高，但是算法的计算复杂度较大。此外，由底向上的网格方法存在不适合处理高维数据的问题。在高维空间，数据的分布是非常稀疏的，网格方法失去其压缩作用，而且属于同一个簇的高密度网格单元也可能不相连，这使得聚类算法不能发现合理数目的簇。

3.4.2 自顶向下的划分方法

自顶向下的网格划分方法采取分治的策略（divide and conquer principle），对数据空间进行递归划分，使问题的规模不断减小。首先将原数据空间划分为几个较大的区域，对于每个得到的区域，划分过程反复执行，直到每个区域包含属于同一个簇的数据点，那么这些区域就是最终的网格单元。基于自顶向下网格方法的聚类算法直接将高密度网格单元识别为一个簇，或是将相连的高密度网格单元识别为簇。

OptiGrid 与 CLTree 是两个典型的基于自顶向下网格划分方法的聚类算法[4]。其中，OptiGrid 用空间数据分布的密度信息来选择最优划分。通过一个密度函数来决定切割平面，可以将数据空间划分为规则的或不规则单元，与传统的等间距的划分相比，可以用此来解决高维聚类的问题。而 CLTree 用划分后的信息增益来选取最优划分。

自顶向下划分方法的主要优点在于不需要用户指定划分参数，而是根据数据的分布对空间进行划分，因此这种划分更为合理。数据空间维度对自顶向下网格方法的影响较小，可以快速将大型高维数据集中的簇分隔开。由于划分是基于数据分布的，而通常认为噪音是在整个空间均匀分布的，因此自顶向下划分方法对噪音不敏感。但是，由于这种方法得到的网格单元的体积远大于由底向上网格方法中的网格单元体积，因此方法产生的簇的描述精度比由底向上的网格方法得到的簇的描述精度要低；而且在自顶向下的划分过程中，同一个簇可能被划分到不同的区域中，最终得到的同一个区域也可能包含不同的簇，这样就进一步降低了算法的正确度。这类划分方法的另一个缺点是它在划分过程中，需要对数据集进行多次扫描。

而由底向上划分方法只需对数据集进行一次线性扫描且具有较高的簇的描述精度。

因此，两类方法适用于不同的问题。前者适于处理高维数据集，后者能有效处理存取代价较大的超大型数据集与动态数据。

4 聚类算法分析

4.1 基于网格的聚类过程

基于网格的聚类算法的基本过程是，首先将数据空间 W

划分为网格单元，将数据对象集 O 映射到网格单元中，并计算每个单元的密度，根据用户输入的密度阈值 $MinPts$ 判断每个网格单元是否为高密度单元，由邻近的稠密单元组形成簇。

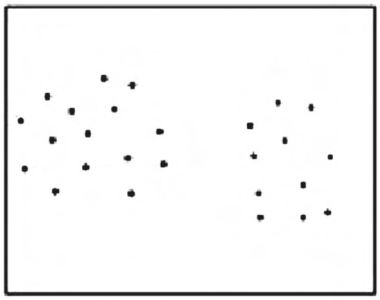
基于网格的聚类算法[5]主要包括 GRIDCLUS[1]，STING，Wave Cluster，CLIQUE，OptiGrid。其中，GRIDCLUS 是基于索引文件 GridFlie[2] 中的网格数据结构进行聚类分析；STING 主要用于回答查询；Wave Cluster 使用小波分析使簇的边界变得更加清晰；CLIQUE 进行了改进；OptiGrid使用数据在每维的投影的分布来划分网格空间。虽然这些算法使用了不同的网格划分方法，并对网格数据结构进行了不同的处理，但它们的核心步骤都是相同的，那就是，它们都使用网格单元内数据的统计信息对数据进行压缩表达，然后基于这些统计信息判断高密度网格单元，最后将相连的高密度网格单元识别为簇。

由此可见，网格方法本质上只能看作一种压缩手段，它必须与密度结合起来才能进行聚类分析。一些学者已经认识到这一点，直接指出他们的算法是同时基于网格方法和密度的。而基于密度的聚类算法方法也经常通过将空间网格化来降低计算量，避免在空间中每个位置都要计算密度。算法 DENCLUE 是一个基于密度的聚类算法，它使用核函数计算空间数据的分布密度。由于不可能计算空间每一点的密度，算法将空间网格化，只计算每个网格单元的密度。此时，网格方法的作用就是将连续空间离散化，以减少计算量。

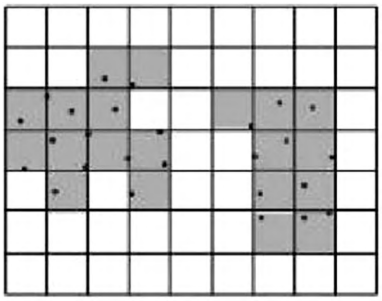
大部分基于网格的聚类算法都使用了两个关键参数：网格划分参数 k 和密度阈值 E。算法 STING、Wave Cluster、CLIQUE、MAFIA 都是通过将数据空间的每一维划分为 k 个段来产生一个均匀划分的网格[6]。其中，每一维的划分段数 k 可以相同，也可以不同。下面用一个简单的例子说明划分参数 k 的取值对聚类结果的影响。

图 1(a)给出一个包含两个簇的二维数据集，图 1(b)－(d)给出网格划分参数取不同值时的聚类结果。假设当网格单元中至少包含一个数据点时就认为它是一个高密度网格单元，图 1(b)－(d)中的阴影区域表示聚类算法得到的簇。从图中可以看到，当网格划分参数 k 取不同值时，算法得到的聚类结果相差很大。当划分参数过大时，网格单元的粒度太小，同一个簇内的高密度网格单元可能会不相连，导致算法生成多个小的簇；而当划分参数过小时，网格单元粒度太大，所有的高密度网格单元都连在一起，导致数据中存在的多个簇产生合并。网格划分参数除了对聚类结果的影响很大，它对聚类算法的计算复杂度也有很大影响。因此，选取合适的网格划分参数，对基于网格方法的聚类算法非常重要。但是，用户很难在处理数据前找到一个合适的划分参数值；而且，即使用户拥有关于数据分布的一些先验知识，这个参数也很难确定。目前设置有效划分参数的唯一方法是使用多个不同的划分参数分别对数据进行处理，然后比较处理结果，选取结果最好的。一个可行的方法是将划分参数从大到小逐步减小，即网格单元粒度从小到大逐步增加，直到获得满意的聚类结果。

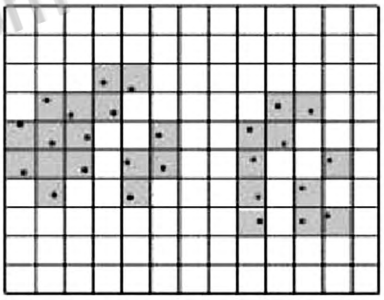
基于网格方法的聚类算法中的另一个参数 E 用于判断一个网格单元是否是高密度网格单元。当一个网格单元中的数据点数目超过 E 时，它被认为是一个高密度网格单元。参数 E 对聚类结果也有很大影响，但是它的取值可以通过分析数据分布的情况来设置。当数据中存在噪音时，将相连的高密度单元识别为簇的聚类算法可能对 E 的取值非常敏感，目前只有算法 Wave Cluster 考虑了这个问题，并通过小波变换消除数据中的噪音，使得算法能处理包含噪音的数据。



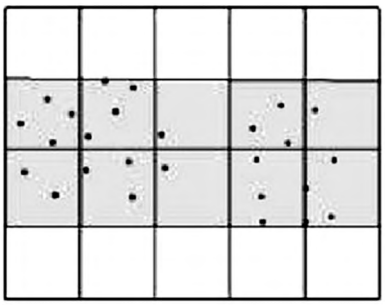
(a)包含两个簇的简单二维数据集



(b)网格划分参数合适时的聚类结果



(c)网格划分参数过大时的聚类结果



(d)网格划分参数过小时的聚类结果

图 1　不同划分参数对聚类结果的影响

4.1.1　网格单元的密度

簇就是一个区域，该区域中的点的密度大于与之相邻的区域。在网格数据结构中，由于每个网格单元都有相同的体积，因此网格单元中数据点的密度即是落到单元中的点的个数。据此可以得到稠密网格单元的密度，设在某一时刻 t 一个网格单元的密度为 $density$，定义 $density$＝单元内的数据点数/数据空间中的总的数据点数，设密度阈值为 p，为用户输入的密度阈值，当 $density > p$ 时，该网格单元是一个稠密网

格单元。

相对于稠密网格单元来说，大多数网格单元包含非常少甚至空的数据，这一类网格单元被称为稀疏网格单元。大量的稀疏网格单元的簇会极大地降低聚类的速度，需要在聚类之前对稀疏网格单元进行处理，定义稀疏密度阈值为 q，当 $density<q$ 时，该网格单元是一个稀疏单元，对于稀疏网格单元的处理一般采用压缩的方法或者直接删除的方法，如果需要保留稀疏网格单元用于后续处理，可以使用压缩的方法；如果在现有数据的基础之上直接聚类，可以删除稀疏网格单元，理论分析和实验证明删除稀疏网格单元并不影响聚类的质量[6]。

4.1.2 由稠密网格单元形成簇

在基于网格的聚类算法中，根据以上分析，由邻接的稠密单元形成簇是相对直截了当的，这也是基于网格的方法的优点之一。但是需要首先定义邻接单元的含义。设 n 维空间中存在任意两个网格单元 U_1 和 U_2，当这两个网格单元在同一个维上有交集或是具有一个公共面时，称它们为邻接网格单元。

在二维空间中，比较常使用的是 4-connection 相邻定义和 8-connection 相邻定义，4-connection 更适合在聚类算法中使用。因为当寻找某个网格单元的邻居时，在 4-connection 定义下，一个网格单元只有 $2d$ 个邻居，而在 8-connection 定义下，有 $3d-1$ 个邻居，当数据维度 d 较大时，这个数目非常大。使用 4-connection 不仅参与计算的单元数目大为减少，而且单元增加与维数的关系由指数增长变为线性增长，所以能进一步减少算法运行所需的时间，具有较低的计算复杂度。此外，只有在非常特殊的情况下，使用 4-connection 定义得到的聚类结果才会与使用 8-connection 定义得到的聚类结果不同，这是因为，当 4-connection 的网格单元是高密度网格单元时，4 个对角线上的网格单元不论是否是高密度网格单元，都能被正确地聚类；只有当与对角线上的网格单元相邻的 2 个网格单元同时为空且该单元本身是高密度网格单元时，才不能正确聚类，在划分网格时，通常都要求网格单元的大小远小于簇的大小，因此可以认为这种情况出现的可能性很小。

4.2 基于密度的聚类方法

基于密度的聚类方法将数据空间的高密度对象区域看成是簇，这一个个簇是被低密度区域分割开来的，这些簇所代表的区域就称为一个聚类。这类方法的代表算法有：DBSCAN、OPTICS、DENCLUE 等[7]。

· DBSCAN

DBSCAN(Density Based Spatial Clustering of Applications with Noise)是一个基于密度的聚类算法。该算法将具有足够高密度的区域划分为簇，并可以在带有“噪音”的空间数据库中发现任意形状的聚类。它定义簇为密度相连的点的最大集合。

为了能够理解基于密度的聚类算法，必须掌握一些有关密度的相关概念，例如：E-领域、核心对象、密度可达、密度相连等。给定对象半径 E 内的区域称为该对象的 E-领域。

如果一个对象的 E-领域至少包含有最小数目 $MinPts$ 个对象，则称该对象为核心对象。

给定一个对象集合 D，如果有 p 在 q 的 E-领域内，而 q 是一个核心对象，我们说对象 p 从对象 q 出发是直接密度粒大的。

如果存在一个对象链 $p_1, p_2, \cdots, p_n, p_1=q, p_n=q$，对 p_i $(1\leqslant i\leqslant n)$ 属于 D，p_{i+1} 是从 p_i 关于 E 和 $MinPts$ 直接密度可达的，则对象 p 是从对象 q 关于 E 和 $MinPts$ 密度可达的(density-reachable)。

如果对象集合 D 中存在一个对象 o，使得对象 p 和 q 是从 o 关于 E 和 $MinPts$ 密度可达的，那么对象 p 和 q 是关于 E 和 $MinPts$ 密度相连的(density-connected)。

密度可达是直接密度可达的传递闭包，这种关系是非对称的，只有核心对象之间是相互密度粒大的。然而，密度相连性是一个对称关系。

一个基于密度的簇是基于 idol 可达性的最大的密度相连对象的集合。不包含在任何簇中的对象被认为是“噪音”。

DBSCAN 通过检查数据库中每个点的 E-领域来寻找聚类。如果一个点 p 的 E-领域包含多余 $MinPts$ 个点，则创建一个以 p 作为核心对象的信簇。然后，DBSCAN 反复地寻找从这些核心对象密度可达的对象，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。

如果采用空间索引，DBSCAN 的计算复杂度是 $O(\log(n))$，n 是数据库中对象的数目；否则计算复杂度 $O(n^2)$。该算法对用户定义的参数是敏感的。

· OPTICS

尽管 DBSCAN 能根据给定输入参数 E 和 $MinPts$ 对对象进行聚类，但它仍然将选择能产生可接受的聚类结果的参数值的责任留给了用户。事实上，这也是许多其它聚类算法存在的问题。对于真实的高维数据集合而言，参数通常是依靠经验设置的，难以确定。绝大多数算法对参数值是非常敏感：设置的细微不同可能导致差别很大的聚类结果。而且，真实的高维数据集合经常分布不均，全局密度参数不能刻画其内在的聚类结构。

为了解决这个问题，OPTICS(Ordering Points to Identify the Clustering Structure)聚类分析方法应运而生。OPTICS 没有显式地产生一个数据集合簇，它为自动和交互的聚类分析计算一个簇次序(cluster ordering)，这个次序代表了数据的基于密度的聚类结构。它包含的信息等同于从一个宽广的参数设置范围所获得的基于密度的聚类。

考察 DBSCAN，可以看到，对一个恒定的 $MinPts$，关于高密度的(即较小的 E 值)聚类结果被完全包含在根据较低密度所获得的密度相连的集合中。参数 E 是距离，它是邻域的半径。因此，为了生成基于密度聚类的集合或次序，可以扩展 DBSCAN 算法来同时处理一组距离参数值。为了同时构建不同的聚类，对象应当以特定的顺序来处理。这个次序选择根据最小的 E 值密度可达的对象，以便高密度的聚类能被首先完成，基于这个思想，每个对象需要存储两个值——核心距离(core distance)和可达距离(reachable distance)。

一个对象 p 的核心距离是使得 p 成为核心对象的最小 E。如果 p 不是核心对象，p 的核心距离没有定义。

一个对象 q 关于另一个对象 p 的可达距离是 p 的核心距离和 p 与 q 的距离两者中的较大值。如果 p 不是核心对象，

p 和 q 之间的可达距离没有定义。

OPTICS 算法创建了数据库中对象的一个次序，额外存储了每个对象的核心距离和一个适当的可达距离。已经出现了一种算法，基于 OPTICS 产生的次序信息来抽取聚类。对于小于该次序时采用的距离 E 以及为抽取所有基于密度的聚类，这些信息是足够的。

由于与 DBSCAN 在结构上的等价性，OPTICS 算法具有和 DBSCAN 相同的时间复杂度。

· DENCLUE

DENCLUE(Density-based Clustering)是一个基于一组密度分布函数的聚类算法。该算法主要基于下面的想法：(i)每个数据点的影响可以用一个数学函数来形式化地模拟，它描述了一个数据点在领域内的影响，被称为影响函数(influence function)；(ii)数据空间的整体都可以被模型化为所有数据点的影响函数的综合；(iii)聚类可以通过确定密度吸引点(density attractor)来得到，这里的密度吸引点是全局密度函数的局部最大。

假设 x 和 y 是 d 维特征空间 Fd 中的对象。数据对象 y 对 x 的影响函数是一个函数，它是根据一个基本的影响函数定义的。原则上，影响函数可以是一个任意的函数，它由某个领域内的两个对象之间的距离来决定。距离函数 $d(x,y)$ 应当是自反的和对称的，例如欧几里得距离。它用来计算一个方波影响函数(square wave influence function)或者一个高斯影响函数。

根据密度函数，我们能够定义该函数的梯度和密度吸引点(全局密度函数的局部最大)。一个点 x 是被一个密度吸引点 x 密度吸引的，如果存在一组点 $x_0, x_1, \cdots, x_k, x_0 = x, x_k = x$，对 $0<i<k, x_{i-1}$ 的梯度是在 x_i 的方向上对一个连续的和可微的影响函数，一个用梯度指导的爬山算法能用来计算一组数据点的密度吸引点。基于这些概念，能够形式化地定义中心定义的簇(center-defined cluster)和任意形状的簇(arbitrary-shape cluster)。密度吸引点 x 的中心的簇是一个被 x 密度吸引的子集 C，在 x 的密度函数不小于一个阈值 a 时；否则，它被认为是孤立点。一个任意形状的簇是子集 C 的集合，每一个簇密度吸引，又不小于阈值 E 密度函数值，并从每个区域到另一个都存在一条路径 P，该路径上的每个点的密度函数值都不小于 a。

DENCLUE 的优点有：(i)它有一个坚实的数学基础，概括了其它的聚类方法，包括基于划分的、层次的及基于位置的方法；(ii)对于大量“噪音”的数据集合，它有良好的聚类特性；(iii)对高维数据任意形状的聚类，它给出了简洁的数学描述；(iv)它使用了网格单元，只保存关于实际包含数据点的网格单元的信息，它以一个基于树的存取结构来管理这些单元，因此比一些有影响的算法(如 DBSCAN)速度要快。但是这个方法要求对密度参数 q 和“噪音”阈值 E 进行仔细的选择，因为这些参数的选择可能显著地影响聚类结果的质量。

4.3 基于网格的聚类方法

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。

基于网格的方法包括：STING，它利用存储在网格单元中的统计信息；Wave Cluster，它利用一种小波转换方法来聚类；CLIQUE，它是在高维数据空间基于网格和密度的聚类方法；SCI 等。

4.4 传统的网格聚类算法

(1) STING

STING(Statistical Information Grid)[8] 是一种基于网格的多分辨率聚类技术，它将空间区域划分为矩型单元。针对不同级别的分辨率，通常存在多个级别的矩形单元，这些单元形成了一个层次结构；高层的每个单元被划分为多个低一层的单元。每个网格单元属性的统计信息(例如平均值、最大值和最小值)被预先计算和存储。这些统计信息对于下面描述的查询处理是有用的。

高层单元的统计参数可以很容易地从低层单元的计算得到。这些统计参数包括：属性无关的参数 $count$；属性相关的参数 m(平均值)，s(标准偏差)，min(最小值)，max(最大值)，以及该单元中属性值遵循的分布(distribution)类型，例如正太分布、一致分布、指数分布或无(如果分布未知)。当数据被装载进数据库时，底层单元的参数 $count, m, s, min$ 和 max 直接进行计算。如果分布的类型事先知道，distribution 的值可以由用户指定，也可以通过假设检验(如 x2 检验)来获得。一个高层单元的分布类型可以给予它对应的低层单元多数的分布类型，用一个阈值过滤过程来计算。如果低层单元的分布彼此不同，阈值检验失败，高层单元的分布类型被置为 none。

统计参数可以按照以下自顶向下的基于网格的方法使用。首先，在层次结构中选定一层作为查询处理的开始点。通常，该层包含少量的单元。对当前层次的每个单元，计算置信度区间(或者估算其概率范围)，用以反映该单元与给定查询的关联程度。不相关的单元不再考虑。低一层的处理就只检查剩余的相关单元。这个处理过程反复进行，直到达到底层。此时，如果查询要求被满足，那么返回相关单元的区域；否则，检索和进一步地处理落在相关单元中的数据，直到它们满足查询要求。

STING 有几个优点：(1)由于存储在每个单元中的统计信息提供了单元中的数据不依赖于查询的汇总信息，因此基于网格的计算是独立于查询的。(2)网格结构有利于并行处理和增量更新。(3)效率很高。STING 扫描数据库一次来计算单元的统计信息，因此产生聚类的时间复杂度是 $O(n)$，n 是对象的数目。在层次结构建立后，查询处理时间是 $O(g)$，这里 g 是最底层网格单元的数目，通常远远小于 n。

由于采用了一个多分辨率的方法来进行聚类分析，STING 聚类的质量取决于网格结构的最底层的粒度。如果粒度比较细，处理的代价会显著增加；但是，如果网格结构的最底层的粒度太粗，将会降低聚类分析的质量。而且，STING 在构建一个父亲单元时，没有考虑孩子单元和其相邻单元的关系。因此，结果簇的边界要么是水平的，要么是竖直的，没有对角的边界。该技术尽管有较快的处理速度，但可能会降低簇的质量和精确性。

(2) Wave Cluster

Wave Cluster 是一种多分辨率的聚类算法，它首先通过

在数据空间上强加一个多维网格结构来汇总数据，然后采用一种小波变换来变换原特征空间，在变换后的空间中找到密集区域。在该方法中，每个网格单元汇总了一组映射到该单元中的点的信息。这种汇总信息适合于在内存中进行多分辨率小波变换时以及随后的聚类分析使用。

小波变换是一种信号处理技术，它将一个信号分解为不同频率的子波段。通过应用一维小波变换 n 次，小波模型可以应用于 n 维信号。在进行小波变换时，数据被变换在不同的分辨率层次保留对象间的相对距离。这使得数据的自然聚类变得更加容易区别。通过在新的空间中寻找高密度区域，可以确定聚类。

小波变换对聚类有如下优点：

提供了无指导的聚类。它采用了帽形(hat-shape)过滤，强调点密集的区域，而忽视了在密集区域外的较弱的信息。这样，在原特征空间中的密集区域成为了附近点的吸引点(attractor)，距离较远的点成为抑制点(inhibitor)。这意味着数据的聚类自动地显示出来，并"清理"了周围的区域。这样，小波变换的另一个优点是能够自动地排除孤立点。小波变换的多分辨率特性对不同精确性层次的聚类探测是有帮助的。

基于小波变换的聚类速度很快，计算复杂度为 $O(n)$，这里 n 是数据库中对象的数目。这个算法事先可以并行化。

(3)CLIQUE

CLIQUE(Clustering in Quest)聚类算法综合了基于网格和基于密度的聚类方法。它对大规模数据库中的高维数据的聚类非常有效。CLIQUE 的中心思想如下：

给定一个多维数据点的大集合，数据点在数据空间中通常不是均衡分布的。CLIQUE 区分空间中稀疏的和"拥挤的"区域(或单元)，以发现数据集合的全局分布模式。

如果一个单元中的数据点的数目超过了某个输入模型参数，则该单元是密集的。在 CLIQUE 中，簇定义为相连的密集单元的最大集合。

CLIQUE 分两步进行多维聚类：

(1)CLIQUE 将 n 维数据空间划分为互不相交的长方形单元，识别其中的密集单元。该工作对每一维进行。代表密集单元的相交子空间形成了一个候选搜索空间，其中可能存在更高维的密集单元。

CLIQUE 将更高维的密集单元的搜索限制在子空间的密集单元的交集中。这种方法来自于关联规则挖掘中的先验性质(apriori property)。一般来说，该性质在搜索空间中利用数据项的先验知识，以便裁减空间。CLIQUE 所采用的性质如下：如果一个 k 维单元是密集的，那么它在 $k-1$ 维空间上的投影也是密集的。也就是说，给定一个 k 维的候选密集单元，检查它的 $k-1$ 维投影单元，只要有一个是非密集的，那么 k 维单元也不可能是密集的。因此，可以从 $k-1$ 维空间中的密集单元来推断 k 维空间中潜在的(或候选的)密集单元。通常，最终的结果空间比初始空间要小很多，然后检查密集单元决定聚类。

(2)CLIQUE 为每个簇生成最小化的描述。对每个簇，它确定覆盖相连的密集单元的最大区域，然后确定最小的覆盖。

CLIQUE 自动地发现最高维的子空间，高密度聚类存在于这些子空间中。CLIQUE 对元组的输入顺序不敏感，无需架设任何规范的数据分布。它随着输入数据的大小线性地扩展，当数据的维数增加时，具有良好的可伸缩性。但是，由于方法大大简化，聚类结果的精确性可能会降低。

4.5 一些改进的网格聚类算法

(1)SCI

SCI(Subspace Clustering based on Information)[9]聚类算法综合了基于密度和基于网格的聚类方法。网格的划分方法和 CLIQUE 类似，通过对 d 维数据集 D 的每个属性上等分得到，首先将各个属性排序为 $[L_j, U_j]$，$j=1,2,3,4,\cdots,d$，然后通过 k-regular 划分成彼邻的矩形单元格。数据空间被划分为 k 个相同体积的单元格。所以说网格是均匀划分的。

在聚类子空间中，它通过连接稠密单元格的技术获得簇的大体轮廓。落入每个单元格中的数据点的总数就看作该单元格的密度。把单元格分成 3 种类型，即稠密单元格、稀疏单元格和孤立单元格。先通过熵的定理去除某些对于聚类效果信息少的属性，然后稠密单元格彼此相连，被稀疏单元格分离，形成簇的轮廓。而孤立单元格也被稀疏单元格分离，被看作孤立点集，而稀疏单元格中的点可能是簇的边界点，也可能是噪音点，需要进一步处理。处理的方法是，对于每一个在稀疏单元格中的数据点，如果离其最近的单元格是是稠密单元格，则将其归为簇中；否则就是噪音数据。最后形成簇。

聚类的基本思想是分为 4 个阶段。

1. 划分数据空间，单元格的边长 k 是输入参数；

2. 计算在每个单元格中的点数，然后标识每个单元格的类型；

3. 利用熵的定理去除某些属性 X_j：如果 $Info(X_j) \geqslant \log 2k - I^*$ 或 $Info(X_j) \leqslant I^*$，其中 I^* 是输入参数，则去除 X_j；

4. 检测孤立点和簇的边界点，然后连接形成簇。

该算法适合高维的数据集，主要是针对 CLIQUE、STING 中有可能将簇边界的点当成噪音数据去除的情况做出改进的，并且提出属性提取的方法来减少聚类的时间。

算法是时间复杂度为 $O(n)$，n 是数据点的数目；空间复杂度为 $O(k*d)$，其中 k 是在每一维上的单元格边长，d 是维度。

该算法的优点是，对高维属性的伸缩性、对噪音数据的健壮性、对输入数据的顺序不敏感，以及能够在线性时间内发现任意形状的簇；缺点是由于采用熵的定义，输入参数变得十分敏感，信息有可能丢失，同时空间复杂度很大。

(2)MAFIA

MAFIA 聚类算法综合了基于密度和基于网格的聚类算法。网格划分方法是根据数据分布决定网格单元的大小，因此网格的划分是不均匀的。

在 MAFIA 算法中使用了一种自底向上的子空间聚类技术。该算法基本思想可以概况如下：根据数据分布划分网格到单元，k 维候选的高密度单元是通过合并任意两个 $(k-1)$ 维的高密度单元得到的，并且这两个 $(k-1)$ 维的单元有一个共同的 $(k-2)$ 维的子单元，再根据高密度单元进行聚类。

该算法适合高维和大数据集，其时间复杂度是随维数呈指数增长。该算法的优点是不需要用户去输入一般的网格参数；缺点是对参数相当敏感，运行时间随维数呈指数增长。

通过与 CLIQUE 进行比较，得出 MAFIA 性能较好并且有较好的聚类质量，是 CLIQUE 聚类的一种提高。

(3)ENCLUS

ENCLUS 聚类算法是一种基于网格的聚类方法。网格的划分方法是等分数据空间的每一维，所以网格的划分是均匀的。

在 ENCLUS 中采用了一种寻找聚类子空间的技术：根据指定熵的值，由底向上(从一维开始)寻找有效子空间。该算法的基本思想可概括如下：在 CLIQUE 算法提出的搜索有效的子空间技术的基础上，提出一种基于熵的搜索有效子空间的方法，对每一个子空间计算其熵值，若值低于指定的熵值，就认为此单元是有效的，在找出的有效的子空间中，使用现有的聚类算法都可以进行聚类。

该算法的时间和空间复杂度都是线性的，类似于 CLIQUE 算法。ENCLUS 算法的优点是提出了一种有效的基于熵的搜索子空间的标准，效率高；缺点是对参数非常敏感。

(4)DCLUST

DCLUST 聚类算法综合了基于密度和基于网格的聚类方法。网格的划分是等分数据空间的每一维，所以网格的划分是均匀的。

DCLUST 算法的基本思想可概括如下：首先划分网格，根据密度阈值获得高密度单元，将每个高密度单元的中心作为其代表点，根据这些代表点构造带标点的最小生成树(R-MST)和概要结构(summary structure)，利用 R-MST 进行 Multi-resolution 聚类和增量(incremental)聚类。

该算法的时间和空间复杂度均为 $O(n)$(其中 n 为数据点的数目)，其优点是能处理含噪声的任意形状的簇，并且对数据的顺序不敏感，可以处理增量聚类。DCLUST 算法主要解决传统的空间聚类算法不能有效地处理增量聚类的问题。

(5)MMNG

MMNG 聚类算法是一种基于网格的聚类方法。网格的划分方法是利用一种 P-树的数据结构进行划分，网格的划分是均匀的。

MMNG 算法的基本思想可概括如下：使用了一个 P-树的数据结构来划分数据集，并计算每一个划分单元的中心点，以此进行聚类，从而达到对 MM 算法的一种改进。

算法的优点是当数据维数增加时，MMNG 需评估的簇中心的数目相比 MM 算法呈指数下降。该算法主要是对 MM 算法的一种改进。

(6)GCHL

GCHL 聚类算法是一种综合了基于密度和基于网格的聚类算法。网格的划分方法是利用分割平面来划分数据空间，所以网格的划分不一定均匀。

GCHL 算法的基本思想可概括如下：对输入的数据空间，结合轴平行划分策略(分割平面)和基于密度-网格的方法识别高密度区域进行聚类。

该算法的复杂度是位于 $O(n \cdot p \cdot d)$和 $O(p \cdot d \cdot n \cdot \log n)$之间，其中 n 是数据集个数，d 是维数，p 是划分的块数。

该算法的优点是只扫描一遍数据集；能发现任意形状的簇；效率高并且对数据的顺序不敏感；能处理高维聚类。

(7)GDILC

GDILC 聚类算法是一种基于网格的聚类方法。网格的划分方法是等分数据空间的每一维，所以网格的划分是均匀的。

GDILC 算法的基本思想可概括如下：描述了一个基于网格的等高线聚类，即同一类中的点在同一个等高线上，相邻等高线的距离若小于一个阈值，则合并这两个等高线对应的类。GDILC 算法的时间复杂度是线性的，该算法的优点是能快速、无指导地聚类，并能很好地识别出孤立点和各种形状的簇；缺点是不能很好地分离出各个类。

(8)OptiGrid

OptiGrid 聚类算法是一种基于网格的聚类方法。在 OptiGrid 算法中，网格的划分方法是根据密度函数来选择切割平面进行划分，所以网格的划分不均匀。

OptiGrid 算法的基本思想可概括如下：通过一个密度函数来决定切割平面，可以将空间划分为规则的或不规则的单元，与传统的等间距的划分有所不同，用来解决高维聚类。OptiGrid 算法的时间复杂度是位于 $O(n \times d)$和 $O(d \times n \times \log n)$之间的。该算法的优点是有一个坚固的数学理论作基础，且算法的效率很高，解决了常见算法中的“维”危机。

(9)IGDCA

IGDCA 聚类算法是一种基于网格的聚类方法。在 IGDCA 算法中，网格的划分方法是等分数据空间的每一维，所以网格的划分是均匀的。

IGDCA 算法的基本思想可概括如下：首先将数据点映射到每个单元中并计算单元密度，找出每个密集单元的密度可达和密度相关的单元，将这些单元中的点映射到每个类中。IGDCA 算法的时间复杂度为 $O(n+|C_d|\log|C_{ne}|)$，其中 C_{ne} 是非空单元，C_d 是密集单元集合。该算法的优点是能处理任意形状的簇，以考虑单元代替原来的数据点，提高了 DBSCAN 的效率。

(10)网格化聚类算法的均值近似方法

网格化聚类算法是一种基于网格的聚类方法。该方法的基本思想可概括为：采用数据空间网格划分的基于密度的聚类算法的均值近似方法，对密集单元，通过一个重心点取代原有的保存网格中所有点，有效减少了内存需求；采用一个近似的密度计算来减小密度计算的复杂度。这种算法的优点是通过采用均值计算方法可减少内存需求，大幅度降低计算复杂度。该算法是对目前基于网格和密度的聚类方法的一种改进。

(11)RD-Quadtree

RD-Quadtree 聚类算法是一种综合了基于密度和基于网格的聚类方法。在 RD-Quadtree 算法中，网格的划分是利用了一种 RD-树的数据结果，网格的划分是均匀的。

RD-Quadtree 算法的基本思想是通过引进数据结构 RD-树来划分数个空间为高密或低密空间单元，然后构造 RD-Quadtree 算法，通过合并 RD-Quadtree 中相连的黑色节点得到簇，不相连的黑色节点在不同的簇中。该算法改进了传统算法在聚类过程中对所有的网格单元进行处理的做法，只考虑非空的高密度单元。由于只考虑非空的高密度单元，该算法的时间复杂度大幅度降低。

(12)CBCM

CBCM 聚类算法是一种基于网格的聚类方法。在 CBCM 算法中,网格的划分是利用了一种基于密度的分离索引的数据结构,网格的划分是均匀的。

CBCM 算法的基本思想可概括为:利用一种基于密度的分离索引来划分数据空间的每一维,将密度大于指定的单元密度阈值的单元连接起来形成簇。算法主要解决了传统算法在高位聚类方面的"维"危机和内存不足问题,其优点是算法的速度很快,精度很高。

(13)移动网格聚类算法

移动网格聚类算法是一种综合了基于密度和基于网格的聚类方法。在移动网格聚类算法中,网格的划分方法是等分数据空间的每一维,所以网格的划分是均匀的。该算法的基本思想可概括为:在传统的网格聚类的基础上,使用滑动窗口技术即把每一个网格向外扩展半个网格单元,以提高聚类的精度。该算法的优点是不需要用户输入参数,有较高的精度;缺点是时间复杂度很大。

4.6 现有多密度聚类算法分析

聚类分析所使用的数据集中,各个类的密度往往不尽相同,甚至差别很大。大多数现有的聚类算法都是致力于如何发现任意形状和大小的类,但很难有效地处理密度差别较大的数据集。能够处理多密度数据集的聚类算法有 Chameleon、共享近邻 SNN 算法、多阶段等密度线算法等。

(1)Chameleon

Chameleon[12] 是一个在层次聚类中采用动态模型的聚类算法。在它的聚类过程中,如果两个簇间的互连性和近似度与簇内部对象间的互连性和近似度高度相关,则合并这两个簇。基于动态模型的合并过程有利于自然的和同构的聚类的发现,而且只要定义了相似度函数,就可以用于所有类型的数据。

Chameleon 首先通过一个图划分算法将数据对象聚类为大量相对较小的子聚类,然后用一个凝聚的层次聚类算法通过反复地合并子类,来找到真正的结果簇。它既考虑了互连性,又考虑了簇间的近似度,特别是簇内部的特征,来确定最相似的子簇。这样,它不依赖于一个静态的用户提供的模型,能够自动地适应被合并的簇的内部特征。

Chameleon 通常采用 k-最近邻居方法来描述它的对象。k-最近邻居图中的点代表一个数据对象,如果一个对象是另一个对象的 k-最类似的对象之一,那么在这两个顶点(对象)之间存在一条边。k-最近邻居图 Gk 动态地捕捉邻域的概念:一个对象的邻域半径由对象所在区域的密度决定。在一个密集区域,邻域的定义范围相对狭窄;在一个稀疏区域,它具有更自然的聚类结果。而且,区域的密度作为边的权重被记录下来。就是说,一个密集区域的边趋向于比稀疏区域的边有更大的权重。

Chameleon 通过两个簇的相对互连性 $RI(C_i,C_j)$ 和相对近似性 $RC(C_i,C_j)$ 来决定簇间的相似度。

两个簇 C_i 和 C_j 之间的相对互连性 $RI(C_i,C_j)$ 定义为 C_i 和 C_j 之间的绝对互连关于两个簇内部互连性的规范化。而两个簇 C_i 和 C_j 之间的相对近似性 $RC(C_i,C_j)$ 定义为 C_i 和 C_j 之间的绝对近似关于两个簇内部近似性的规范化。

可以看出,与 CURE 和 DBSCAN 相比,Chameleon 在发现高质量的任意形状的聚类方面有更强的能力。但是,在最坏的情况下,高维数据的处理代价是可能对 n 个对象需要 $O(n^2)$ 的时间。

(2)共享近邻 SNN 算法

共享近邻 SNN 算法[13] 是一个用于处理多密度数据集的聚类算法,其主要思想可概括为:对于数据集中每个点,找出距离其最近的 k 个邻近点,形成一个集合;然后考虑数据集中的任意两个点,若对应于这两个点的 k 个邻近点交集部分的点数超过一个阈值,则将这两个点归于一类。SNN 的具体步骤如下:

1. 计算数据集中每一个数据点在半径 r 的范围内共享的点数。

2. 对每一个数据点,保留共享点数最多的 k 个点,构造共享近邻图。

3. 如果两个数据点的共享点数超过一个阈值,则这两个点是相近的;每个数据点的密度即是其相近点的个数。

4. 一个点的密度大于一个阈值时,则是中心点。

5. 若两个中心点的聚类小于 Eps,则将这两个中心点归于一类。

6. 对每一个非中心点的数据点,将其归到离其最近的中心点所代表的类内。

SNN 算法的优点是可以对不同密度和形状的数据集进行聚类,能处理高维数据集和自动识别簇的数目;缺点是在多密度聚类和处理孤立点或噪声方面 SNN 算法精度都不高,并且该算法对参数是敏感的。

(3)多阶段等密度线算法

多阶段等密度线算法也可以用来处理多密度的数据集,其主要思想可概括为:首先计算数据集中每个点的密度,然后考虑每一对数据点,如果它们的密度超过了一个密度阈值,并且二者的距离小于一个距离阈值,则这两个数据点对应的簇被合并为一个类。具体步骤如下:

1. 划分数据空间的每一维为 m 等份,形成 m_d 个网格单元。

2. 计算距离阈值 RT。

3. 计算密度向量和密度阈值 DT。

4. 考虑密度值超过 DT 的每一个数据点,检查其相邻单元到它的距离,距离值若小于 RT,则合并这两个点所在的簇。

5. 去除噪声数据。

多阶段等密度线算法的时间复杂度为 $O(n^2)$,其中 n 是数据点数。该算法的缺点是不能很好地分离出各个类。

4.7 本章小结

本章对数据挖掘技术、基于密度的聚类方法和网格聚类方法进行了详细的阐述。

首先概述了数据挖掘,其中包括数据挖掘的概念、数据挖掘的过程、数据挖掘的功能以及数据挖掘的方法。

然后介绍了基于密度的聚类方法,其中典型的方法有 DBSCAN、OPTICS、DENCLUE 等,并对这些算法进行了分析。

(下转第 524 页)

关联矩阵表达形式，通过数学方法证明通过白盒建模细化得到的软件演化过程模型自身就具备着非常优良的结构性质，无须再对其进行结构验证。

下一步的工作是对通过黑盒建模方法得到的软件演化过程模型进行研究，并在本文的数学基础上，更进一步地提升，从而能够实现对其模型的结构性质证明。

参 考 文 献

[1] 杨芙清. 软件工程技术发展思索[J]. 软件学报，2005，16(1)：1-7

[2] Lehman M M，Peryy D E，Ramil J F. Metrics and Laws of Software Evolution-the Nineties Views[C]// Proceeding of 4th International Symposium on Software Metrics. IEEE Computer Society Press，2000：20-32

[3] Li T. An Approach to Modelling Software Evolution Processes [M]. Springer-Verylage Berlin and Heidelberg GmbH & Co. K，2008

[4] Brauer W，Gold R，Vogler W. A survey of behavior and equivalence preserving refinement of Petri nets [J]. Lecture Notes in Computer Science. 1990，483：1-46

[5] Suzuki I，Murata T. A method for stepwise refinement and abstraction of Petri nets [J]. J. Comput. System Sci.，1983，27：51-76

[6] Valette R. Analysis of Petri nets by stepwise refinements [J]. J. Comput. System Sci.，1979，18：35-46

[7] Betous-Almeida C，Kanoun K. Construction and stepwise refinement of dependability models [J]. Performance Evaluation，2004，56：277-306

[8] Nketsa A，Valette R. Rapid and modular prototyping-based Petri nets and distributed simulation for manufacturing systems [J]. Applid Mathematics and Computation，2001，120：265-278

[9] van Hee K，Sidorova N，et al. Soundness and separability of workflow nets in the stepwise refinement approach [C]// Proc the 24th International Conference on Application and Theory of Petri Nets. Eindhoven，The Netherlands，2003，2679：337-356

[10] 吴哲辉. Petri 网导论[M]. 机械工业出版社，2006

(上接第 499 页)

最后介绍了常见的网格聚类算法，其中典型的算法有 STING、Wave Cluster、CLIQUE 等，并对这些算法进行了分析。在此基础上介绍了常见的处理多密度数据集的算法，其中包括 Chameleon、共享近邻 SNN 算法、多阶段等密度算法等，并对这些算法进行了分析。

结束语　基于网格聚类方法的优点是：它的处理速度快，因为其速度与数据对象的个数无关，而只依赖于数据空间中每个维的单元的个数；能发现任意形状和大小的簇；计算结果与数据输入顺序无关；计算时间与数据量无关；同时不要求向 k 均值意义预先指定簇个数等。但是，基于网格方法的聚类算法的输入参数对聚类结果影响较大，而且这些参数较难设置；当数据中有噪音时，如果不加特殊处理，算法的聚类质量会很差；而且，算法对于数据维度的可伸缩性较差。

基于网格的聚类方法目前还存在一些亟需解决的问题，主要有以下几点：(1)当簇具有不同的密度时，全局的密度参数不能有效发现这样的簇，需要开发具有可变密度参数的算法；(2)对于不同类型数据的聚类问题，比如对于高维数据，网格的数据将急剧增加，需要有效的技术发现近邻单元；(3)当数据集的规模巨大以及数据具有地理分布特性时，需要开发有效的并行技术算法来提高处理的速度；(4)对现有网格算法的优化，从不同方面提高网格算法的有效性，比如开发稀疏网格的压缩算法和密度相似网格的合并算法等。

本文对基于网格的聚类方法的已有研究进行了分析和总结，包括网格的定义与划分方法、网格单元密度的确定、由邻接网格单元形成聚簇的聚类过程；最后对网格聚类方法与局限性进行总结，在已有研究分析的基础上，提出后续需要重点解决的问题。

参 考 文 献

[1] 马刚，李志刚. 数据仓库与数据挖掘的原理及应用[M]. 北京：高等教育出版社，2012：20-42

[2] 陈志泊. 数据仓库与数据挖掘[M]. 北京：清华大学出版社，2011：8-37

[3] Tan Pang-ning，Steinbach M，Kumar V. 数据挖掘导论[M]. 范明，译. 北京：人民邮电出版社，2013：6-53

[4] Dunham M H. DATA MINING Introductory and Advanced Topics [M]. 北京：清华大学出版社，2010：23-60

[5] Ng R T，Han J. Efficient and effective clustering methods for spatial data mining[C]// Proc of the 20th VLDB Conference. Chile，Santia，2010：144-155

[6] Spivak G. Victory in Limbo：Imagism [C]. Nelson C，Grasberg L，eds. Urbana：University of Illinois Press，2010：271-313

[7] Zhang T，Rrmakrishnan R，Livny M. An efficient data clustering method for very large databases[C]// Proc of ACM SIGMOD International Conference on Management of Data. New York：ACM Press，2012：103-114

[8] Tan Pang-ning，Steinbach M. Introduction to Data Mining[M]. 2010：372-373

[9] Chen Y，Tu L. Density-Based Clustering for Real-Time Stream Data[C]// Proceedings of the 13th ACM SIGKDD International Conference of Knowledge Discovery and Data Mining. San Jose，California，USA，2009：133-142

[10] 曹洪其，余岚，孙志辉. 基于网格聚类技术的离群点挖掘算法[J]. 计算机工程，2006(11)：18-96

[11] 孙玉芬. 基于网格方法的聚类算法研究[D]. 武汉：华中科技大学，2011

[12] Han J，Kamber M. Data Mining：Concepts and Techniques [J]. Morgan Kaufmann Publishers，2011，2(9)：33-82

[13] Chen Ming-yan，Han Jia-wei，Philip S Y. Data mining：an overview from a database perspective [J]. IEEE Trans on Knowledge and Data Eng.，1996，8(6)：806-833

阅读此文的还阅读了：

1. 聚类算法的若干问题研究
2. 一种基于划分的聚类算法分析与改进
3. 数据挖掘中聚类分析技术方法的比较研究
4. 基于网格技术的高精度聚类算法
5. 基于衰减滑动窗口数据流聚类算法研究
6. 基于土地规划的空间聚类算法
7. k-means聚类问题的改进近似算法
8. 数据挖掘中聚类方法比较研究
9. 基于相交划分的动态网格聚类算法
10. 交互式多模型粒子滤波算法综述
11. 一种基于聚类的空间数据知识获取算法
12. 精准农业中管理区划分方法研究
13. 用于邮件自动分类的聚类剪辑算法设计
14. 混合流水线调度研究进展
15. 中文智能答疑系统相关技术的研究与实现
16. 一种基于人工免疫系统的聚类算法

17. 关联规则挖掘算法综述

18. 基于测度的网格聚类算法

19. 一种基于划分的聚类算法分析与改进

20. k-means聚类问题的改进近似算法

21. 离散粒子群优化算法研究综述

22. 随机需求库存-路径问题:研究现状及展望

23. 判别分析方法在医学应用中的进展

24. 计算机辅助排样算法综述

25. 基于距离的K-Means划分式聚类算法及其编程实现

26. 部分可观测马尔可夫决策过程算法综述

27. 离散设施选址问题研究综述

28. K-means算法的初始点优化研究

29. 基于二值属性的聚类分析算法

30. 基于密度偏差抽样的聚类算法研究

31. 异常入侵检测中聚类算法的改进与应用

32. 数据挖掘中聚类方法比较研究

33. 车辆路径规划问题及其求解方法研究进展

34. 现代信息数据的挖掘与发展

35. 数据挖掘分类算法研究综述

36. 基于模糊聚类分析的航天发射故障诊断技术

37. 基于网格距离的高精度聚类算法

38. 基于K-Means和EM算法的聚类分析

39. 基于对象“形状”的聚类算法

40. 改进的模糊BP神经网络及在犯罪预测中的应用

41. 同杆平行双回线的故障测距综述

42. 一种综合多种聚类结果的算法

43. 车辆路径规划问题及其求解方法研究进展

44. 空间聚类算法在数字化城市管理中的应用

45. 数据挖掘中聚类算法研究综述

46. 采用蚁群爬山法进行聚类分析的算法

47. 一种改进的凝聚层次聚类法

48. 基于小波聚类的数据集简化算法研究

49. 关于常用聚类算法的实现与分析

50. 基于对象"形状"的聚类算法