

# 基于混合网格划分的子空间高维数据聚类算法

许倡森

(华南师范大学 计算机学院, 广东 广州 510631)

**摘要:**提出一种基于混合网格划分的子空间高维数据聚类算法。该算法消除了各个属性分量数值范围大小对计算的影响;有效去除冗余属性以提高聚类准确性与降低时间复杂度。根据数据分布情况灵活选择固定网格划分或是自适应网格划分,利用这二种不同的网格划分方法具有的优点,以实现进一步降低算法的时间复杂度和提高聚类结果的准确性,并使算法具有更优的可伸缩性。实验使用仿真数据表明,该算法在处理具有属性值域范围大的高维大规模数据时是实用有效的。

**关键词:**高维聚类;子空间聚类;相对熵;网格划分

**中图分类号:**TP301.6

**文献标识码:**A

**文章编号:**1673-629X(2010)10-0150-04

## A Subspace Clustering Algorithm of High Dimension Data Based on Hybrid - Grid Partitioning

XU Chang-sen

(School of Computer, South China Normal University, Guangzhou 510631, China)

**Abstract:** A subspace clustering algorithm of high dimension data set based on hybrid - grid partitioning is proposed. The impact of attribute values range to the calculation is eliminated, filtering out redundant attributes is effective to enhance the clustering accuracy and reduce time complexity. The flexibility to choose a fixed or adaptive grid partition using the advantage of them to improve time complexity and the accuracy of clustering according to the data distribution. The algorithm has better scalability, too. A set of experiments on a synthetic dataset demonstrate the effectiveness and efficiency of the algorithms when clustering on high dimensional and large - scale data with the big range of the attribute value.

**Key words:** high dimensional clustering; subspace clustering; relative entropy; grid partition

### 0 引言

科学研究与应用领域的迅速发展,积累了大量维度(属性)通常可以达到成百上千维的、属性值域大的高维数据,对此类高维数据的挖掘变得越来越重要。但是,受“维度效应”的影响,在高维数据聚类中,当数据维数高于20时,传统聚类分析的性能会急剧下降,甚至无法完成聚类任务<sup>[1]</sup>。高维数据的研究者们发现,很多真实数据的类仅存在于子空间内。迄今为止,研究者已经提出了许多子空间高维聚类算法,如综合运用基于密度和网格方法的 CLIQUE<sup>[2]</sup>、基于信息熵的 ENCLUS<sup>[3]</sup>、基于动态区间分割的 MAFIA<sup>[4]</sup>,以不同的方法来实现对大规模高维数据集进行聚类,但总的来说,各有优缺点,都没有取得较全面的性能目标。

CLIQUE<sup>[5]</sup>算法根据用户输入的参数,把数据集

每一维以网格进行等宽划分,常常使可能是某一聚类但被固定网格分割成多个区域造成边界不清晰和小的聚类被忽视,并且通过计算数据在网格中的分布,设定一个密度阈值把所得网格单元划分为稀疏和稠密两种类型,在覆盖相连密集区域时再将其相连,在高维情况下自底向上进行聚类的过程中,划分单元的数目增加使得产生大量的候选集;ENCLUS<sup>[6]</sup>利用信息熵来作为对聚类进行评价的标准,但其执行效率与 CLIQUE 方法相比没有得到明显的改善;MAFIA 根据数据分布特点采用动态区间分割对数据空间进行划分,执行效率与 CLIQUE 算法相比有很大的提高,但它适合于并行环境,实现较为复杂<sup>[6]</sup>。

基于上述各种方法存在的不足,文中提出一种基于混合网格划分子空间高维数据聚类算法:HgCluster (A subspace clustering algorithm of high dimension data set based on hybrid - grid partitioning)。实验结果表明该算法能够在保证快速聚类的前提下同时提高聚类结果的准确性,并且当数据规模越大时效果越明显。

收稿日期:2010-01-07;修回日期:2010-04-09

作者简介:许倡森(1984-),男,广东梅州人,硕士研究生,研究方向为高性能数据库计算研究。

# 1 基于混合网格划分的子空间高维数据聚类算法

## 1.1 算法主要思想

主要思路是先对数据进行标准化处理,将原始数据转化到同一个数量级别上,对一维空间的样本值排序、统计并记录存储,消除各个属性分量之间数值范围大小对计算的影响;自底向上在每个一维子空间上,根据属性值的统计信息,利用相对熵判断是否为冗余属性,若是,则去除对聚类无贡献或贡献很小的冗余属性,否则根据算法原则判断是采用固定网格划分,或是采用自适应网格划分,提高聚类精度和降低聚类时间复杂度;在一维空间中采用基于数据集划分的聚类发现算法,得到基于子空间的聚类。由此最大限度保留维度的信息,能够更加有效地解决维度困扰的问题,得到基于子空间的聚类。

## 1.2 算法描述

### 1.2.1 原始数据标准化与排序、统计

为消除各个属性分量之间数值范围大小对计算的影响<sup>[7]</sup>,首先标准化原始数据集,将原始数据均转化到同一个数量级别上。数据标准化处理时为了增加聚类的准确性,令其结果介于 0~100 之间,具体计算公式

为:  $z_i = \frac{x_i - x_i^{\min}}{x_i^{\max} - x_i^{\min}} * 100$ 。式中,  $x_i$  为原始样本值,  $z_i$

为规范化处理后的样本值,  $x_i^{\min}$  为样本集中某个属性特征的最小值,  $x_i^{\max}$  为样本集中某个属性特征的最大值。接着,对每个属性下的标准化样本值排序并统计一维空间中每一标准化属性值的样本点个数,最后存储以便于后面进一步的聚类工作。以表 1 所示的数据库为例,将原始数据库进行转置转换,转换结果如表 2 所示:对每个维的每个属性值,记录一个 tuple ID(tid)与之关联。例如,属性值 a1 出现在 tuple 1,3,4,7,9 中, a1 的 tid 列就有 5 项,称为 1,3,4,7,9(设定  $a1 > a3 > a2$ ,  $b3 > b2 > b1$ ,  $c1 > c3 > c2$ )。

表 1 原始数据表

Tid	A	B	C
0	a3	b1	c1
1	a1	b3	c1
2	a3	b1	c3
3	a1	b1	c1
4	a1	b1	c1
5	a3	b3	c2
6	a3	b3	c3
7	a1	b3	c1
8	a3	b1	c3
9	a1	b2	c1
10	a2	b3	c1

表 2 对表 1 数据的标准化及排序转置

Attribute Value	Tidlist	List Size
a1'	1,3,4,7,9	5
a3'	0,2,5,6,8	5
a2'	10	1
b3'	1,5,6,7,10	5
b2'	9	1
b1'	0,2,3,4,8	5
c1'	0,1,3,4,7,9	6
c3'	2,6,8,10	4
c2'	5	1

### 1.2.2 冗余属性的度量与过滤

在高维数据空间中,将数据点归入不同类的过程中起划分作用、对最终簇的产生有贡献的维,称为非冗余属性<sup>[3,8]</sup>。冗余属性的分布更加倾向于均匀分布,并不起划分作用<sup>[9]</sup>,会产生大量的噪声,增加算法不必要的开销,影响算法处理效果和性能<sup>[10]</sup>。所以,去除高维空间中不相关属性、降低搜索空间非常重要。熵<sup>[8]</sup>用来度量数据空间维度上数据分布的差异:当数据对象的分布在数据空间维度上趋于均匀,那么任一数据对象在特征空间哪一片区域的不确定性最大,熵值也就最大;相反,数据空间中数据对象的分布如果越集中,一个数据对象在密集区域的可能性也就越大,不确定性就越小,熵值将会越小。文中引用文献[8]相对熵的概念。相对熵<sup>[8]</sup>是数据空间中的数据对象在实际分布下计算所得的熵值与同一数据集数据对象在均匀分布情况下计算得到的熵值进行比较得到的比值,其反映了数据对象的实际分布情况趋近于均匀分布情况的程度,可以用相对熵对属性冗余与否进行区别。数据集中的数据对象在某一维度上进行投影,通过计算所得的相对熵值越大,则说明数据对象在数据空间维度上的实际分布越有可能是均匀分布的,反之非均匀分布可能性越大<sup>[8]</sup>。相对熵计算公式<sup>[8]</sup>:

$$Hr(X) = \frac{-\sum_{x \in X} d(x) \log_2(d(x))}{-\sum_{T=1}^T \frac{1}{T} \log_2(\frac{1}{T})} = \frac{H(X)}{\log_2(T)} = \frac{H(X)}{Havg(X)}, (T = |X|) \quad (1)$$

式中,  $X$  表示维度上划分后得到的直方图格 bin 的集合;  $H(X)$  表示数据对象在实际分布下直方图的熵;  $Havg(X)$  则表示相同的数据对象在当前子空间下按均匀分布得到的熵值;  $T$  是维度划分后直方图中格 bin 的数量,则  $1/T$  表示同一数据集数据对象在均匀分布情况下,所形成的每个直方图格 bin 包含的数据对象

数量在数据集中所占的比例; $x$ 表示某一个直方图格(bin); $d(x)$ 表示在实际数据分布下形成直方图中的直方图格 $x$ 对应的区域包含的数据对象数量在数据空间数据集中所占的比例。所以, $Hr(X) \in (0,1]$ 表示在由网格划分所得到的当前直方图对应的子空间中,数据集数据对象的实际分布与同一数据集对象均匀分布之间的相似度,若 $Hr(X)$ 越接近于1,维度上数据集数据对象的实际分布越接近于均匀分布<sup>[8]</sup>。

### 1.2.3 冗余属性的过滤算法

通过设置一个相对熵的阈值来区分冗余属性与非冗余属性,而不需要针对不同的数据集设定不同的阈值。相对熵的值一般设为略低于1的数据值,来作为区分冗余维度的阈值。当计算当前维的相对熵值大于此阈值时,删除当前维。

### 1.3 单维初始簇的生成

由于相对熵能够反映数据对象实际分布情况趋近于均匀分布情况的程度,所以本算法中利用用户输入的参数相对熵阈值 $H$ 为依据,判断当前维空间上应该采用固定网格划分<sup>[2]</sup>还是自适应网格划分<sup>[11]</sup>。当计算所得的相对熵值 $Hr(X)$ 大于输入参数 $H2$ 时,则采用固定网格划分进行聚类;当计算所得的相对熵值 $Hr(X) \leq H2$ 时,则采用自适应网格划分进行聚类。

算法步骤描述如下:

数据预处理,高维数据表 $T1$ 标准化、排序统计后存储得到数据表 $T2$ 。

输入:数据表 $T2$ , $T2$ 中的所有 $tid$ 的集合 $C[0][i](1,2,\dots,d)$ , $T1$ 的维数 $d$ ,区间密度阈值 $\tau$ ,固定网格划分区间数 $X$ ,用以判定某一维是否为冗余属性的相对熵的阈值 $H1$ ,用以判定某一维使用固定网格划分还是自适应网格划分的相对熵的阈值 $H2$ 。

输出:聚类结果 $C[d][m](m=1,2,\dots,k)$ (其中 $d$ 表示在第 $d$ 维空间上以第 $d-1$ 维上聚类得到的结果为输入进行聚类, $k$ 值为表示簇的数目),其中, $C[i][j]$ 表示第 $i$ 维上聚类时得到的第 $j$ 个聚类。

算法步骤:

Begin

(1) 数据预处理:对原始数据集进行标准化并排序统计后存储,得到数据表 $T2$ 及所有 $tid$ 的集合 $C[0][i]$ ;

(2) 自底向上以维为层次进行聚类:

1) 以 $C[0][1]$ 为输入,在第1维上以 $X$ 为参数对数据进行固定网格划分,运用相对熵计算公式来判定该维是否为冗余属性;若为True,则删除该维数据,转入第二步;若为False,则进一步判断是应采用固定网

格划分还是自适应网格划分方法,并进行单维聚类,得到 $C[1][m](m=1,2,\dots,k)$ ,并令 $k=p$ ;

2) 从 $i=1$ 至 $d$ 执行第3~4步;

3) 从 $m=1$ 至 $k$ ,以 $C[i-1][m](i=1,2,\dots,d,m=1,2,\dots,k)$ 为输入,在第 $i$ 维首先对数据进行固定网格划分,运用相对熵计算公式来判定该维是否为冗余属性;若为True,则删除当前维,转第4)步;若为False,则进一步判断是应采用固定网格划分还是自适应网格划分方法,并在第 $i$ 维进行聚类,得到的聚类结果为 $C[i][1]$ 至 $C[i][p]$ ;

4) 令 $i=i+1, k=p$ ;

5) 输出聚类结果 $C[d][m](m=1,2,\dots,k)$ 。

## 2 实验结果与分析

实验采用仿真数据集来测试算法的有效性,并从3个方面对算法HgCluster与CLIQUE算法的性能进行比较。本实验平台配置为: Intel 2.8 G/1G, Windows XP。

### 2.1 实验数据

本算法是用参考文献[12]的合成方法编写的数据生成器随机生成的,用户可以通过输入参数来控制获得不同的数据集,参数包括数据集中的样本数目、样本的取值范围、属性维数、子空间簇的个数、噪音数据的比例以及包含簇的子空间的维度。其中在设定相对熵阈值参数 $H1=0.96$ 、 $H2=0.78$ 时综合测试性能较好。

### 2.2 实验结果

(1) 不同数据集规模的性能比较。

如图1分析可知,数据集数据对象的规模从1000逐步增大至5000。其中数据集数据空间的维数是30,通过聚类最终在不同的7维子空间共得到5个聚类。从图1显示,当相同维数但不同数据规模情况下Gh-Cluster的聚类时间比CLIQUE算法有较明显提高,并且数据集的规模越大越明显。

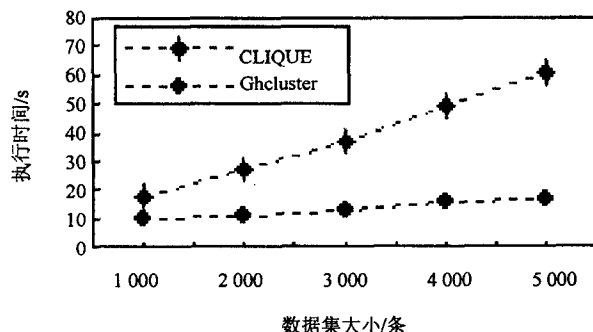


图1 不同数据集规模的性能比较

### (2) 不同数据空间维数的性能比较。

图 2 中数据集的数据空间维数由 10 到 50, 数据集数据对象的规模设为 5000, 聚类结果为在不同的 6 维子空间中通过聚类, 最后得到 8 个结果簇。依图 2 分析可知, 在相同数据集规模条件下, 本算法对数据空间的维度更具有伸缩性。

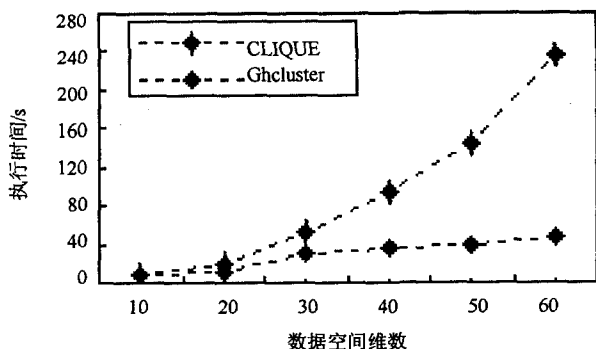


图 2 数据空间维数不同时的性能比较

### (3) 不同最高聚类维数时的性能比较情况。

如图 3 所示, 数据集中通过聚类后得到的聚类其最高维度从 4 增大至 11。数据集数据空间的维数为 50, 数据规模为 5000。如图 3 分析可知, 本算法 Gh-Cluster 与 CLIQUE 相比对数据规模具有较好的伸缩性, 性能方面也有较大的提高。

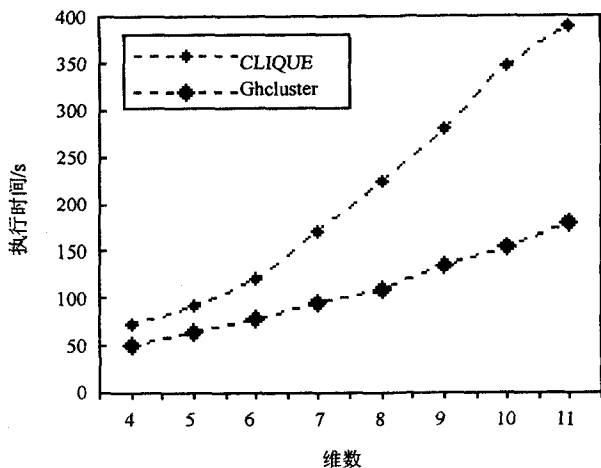


图 3 不同最高聚类维数时的性能比较情况

## 3 结束语

通过上述算法有效性验证结果表明: 文中采用的数据预处理方法消除属性值域范围对聚类的影响, 利用相对熵去除冗余维度的算法可行、有效, 在相对熵基础上灵活采用固定网格算法与自适应网格划分能较有效实现提高聚类质量, 特别是数据集规模越大, 越能体现对原算法的改进, 并能提高聚类准确性。

### 参考文献:

- [1] 和亚丽. 基于高维空间的聚类技术研究[D]. 广州: 中山大学, 2005.
- [2] 邓庚盛, 刘承启, 熊 艳. 基于网格和密度的 CLIQUE 聚类算法的研究与实现[D]. 南昌: 南昌大学, 2008.
- [3] Cheng C H, Fu A W, Zhang Y. Entropy-based subspace clustering for mining numerical data[C]//Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Diego, CA: ACM Press, 1999: 84-93.
- [4] Goil S, Nagesh H, Choudhary A. MAFIA: Efficient and scalable subspace clustering for very large datasets[R]. Evanston: Northwestern University, 1999.
- [5] 陈惠, 王 煌, 王建东. 子空间聚类算法的研究新进展[D]. 常州: 河海大学计算机信息工程学院, 2007.
- [6] 周晓云, 孙志挥, 张柏礼. 一种大规模高维数据集的高效聚类算法[D]. 南京: 东南大学, 2006.
- [7] He J, Lan M, Tan C L, et al. Initialization of cluster refinement algorithms: a review and comparative study[C]//Proceedings of IEEE International Joint Conference on Neural Networks. USA: IEEE Computer Society, 2004: 297-302.
- [8] 高 嵩. 基于相对熵的投影聚类算法研究[D]. 重庆: 重庆邮电大学, 2007.
- [9] 刘佳佳, 胡孔法, 陈 凌. 基于单维分割的高维数据聚类算法 HDCA-SDP[D]. 扬州: 扬州大学, 2008.
- [10] 夏 英, 李克非. 基于属性相关性分析的子空间搜索算法[D]. 成都: 西南交通大学, 2009.
- [11] 张伟莉, 倪志伟, 赖建章. 一种新的基于网格的聚类算法[D]. 合肥: 合肥工业大学, 2008.

(上接第 149 页)

- 2007, 17(6): 8-11.
- [6] Fowler M. Refactoring: Improving the Design of Existing Code[M]. [s.l.]: Addison Wesley Longman Inc, 1999.
- [7] Inmon W H. Building the data warehouse[M]. Fourth Edition. [s.l.]: Wiley Publishing, 2005.
- [8] Inmon W H. Commentary: The Migration Path[J]. Computer World, 1996(29): 66-72.
- [9] Dennis A, Wixom B H, Roth R M. System Analysis & Design

[M]. Third Edition. [s.l.]: John Wiley & Sons Inc, 2006.

- [10] 杜国宁, 朱仲英. 基于 Web 技术的数据挖掘系统研究与设计[J]. 微型电脑应用, 2005(8): 2-24.
- [11] Glibreath, Roy M D, Schilp J, et al. Toward to outcomes Management Information Processing Architecture[J]. Healthcare Information Management, 1996(10): 34-42.
- [12] Paul W. Modeling the Data Warehouse and the DataMart[C]//INFODB. [s.l.]: [s.n.], 1996.