DBSCAN 聚类算法的研究与改进

冯少荣[1,2]，肖文俊[1]
（1. 华南理工大学 计算机科学与工程学院，广东 广州 510641；
2. 厦门大学 信息科学与技术学院，福建 厦门 361005）

摘要：针对“基于密度的带有噪声的空间聚类”（DBSCAN）算法存在的不足，提出“分而治之”和高效的并行方法对DBSCAN算法进行改进.通过对数据进行划分，利用“分而治之”思想减少全局变量Eps值的影响；利用并行处理方法和降维技术提高聚类效率，降低DBSCAN算法对内存的较高要求；采用增量式处理方式解决数据对象的增加和删除对聚类的影响.结果表明：新方法有效地解决了DBSCAN算法存在的问题，其聚类效率和聚类效果明显优于传统DBSCAN聚类算法.

关键词：聚类；DBSCAN；划分；并行
中图分类号：TP 311 文献标识码：A 文章编号：1000-1964(2008)01-0105-07

An Improved DBSCAN Clustering Algorithm

FENG Shao-rong[1,2], XIAO Wen-jun[1]
（1. School of Computer Science and Engineering, South China University of Technology, Guangzhou, Guangdong 510641, China;
2. College of Information Science and Technology, Xiamen University, Xiamen, Fujiang 361005, China）

Abstract: An improved density based spatial clustering of applications with noise(DBSCAN) algorithm, which can considerably improve cluster quality, is proposed. The algorithm is based on two ideas: dividing and ruling, and; high performance parallel methods. The idea of dividing and ruling was used to reduce the effect of the global variable Eps by data partition. Parallel processing methods and the technique of reducing dimensionality were used to improve the efficiency of clustering and to reduce the large memory space requirements of the DBSCAN algorithm. Finally, an incremental processing method was applied to determine the influence on clustering of inserting or deleting data objects. The results show that an implementation of the new method solves existing problems treated by the DBSCAN algorithm: Both the efficiency and the cluster quality are better than for the original DBSCAN algorithm. Key words: clustering; DBSCAN; partition; parallel

Key words: clustering; DBSCAN; partition; parallel

数据挖掘技术目前已成为数据库技术的一个研究热点，在许多领域得到广泛应用[1-3]. DBSCAN[4]算法是聚类分析中基于密度的聚类算法[5-6]，其基本思想是：对于簇中的每一个点在其给定的半径范围内都至少包含给定数目的点.该算法将具有足够高密度的区域划分为一类，并可以在带有“噪声”(outliers)的空间数据库中发现任意形状的聚类，而且聚类速度快，可以作为增量聚类算法[7]的基础.但是，由于它直接对整个数据库进行操作，聚类时使用了一个全局性的表征密度的参

收稿日期：2007-01-22
基金项目：福建省自然科学基金项目(A0310008)；福建省高新技术研究开放计划重点项目(2003H043)
作者简介：冯少荣(1964-)，男，河北省南宫市人，副教授，博士研究生，从事并行分布数据库、数据仓库、数据挖掘等方面的研究.
E-mail: shaorong@xmu.edu.cn Tel: 0592-2186825

数,因此,具有两个比较明显的弱点:1)当数据量增大时,要求较大的内存支持,I/O消耗也很大;2)当空间聚类的密度不均匀,聚类间距离相差很大时,聚类的质量较差.针对DBSCAN算法存在的不足,本文提出利用"分而治之"和高效的并行算法思想对其进行改进,采用降维技术和增量处理方式提高聚类的效率.通过在标准数据集上与原DBSCAN算法进行实验比较,结果表明,新算法高效可行.

1 DBSCAN算法存在的问题

1)在聚类过程中,DBSCAN一旦找到一个核心对象,即以该核心对象为中心向外扩展.此过程中核心对象将不断增多,未处理的对象被保留在内存中.若数据库中存在庞大的聚类,将需要很大的内存来存储核心对象信息,其需求难以预料.

2)输入参数敏感.确定参数Eps,MinPts困难,若选取不当,将造成聚类质量下降.

3)由于在DBSCAN算法中,变量Eps,MinPts是全局惟一的,当数据分布不均匀时聚类质量较差.

针对DBSCAN算法存在的问题,虽然已有一些改进方法[8],但它们或多或少都存在一些不足,并没有很好解决存在问题.

2 DBSCAN算法的改进

2.1 输入参数的处理

针对DBSCAN算法对输入参数(聚类半径Eps,聚类点数MinPts)敏感问题,作如下处理.

由于参数的设置通常是依赖经验,当数据密度相差较大和类间距离分布不均匀时,很难选取一个合适的Eps值来进行聚类且得到比较准确的结果.因此,事先确定算法的参数值大小是不可取的,应在聚类过程中根据聚类结果的好坏对参数进行适当的调整.比如选择适当的评价函数作为评价聚类结果的尺度.反复调整算法的参数,重新聚类,直到聚类结果满足要求.

尽管DBSCAN算法提供了利用绘制降序k-距离图的可视化方法来选择Eps,选定的Eps值已经比较接近"理想"值;但常有微小差距,最终造成聚类结果的相差很大.可以考虑采用如下方法来加以改善:

1)可以对所有聚类对象按照从一个簇到另一个簇,按簇边缘→簇核心→簇边缘的顺序排序.这样,该对象序列就可以反映出数据空间基于密度的簇结构信息.基于这些信息可以容易地确定合适的Eps值,并随之发现各个簇.

2)不对原始数据集进行聚类,而是通过从数据集合中抽取高密度点生成新的数据集合,并修改密度参数,反复进行这一过程,直到生成的数据集合可以很容易地被聚类为止,然后以此结果为基础,再将其它点逐层地吸附到各个类中.这样,就避免了DBSCAN算法中输入参数对聚类结果的影响.

3)采用核聚类的思想对样本集进行非线性变换,使样本集的分布尽可能地均匀,经过核函数的映射使原来没有显现的特征突现出来,然后再用全局参量Eps,从而能够更好地聚类,得到较好的结果.

4)在绝大多数聚类结果不理想的情况下,是Eps值选择过小,导致本应为一个簇的对象集合被分析成了多个子簇.被分开的子簇共享一些对象,可以认为子簇通过这些共享的对象相互连接.而DBSCAN算法将子簇的连接信息简单地丢掉.因此,可以通过记录下所有的簇连接信息,由用户根据实际的聚类结果和簇连接信息,将被错误分开的子簇合并.这样可以提高聚类的效果,而输入参数Eps的变化对聚类结果的影响,就被最后的合并过程屏蔽掉.

可以考虑以下两种方式进行改进:

1)并行化[9].从DBSCAN算法可以看出,全局变量Eps值影响了聚类质量,尤其是数据分布不均匀时.因此,考虑对数据进行划分,每一个划分中的数据分布相对较均匀,根据每个划分中数据的分布密集程度来选取Eps值.这样一方面降低了全局变量Eps值的影响,另一方面由于具有多个划分,因此考虑并行处理,从而提高聚类效率,也降低了DBSCAN算法对内存的较高要求.

2)增量式处理.当数据增加或者删除时,只考虑其增加或删除的数据所影响到的那些类.这种方法在处理大数据集时非常有效,不需要重新对数据库中的数据进行聚类,只需要对类进行渐进性地更新,修正和加强已发现的类.另外,由于高维数据的复杂性,使聚类分析的效率和实用性都很差.通过确定聚类空间中和聚类主题相关性较强的数据维,来降低聚类空间的维度.利用数据降维可以降低数据结构上的复杂性.目前,有多种降维技术[10-11],均可用于特征空间的削减.在方法的选择上应根据降维后,信息丢失率在可接收范围内,来选择一种合适的降维方法.

2.2　数据划分——网格的产生

数据经过降维处理，实现从多维到二维的多维变换后，就可以实现数据的划分.

2.2.1　数据划分过程

在数据的分区中，可以利用在关系数据库系统提供了 5 种内部聚集函数 count()，sum()，avg()，max()，min()来进行数据分布特性的统计.

定义　所谓“步长密度”是指在一个度量长度内的点数除以整个样本数据中的点数.

在划分中使用直方图来确定数据分布的密度情况.直方图由一组矩形组成，这些矩形反映了落在给定区间内的点数占总的样本数据的多少.在画直方图之前，首先要确定分组数和数据样本点中的最大值和最小值.最大值和最小值可以使用内部聚集函数求出，而组数的值不宜过大或过小，如果组数取得过大，则有的区间内没有样本观测值，过小则无法看出数据点的分布情况.组数（m）和数据点（n）之间一般应该满足

$$m \approx 1.87(n-1)^{\frac{2}{3}}.$$

组数确定之后，可求出每一个区间内的点数，从而求出该区间内的点的密度情况，如图 1 所示.

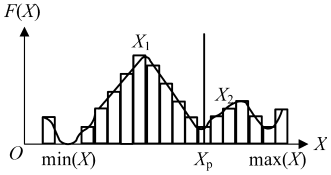

图 1　数据在 X 轴的分布密度

Fig. 1　Distribution density of data on X axis

通过每个矩形上边的中点顺次连接成一条曲线，该曲线反映出数据点的大概密度分布情况.为了避免把一个类分裂成多个类，在两个相邻波峰之间的波谷位置选择划分点，即数据分布最稀疏的位置确定划分点.在划分中，给定一阀值 λ，用来决定是否需要在相邻的两个波峰之间确定一个划分点，用向量（L_1，S_0，L_2）来存储相邻的两个最大值点 L_1（波峰）和 L_2（波峰）以及相邻两个波峰之间的波谷 S_0，当 $|L_1 - L_2| > \lambda$ 时，在 S_0 所对应的坐标轴上的点处进行划分，所有这些坐标轴上的点构成集合 S，S 中存储了所有的划分点.从图 1 中的曲线可以看出，X_1，X_2 为相邻的两个波峰值，而且这两个波峰值相差较大，因此，可以在波谷位置确定划分点为 X_p.分别对 X，Y 轴作划分步骤，确定划分点.从图 1 中可以看出，并不是所有的情况都需要进行数据的划分，当发现数据的分布密度相差不大时，就不需要进行划分（连续分布时）.

当出现下面 3 种情况时：1）数据分布密度为一条直线；2）数据分布密度呈现递增的情况；3）数据分布密度呈现递减的情况.说明相邻区域内的数据分布密度较一致，不存在数据分布较稀疏的区域（边界区域除外），这时需要把两个维上的数据划分结合起来考虑，并根据处理机数目进行数据划分.假设处理机数目为 N 个，在某一维上划分了 K_i（$i = X$ 或 $i = Y$）个区域，另一维上平均划分成 N/K_i 个区域.

若不是上述 3 种情况，就需要用上述阀值进行确定.但当出现互相包含的环状类和互相缠绕的螺旋状类时，用这种简单的投影到 X，Y 轴的方法并不见效.

2.2.2　数据划分算法 Partition

算法：Partition 算法

输入：二维的样本数据文件 F，处理机数目 N.

输出：每维上的划分点 S（$S \geqslant 1$）.

1）根据样本点数确定组数 m 和样本数据中的最大值 max 和最小值 min.

2）计算步长 $d = \dfrac{\max - \min}{m}$.

3）对 m 个组的每一个执行：

a. 计算步长密度 $\rho_i = \dfrac{\text{步长之间的点数}}{\text{总的样本点数}}$；

b. 存储 m 个向量（步长起点，步长终点，ρ_i）.

4）对 m 个组的每一个执行：// 判断 ρ_i 值的变化情况.

if ρ_i 没有变化 or ρ_i 呈单调递增 or ρ_i 呈单调递减

then

平均进行分配 N/K_i；确定 S 个（$S = N/K_i - 1$）划分点.

else

依次判断相邻的两个波峰值之间的差的绝对值是否大于阀值 λ，如果大于，记下波谷 S_0 对应的坐标值 S_i；

$$S = S \cup S_i.$$

5）输出所有的划分点 S.

对 X 轴和 Y 轴分别采用这种数据划分方法确定划分点，在某一维上可以定义一个划分向量 V_i（$i = X$ 或 Y）=（P_{i0}，P_{i1}，…，P_{in}），设第 i 维的第 k 区间记为 $I_{ik} = [l_{ik}, h_{ik}]$，这样每一个长方形为 2 个不同维区间的笛卡儿乘积 $[l_{1k1}, h_{1k1}] \times [l_{2k2}, h_{2k2}]$，该长方形称之为网格，每个网格可以用表达式（$l_{1k1} \leqslant x \leqslant h_{1k1} \wedge l_{2k2} \leqslant y \leqslant h_{2k2}$）表示，这个网格可以使用坐标定义为（$K_1$，$K_2$），这就可实现把

分布较均匀的数据划分到一个长方形的网格中,从而把各个网格分配给多个处理机进行单独的DBSCAN聚类.这样处理之后,数据分布较均匀,各个区域之间不会由于Eps值的选择而受到影响,提高了聚类质量.另一方面,由于多个处理机对数据进行聚类,从而提高了聚类效率.

2.3 网格分配算法

网格分配算法实现在处理机间分配由Partition算法产生的网格.在该算法中使用了一个坐标和求模函数:$CMD(K_1, K_2) = (K_1 + K_2) \bmod N$,实现处理机之间分配网格的,即坐标$(K_1, K_2)$的网格被分配给处理机$CMD(K_1, K_2)$.

算法:Dallocation算法.

输入:存储在处理机O上的数据文件F,以及划分向量$(P_{i0}, P_{i1}, \cdots, P_{in})$,$i = X$或$Y$.

输出:由Partition算法产生的网格

1)对数据文件F的每个网格T执行:

a.由F的划分向量$(P_{i0}, P_{i1}, \cdots, P_{in})$计算网格$T$所属网格的坐标定义$(K_1, K_2)$;

b.$k = CMD(K_1, K_2)$;

c.把网格$(l_{1k1} \leqslant x \leqslant h_{1k1} \wedge l_{2k2} \leqslant y \leqslant h_{2k2})$发送到处理机$k$.

2)对N个处理机的每一个执行:

处理机i接受并存储处理机O发送来的网格T,第一个FOR循环BEGIN和END之间的语句均在处理机O上执行.

2.4 基于数据划分的并行DBSCAN算法

一旦数据划分完成,把各个网格分配给处理机之后,就可以并行地对各个处理机使用DBSCAN算法,每一个数据区域可以选取本机的Eps值.各个处理机也建立了一个R^*树,便于区域查询.具体过程为:1)构建本处理机数据的R^*树;2)选取适合本机的Eps值;3)根据本机的Eps值进行聚类.

为了提高局部聚类的合并效率,需要在局部聚类过程中记录下噪声点、类的边界点以及划分的边界信息.因它们可能是全局中某个聚类的边界点或某一被分割的小聚类中的点.由于划分可能使得本属于同一个类的数据点被划分到相邻的两个区域中,最后需要对两个类进行合并.为了提高聚类效率,在局部聚类过程中已经将可能有用的信息保存了下来,如噪声点,边界点信息.类的合并涉及到3种情况:

1)两个类A,B合并,当且仅当:

a.A,B分别处于相邻的两个网格P_A,P_B中;

b.设Eps(A),Eps(B)分别是网格P_A,P_B的Eps值,Eps(P_A, P_B) = min{Eps(A),Eps(B)},两个类中的边界点之间的距离小于等于Eps(P_A, P_B).

2)对噪声点的归并

处在网格边界的噪声点N可能是全局中某个类C的边界点,此时需要把噪声点N归到该类C中.当噪声点N满足以下条件时就进行归并:

a.类C和噪声点N分别处在两个相邻的网格中;

b.设E_C是类C中的边界点,如果边界点和噪声点N之间的距离小于等于Eps(P_C).

3)多个噪声点产生新类

在数据划分时,一些较小的类被分到不同的网格中,由于在同一个网格中的数据量较小,因此这些点被认为是噪声点.首先随机选取一个噪声点E_i,如果噪声点E_i和其它噪声点E_j之间的距离小于等于Eps值,则把E_i看成一个新类的核心对象点,然后剩下的噪声点按照第二种方法进行归并.

2.5 DBSCAN算法中的增量聚类分析

数据库中的数据不是一成不变的,有的数据库需要定期进行更新操作.这样就会增加一些数据,或者删除一些数据.这些数据的增加或者删除都会影响系统中已经存在的类.可能会出现:1)有的类元素减小;2)新增加一些类;3)出现一些新的噪声点;4)使原来的多个类合并成一个类;5)使原来的类分裂成多个类.对数据库中的数据进行增加或者删除都是一项非常频繁的操作,尤其是在WWW日志数据库中.

因此,若对更新后的数据重新聚类是一个非常耗时的工作.因为增加或删除的数据只会影响到与这个数据相邻的小部分数据类别,而其它大部分数据类别是不会受到增加或删除数据的影响.另一方面,如果数据一更新,就重新聚类,这一点在时间效率上也不允许,因为数据的更新会经常发生,而且在大型数据库中这样做也不现实.

所以,基于DBSCAN算法,在此讨论增加或者删除数据元素对已有类的影响.不论是增加还是删除数据都可能会改变邻域内对象的属性,即原来是核心对象的可能会成为非核心对象,原来是非核心对象而成为核心对象.当增加了数据对象P,则P的邻域内原来是边界点或者噪声点的那些点可能变成核心对象,相应会建立新的密度相连的链,当删除P,原来P的Eps邻域内的核心对象可能会成为非核心对象,原来的一些密度相连的链就会被

删除.

2.5.1　数据元素的增加

增加数据可能对已有类的影响[2-3]：

1) 增加的数据点可能本身就是一个噪声点，这种情况下，该噪声点不会影响其它的类.

2) 原来的一些噪声点和新增加对象 P 建立一个新的类别.

3) 新增的对象 P 成为原来某类中的一个对象.

4) 原来的两类由于新增对象 P 的插入而进行合并.

2.5.2　数据对象的删除

数据对象的删除也可能会影响删除对象邻域内的对象的核心对象属性，使原来的核心对象成为非核心对象，所以，原来密度相连的一些链的序列就被删除，引起一些类的分裂；也可能会使原来的一些核心对象变成噪声点，从而使类中的数据对象减少.也可能使原来的一些数据对象比较少的类被删除，类中原来的数据对象成为噪声点.

因此，如果能够很好地利用原始数据的聚类结果来对现在更新了的数据进行聚类分析，就可以在很大程度上解决对大数据量进行聚类的效率问题.

2.6　算法复杂性讨论

DBSCAN 时间复杂度为 $O(n^2)$，其中 n 为数据对象数目.当 n 很大时，DBSCAN 算法耗时 T_D 为 n^2t.改进的算法耗时主要在局部聚类及聚类合并时交叠区点集聚类信息传输上，而数据传输并行执.设 k 为数据分区数，φ 为交叠比，因为数据近似均匀划分，每个节点上对象数目为 $[n(1+\varphi)]/k$，又设传输一个整数(对象聚集结果用整数表示)的时间为 T，则数据传输所花时间为 $n\varphi T/k$，不考虑聚类合并耗时，改进的 DBSCAN 耗时 $T_{改进D}$ 为：$[n(1+\varphi)/k]^2t+n\varphi T/k$，加速比为

$$\alpha=\frac{T_D}{kT_{改进D}}=\frac{n^2k}{n^2(1+\varphi)^2+n\varphi k\dfrac{T}{t}}. \quad (1)$$

若采用空间索引，DBSCAN 算法的时间复杂度为 $O(n\lg n)$，改进的 DBSCAN 加速比为

$$\alpha=\frac{n\lg n}{nk(1+\varphi)[\lg n^{+}\lg(1+\varphi)^{-}\log k]+n\varphi\dfrac{T}{t}}. \quad (2)$$

对于同一个系统，T/t 值是确定的，从式(1)可看出控制交叠比并选择合适的数据分区数 k，可以得到一个好的加速比.

对于高维情况，可以像二维空间那样进行数据划分，只是划分更加复杂而已.虽然改进后的 DBSCAN 算法其时间复杂度由 3 部分(数据划分及网格分配、并行聚类、合并)决定，但把数据划分及网格分配作为预处理部分，其耗时没有计入到算法消耗的总时间中，如同 DBSCAN 算法没有将构建 R^*-树所花时间(尽管很花时间)计入算法总时间内一样.若不考虑类的合并耗时，在时间复杂性方面，改进后的 DBSCAN 算法时间复杂度与 DBSCAN 算法时间复杂度 ($O(n\lg n)$) 比较相似.可粗略地表示为 $O(n_1\lg n_1+n_2\lg n_2+\cdots+n_k\lg n_k)$，$n_i$ ($i=1,2,\cdots,k$) 为各个数据分区的数据对象总量.

3　实验结果

使用改进后的 DBSCAN 算法对图 2 中的数据对象进行聚类分析.根据数据密集程度，把数据分成 3 个区域(如图 3 所示).区域 1：$2.05\leqslant x\leqslant 4.33\wedge 3\leqslant y\leqslant 4.66$；区域 2：$4.33\leqslant x\leqslant 7.66\wedge 0.66\leqslant y\leqslant 3$；区域 3：$0.2\leqslant x\leqslant 3.1\wedge 0.66\leqslant y\leqslant 3$.

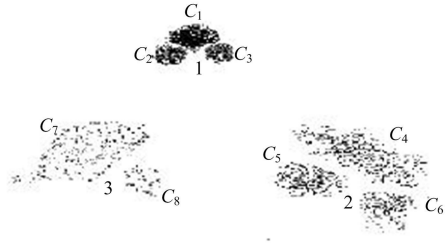

图 2　分布不均匀的数据对象

Fig. 2　Non uniformly distributed data objects

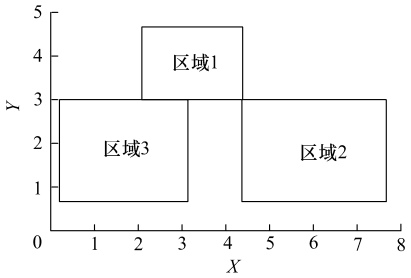

图 3　根据密度划分的区域

Fig. 3　Area based on density partition

对 3 个区域按 Eps=0.16，0.20，0.23 分别进行聚类，聚类结果如表 1.若使用统一的 Eps，则结果为表 2.

表 1　不同 **Eps** 的聚类结果

Table 1　Clustering result of different Eps

区域	Eps 值	聚类结果
1	0.16	3 个类(C_1, C_2, C_3)
2	0.20	3 个类(C_4, C_5, C_6)
3	0.23	2 个类(C_7, C_8)

表 2 统一的 Eps 的聚类结果

Table 2 Clustering result of unified Eps

Eps 值	聚类结果
0.20	6 个类，C_1, C_2, C_3 合并成一个类
0.18	7 个类，C_7 分裂成两个类

结果表明，DBSCAN 无论取什么 Eps 值，都无法得到完全正确的聚类结果；而使用改进后的 DBSCAN，由于对 3 个数据分区分别取不同的 Eps 值，对 3 个数据分区并行进行聚类，获得的聚类结果与实际情况一致. 同时，使用分区之后的聚类更加合理，质量更高.

使用分区后的执行效率也得到了一定的提高，如图 4 所示. 各个分区的执行时间明显小于没分区所需要的总的时间. 而且分区后总的时间小于没分区时总的时间，因此分区对效率也有一定的提高.

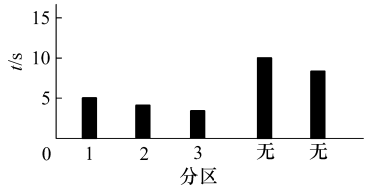

图 4 各分区的执行时间

Fig. 4 Each partition execution time

如果同时使用模拟数据和真实数据进行测试，模拟测试数据集来自 DBSCAN 算法中的 database，测试的真实数据用的是 SEQUOIA2000 数据库. 数据量从 2 万到 15 万个点. 测试的硬件环境是：P4-2 4 GHZ 的CPU，主存为512 M，硬盘为80 G，7 200 r/min. 软件环境是：操作系统为 Microsoft Windows 2000 Server，算法实现的语言为标准 C++. 表 3 给出了一组测试结果，从中可以看到，改进后 DBSCAN 算法的运行时间快于原始 DBSCAN 算法的运行时间，且随着数据量的加大，差距更加明显. 可见新算法是一种效率非常高的聚类算法. 且当数据量增大时，改进后的DBSCAN 算法所需时间的增幅比原始 DBSCAN 算法小.

表 3 SEQUOIA2000 数据库的一组测试结果

Table 3 A group of testing results of SEQUOIA2000 database

数据量（点数）/万	改进后DBSCAN 算法需要的运行时间/s	原始 DBSCAN 算法需要的运行时间/s
2	0.13	0.85
4	0.32	5.40
6	0.66	12.70
8	0.98	19.34
10	2.05	28.12
15	3.70	61.33

4 结 论

本文对DBSCAN 算法的优缺点进行了详细分析和研究，分析了聚类质量对输入参数 Eps 的依赖. 根据 DBSCAN 存在的问题，使用了“分而治之”和高效的并行算法思想，把数据划分成分布均匀的网格，对每个网格单独处理，分配网格到多个处理机共同聚类. 这样一方面克服了全局变量 Eps 的影响，提高了聚类质量；另一方面，提高了聚类效率，也降低了 DBSCAN 对主存的较高要求. 最后分析了聚类过程中数据对象的增加和删除对聚类的影响；对改进的聚类算法给出了实验结果. 实验结果中表明：改进的聚类算法无论是在聚类质量还是在聚类效率上都得到了比较满意的结果.

参考文献：

[1] LIU Zhen-hua, JIANG Zhen-quan, ZUO Ru-song. Study of fussy clustering of engineering geological environment with GIS[J]. Journal of China University of Mining & Technology, 2003, 13(2): 196-200.

[2] 沈 斌，姚 敏，温长洋. 一种基于混合模型的时间序列数据挖掘系统[J]. 中国矿业大学学报，2003, 32(3)：284-288.
SHEN Bin, YAO Min, WEN Chang-yang. A system of time series data mining based on hybrid model[J]. Journal of China University of Mining & Technology, 2003, 32(3): 284-288.

[3] 刘高军，朱 嬿. 基于数据挖掘技术的建筑企业信用评价[J]. 中国矿业大学学报，2005, 34(4)：494-499.
LIU Gao-jun, ZHU Yan. Credit evaluation of onstruction companies based on data mining[J]. Journal of China University of Mining & Technology, 2005, 34(4): 494-499.

[4] ESTER M, KRIEGEL H, SANDER J, et al. A density-based algorithm for discovering clusters in large spatial databases with noise[C]// Proc of the 1996 2nd Int'l Conf on Knowledge Discovery and Data Mining. Portland: AAAI Press, 1996: 226-231.

[5] QIAN Wei-ning, GONG Xue-qing, Ao Ying-zhou. Clustering in very large databases based on distance and density[J]. Journal of Computer Science and Technology, 2003, 18(1): 67-76.

[6] 周水庚，周傲英，曹 晶，等. 一种基于密度的快速聚类算法[J]. 计算机研究与发展，2000, 37(11)：1287-1292.
ZHOU Shui-geng, ZHOU Ao-ying, CAO Jing, et

al. A fast density-based clustering algorithm[J]. Journal of Computer Research and Development, 2000, 37(11): 1287-1292.

[7] ESTER M, KRIEGEL H P, SANDER J, et al. Incremental clustering for mining in a data warehousing environment[C]// Proceedings of the Twenty-Fourth Very Large Data Bases (VLDB) Conference. New York: [s. n.], 1998: 323-333.

[8] 周水庚, 范　晔, 周傲英. 基于数据取样的 DBSCAN 算法[J]. 小型微型计算机系统, 2000, 21(12): 1270-1274.
ZHOU Shui-geng, FAN Yu, ZHOU Ao-ying. A sampling-based DBSCAN algorithm[J]. MINI-MICRO Computer System, 2000, 21(12): 1270-1274.

[9] JIN R, YANG G, AGRAWAL G, et al. Shared memory parallelization of data mining algorithms: Techniques, programming interface, and performance[J]. IEEE Transactions on Knowledge and Data Engineering, 2005, 17(1): 71-89.

[10] SHI Yong, SONG Yu-qing, ZHANG Ai-dong. A shrinking-based clustering approach for multidimensional data[J]. IEEE Transactions on Knowledge and Data Engineering, 2005, 17(10): 1389-1403.

[11] 陈　莉, 焦李成. 文档挖掘与降维技术[J]. 西北大学学报: 自然科学版, 2003, 33(3): 267-271.
CHEN Li, JIAO Li-cheng. Text mining and technology of dimensionality reduction[J]. Journal of North West University: Natural Science Edition, 2003, 33(3): 267-271.

（责任编辑　邓　群）

《中国矿业大学学报》(中文版)2007 年第 5 期被 Ei 收录论文(二)

论文题目	第一作者
神府煤显微组分表面性质研究	段旭琴
煤孔隙结构对煤层中 CO 扩散的影响	郭立稳
基于黏弹性的超高分子量聚乙烯齿轮计算分析	郑晓雯
龙口褐煤萃取后的微孔结构及化学组成变化	张小东
$MnZnFe_2O_4$ 磁性纳米添加剂抗磨减磨及自修复效应研究	冯雪君
直剪试验的面积校正方法及误差分析	徐志伟
采煤塌陷地充填复垦土壤呼吸的研究	牟守国
基于井距分布的多类资料整合砂体建模	袁新涛
数论变换与周期性序列关系在图像压缩中应用	张　虹
微生物异化还原金属氧化物的机理及应用	李浩然
电热还原法制备富含一氧化碳煤气研究	田玉仙
充填复垦土壤水分竖直运动模拟研究	王　辉
平均空间占有量对混沌的度量——均匀度理论之应用	罗传文
改性黏土对 PET 纳米复合材料结晶性能的影响	陈汉周

摘自《Engineering Village 2》