Computer Science and Application 计算机科学与应用, 2012, 2, 114-120
doi:10.4236/csa.2012.22021 Published Online June 2012 (http://www.hanspub.org/journal/csa)

A Spatial Clustering Algorithm Based on Neighbor Distribution*

Zengfang Yang[1], Yang Yang[1], Min Xie[2]

[1]School of Information Technology and Engineering, Yuxi Normal University, Yuxi
[2]School of Information, Yunnan Normal University, Kunming
Email: yzf@yxnu.net

Received: May 3rd, 2012; revised: May 21st, 2012; accepted: May 27th, 2012

Abstract: The spatial clustering is an important tool for analyzing spatial data. Clusters in the spatial database may be of arbitrary shape e.g. spherical, drawn-out, linear, elongated etc. Therefore, to discover clusters of arbitrary shape be an important quality standard for spatial clustering algorithm. In this paper, we present a spatial clustering algorithm based on the nearest distance distribution, the algorithm is based on a reasonable assumption that in the particular part of data space, points within same cluster are uniformly distributed. Experimental analysis shows that the method can discover clusters of arbitrary shape and it is efficient and useful for large spatial database.

Keywords: The Nearest Neighbor; Spatial Clustering Algorithm; Cluster

一种基于近邻分布的空间聚类方法*

杨增芳[1]，杨　扬[1]，解　敏[2]

[1]玉溪师范学院信息技术工程学院，玉溪
[2]云南师范大学信息学院，昆明
Email: yzf@yxnu.net

收稿日期：2012 年 5 月 3 日；修回日期：2012 年 5 月 21 日；录用日期：2012 年 5 月 27 日

摘　要：空间聚类是分析空间数据的一种重要工具，空间数据库中的聚类形状可能会是任意形状的，比如球状的、持续的、延伸的等，因此，能否发现任意形状簇成为衡量空间聚类算法质量的一个重要标准。本文中我们提出一种基于最近邻距离分布的空间聚类方法，这个算法是基于这样一个假设，假设在数据空间的某个特定部分，一个聚类内部的点是均匀分布的。实验分析表明，该方法不仅能发现任意形状的簇，而且对于大型空间数据库是高效有用的。

关键词：最近邻；空间聚类算法；聚类

1. 引言

近年来，遥感、地理信息系统、计算机制图学、环境评估与规划等许多领域中收集而来的大量数据被存储到空间数据库中，使得大规模空间数据库中的知识发现变得尤为重要。空间聚类是空间数据挖掘中的一个重要的任务和重要步骤，空间聚类是将数据库中的数据划分成具有一定意义的子类，使得不同子类中的数据尽可能相异，而同一子类中的数据尽可能相同。聚类在空间数据库中的应用如地震学、矿产资源探测等。对于大型空间数据库中的应用而言，对聚类算法有如下要求：一是能够发现任意形状的聚类，二是在大型数据库上效率好。针对以上二个要求，本文中我们将提出一种新的基于最近邻的空间聚类算法。

*资助信息：云南省教育厅科学研究基金资助项目“空间数据挖掘技术在土地利用中的应用研究”(编号：2011Y074)。

本文内容组织如下：第 2 节对空间聚类的研究情况作一个简单的概述，并提出了本文需要研究解决的问题。第 3 节中，提出了最近邻及最近邻距离分布的概念，在此基础上对聚类进行定义。第 4 节中，提出基于最近邻距离分布的聚类算法思想(简记为 SCAND 算法)，并对 SCAND 算法进行描述。第 5 节通过实验对 SCAND 算法的有效性和有用性进行评价。第 6 节对本文所做的工作进行总结。

2. 空间聚类概述

2.1. 空间聚类概述

空间聚类(Spatial Clustering)是要在一个较大的多维数据集中采用距离度量以标识出聚类，使得同一聚类中的对象具有较高的相似度而不同聚类中的对象彼此不同[1]。空间数据挖掘能为地理区域数据库提供许多有用的工具来分析地理数据，而空间聚类分析是其中工具之一，例如，可以用聚类分析来识别出地球观测数据库中使用性质类似的地域，或者合并具有相同天气模式的地区。作为数据挖掘的一个功能，空间聚类可以作为一个单独的工具用于获取数据的分布情况，观察每个聚类的特征，关注一个特定的聚类集合以深入分析。空间聚类也可以作为其它算法的预处理步骤，比如分类和特征描述，这些算法将在已发现的聚类上运行。由于在许多领域有非常广泛的应用，空间聚类已成为数据挖掘研究中的一个很活跃的领域，并且有了许多聚类算法，这些聚类算法可以划分为四大类[2]：划分方法、层次方法、基于密度的方法、基于网格的方法等。

2.2. 相关研究工作

目前，针对于大型空间数据库所提出的聚类算法中，典型代表是 CLARANS 算法[3]和 DBSCAN 算法[4]。Raymond T. Ng 等人提出的 CLARANS 算法是第一个为了大型空间数据库而设计的基于划分的聚类算法，其优点是能够发现孤立点，聚类质量较好，缺点是对数据输入顺序异常敏感，而且该算法只能处理凸面形状或球状聚类，不能发现任意形状的聚类。Ester Martin 等人提出的 DBSCAN 算法是一种基于密度的聚类算法，它能够在有噪声的空间数据库中发现任意形状的聚，聚类速度较快，且能够有效处理噪声点和发现任意形状的聚类，缺点是该方法对参数 Eps 及 MinPts 非常敏感，且这两个参数很难确定。除了采用现有的传统聚类方法外，针对空间数据库特点，现有的一些成熟的技术或思想也可以直接引入空间聚类中。SADBS[5]能够有效地存储和管理空间数据，而且有强大的空间拓扑关系分析功能。鉴于此，文献[6]中针对空间对象的数据结构和索引结构提出了基于密度的 RTDBSCAN 算法，基于 Delaunay 三角剖分提出了一个用于空间约束数据聚类的 DHCA 算法，并设计原型系统 SDCAS 来实现。这种方法借助 SADBS 来存储和管理空间数据，而且充分利用了其强大的空间拓扑关系分析功能，但是该系统忽略了空间对象的非空间属性，聚类质量的完整性和精确性不高。文献[7]将空间信息和属性信息纳入统一的空间距离测度和空间聚类分析系统，将会改善空间分析和空间数据挖掘的信息质量，同时将 GIS 和 SAS 或 SPSS 集成起来,提高了效率，但对于一些 GIS 特有的复杂的地图或地形建模数据，需要额外的数据格式转换工作。

2.3. 本文研究解决的主要问题

对于大型空间数据库中的应用而言，对聚类算法有如下要求：首先是能够发现任意形状的聚类，因为空间数据库中的聚类形状是任意的，比如可能会是球状的、持续的、延伸的等。其次是要求算法在大型数据库上效率好，即，在远远超过几千个对象的数据库上算法能够工作得较好。针对以上两个要求，本文中我们将提出一种新的基于最近邻的空间聚类算法 SCAND 算法(A Spatial Clustering Algorithm Based on the Nearest Distribution)，这个算法是基于这样一个假设：假设在数据空间的某个特定部分，一个聚类内部的点是均匀分布的。下面我们先对算法中用到的相关术语进行定义，然后再详细介绍 SCAND 算法的思想。

3. SCAND 算法的相关工作

3.1. 相关定义

定义 1(最近邻)设 S 是由一些点所构成的集合，$p \in S$，点 p 在 S 中的最近邻表示为 NNs(p)，它是集合 $S-\{p\}$ 中离点 p 距离最近的点。也就是说，如果 $\exists p \in S, \forall q \in S-\{p\}$ 有 $Dist(p,q)=Min\{Dist(p,q)\}$ (其中 $Dist(p,q)$ 表示 p 与 q 之间的距离)，那么点 q 就叫做

集合 S 中点 p 的最近邻，记为 NNs(p)，有 NNs(p) = q。

定义 2(最近邻距离)设 S 是一个点集，$p \in S$，q 是集合 S 中 p 的最近邻，点 p 的最近邻距离表示为 Dist__NNs(p)，它表示 p 与 q 之间的距离，即 Dist__NNs(p) = Dist(p,q)。

定义 3(最近邻距离集)设 S 是一个点集，$s_i \in S$，点集 S 的最近邻距离集表示集合 S 中所有元素的最近邻距离的集合，记为 DistSet__NN(S)，它是所有 Dist__NNs(s_i)取值的集合，即 DistSet__NN(S)={Dist__NNs(s_i)｜$\forall s_i \in S$}。

3.2. 最近邻距离的概率分布

下面我们将对一个聚类中点集的最近邻距离的概率分布作进一步的分析，我们的分析是基于这样一个假设：假设在数据空间的某个特定部分，一个聚类内部的点是均匀分布的，但是并不要求数据库中所有的点都服从同样的分布。

为确定最近邻距离 D 的概率分布，我们假设 N 个点均匀分布在某一个数据空间 R 中，记这个数据空间 R 的容量为 Vol(R)。假设这 N 个点独立地“落入”到数据空间 R 中，设 S 是 R 的一个子空间，容量为 Vol(S)，则其中的一个点落入到子空间 S 中的概率为 Vol(S)/Vol(R)。空间 R 中的点 q 的最近邻距离 D 大于 x 的概率，等于没有一个点落入到以 q 为中心、半径为 x 的超球体中的概率，记这个超球体为 SP(q, x)，有：

$$P(D > x) = 1 - Vol(SP(q,x)) / Vol(R) \quad (1)$$

由于上面已经假设空间 R 中的 N 个点都是独立分布的，所以根据(1)式，这 N 个点都不落入超球体 SP(q, x)中的概率为：

$$P(D > x) = \left(1 - Vol(SP(q,x)) / Vol(R)\right)^N \quad (2)$$

因此，D 不大于 x 的概率等于 N 个点都落入到超球体 SP(q, x)中的概率，其概率为：

$$\begin{aligned} P(D \le x) &= 1 - P(D > x) \\ &= 1 - \left(1 - Vol(SP(q,x)) / Vol(R)\right)^N \end{aligned} \quad (3)$$

因此，在二维空间中，最近邻距离 D 的分布函数就为：

$$F(x) = P(D \le x) = 1 - \left(1 - \pi x^2 / Vol(R)\right)^N \quad (4)$$

从(4)式中我们可以看出，这个分布函数有两个参数 N 和 Vol(R)，这里 N 就是空间 R 中的点数，因此 N 很容易确定，Vol(R)是聚类面积，由于落入到区域 Vol(R)中的点的集合很可能具有任意形状，因此有时候计算 Vol(R)就比较困难一些，这也是我们需要解决的问题，下面 3.3 节中介绍一种方法来计算一个不规则形状的聚类面积。

3.3. 不规则形状聚类面积的计算方法

严格来说，一个点的集合是不存在面积的，但是，我们可以为点集 S 设定某些近似表示，然后通过计算这些近似表示的面积来得到点集 S 的近似面积。那么该如何为某个点集 S 选择合适的近似表示呢？主要考虑以下两点：一是近似表示的形状应该尽可能和聚类的形状相似；二是这个近似表示应该是相连的。我们可以采用一种基于网格的方法来确定聚类的面积，如图 1 所示。

利用网格来计算不规则聚类面积的方法中，关键的问题是：对于某个给定的点集，网格长度值该如何选择才比较合适？假设所选择的网格长度过大，那么它和聚类形状就不太近似；若网格长度选择过小，则该近似表示有可能被分裂成互不相连的几个多边形。对某个聚类 C 而言，通常，我们把网格长度设为其 DistSet__NN (C)集合中的最大值。为了便于阐述聚类面积的计算问题，我们给出如下一些定义：

定义 4(网格长度)网格长度是正方形网格单元的长度。设某个聚类中点的集合为 S，该聚类相似多边形的网格长度记为 GL(S)，则 GL(S)为 S 的最近邻距离集合中的元素的最大值，即，GL(S) = MAX{DistSet__NN(S)}。

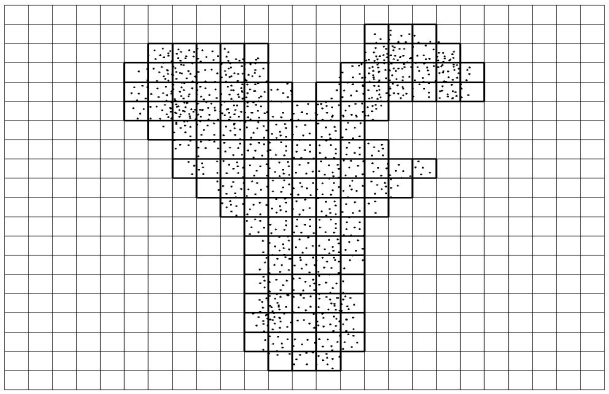

Figure 1. Discover clusters of arbitrary shape
图 1. 发现任意形状的聚类

定义 5(被填充的网格单元)若某个网格单元至少包含了聚类点集 S 中的一个点，则把这个网格单元称为被填充的网格单元。

定义 6(聚类区域的近似表示)聚类 S 的区域的近似表示定义为所有被 S 中的点所填充的网格单元的并集。

定义 7(聚类面积)聚类 S 的面积定义为所有被 S 中的点所填充的网格单元面积之和。

基于以上聚类面积的网格计算方法，可以计算出如图 1 所示聚类的面积，其中，聚类面积用粗线条网格单元所包含的面积来近似表示。

通过以上的讨论，我们可以对基于最近邻距离分布的聚类做如下定义：

定义 8(聚类)设 DB 是数据库中点的集合，一个聚类 C 是 DB 的一个非空子集，且聚类 C 必须满足下列性质：1) 聚类 C 的最近邻距离集 DistSet__NN(C)服从所期望的分布；2) 聚类 C 是最大的，即，聚类 C 可以通过满足条件的邻居点进行扩展；3) 聚类 C 是互连的，即对于聚类 C 中的任意两个点之间总有一条路径相连。

定义 9(噪声，Noise)设 $C_1,\cdots,C_k$ 是数据库 DB 中的聚类，$i=1,\cdots,k,$，噪声(Noise)就是不属于任何一个聚类 C_i 的点集，即， $\text{Noise}=\{p\in DB|i:pC_i\}$。

4. 基于最近邻距离分布的空间聚类算法

4.1. SCAND 算法思想

SCAND 聚类是基于这样一个合理的假设：假设在数据空间的某个特定部分，一个聚类内部的点是均匀分布的。SCAND 是一个增量算法，即仅仅是在目前为止已经被处理的点中来检验是否把查询点安排到当前聚类中，而不是在整个聚类甚至整个数据库中来考虑 SCAND 通过邻居点来增量地扩展一个初始聚类，扩展的条件就是结果聚类的最近邻距离集的分布要服从期望的距离分布。

4.2. 相关定义

为了更好地描述算法，下面给出一些相关术语的定义：

定义 10(种子)一个种子点是当前聚类中刚加入的一个新成员。

定义 11(候选)一个候选是等待检验的点，也就是检验它是否可以加入到当前聚类中。若检验通过则把该点加入到当前聚类中；若检验未通过，则该点就不能加入到当前类中，我们把检验未通过的点称为失败的候选(Fail Candidate)。

定义 12(候选集)由多个候选组成的集合，一个聚类对应着一个候选集。

从以上定义可以看出，种子点就是当前聚类用于扩展的点，它是刚加入到该类中的一个新成员，一个种子点可以产生多个候选，一个聚类对应着一个候选集，候选集中满足条件的候选就可以加入到当前聚类中，聚类就是这样不断被扩展的，直到候选集为空时该聚类扩展结束。

那么，接下来的问题就是：对某个种子点而言，应该采用何种方式来产生候选才可以使提高算法的查询速度，使算法时间性能更优？下面将对这个问题进行讨论。

4.3. 算法优化

一个聚类的候选集可以通过区域查询来获得，使用诸如 R^*-树[8]这样的空间存取方法能够有效地支持区域查询。一次区域查询返回数据库中与特定的查询区域相交的所有对象，比如圆查询的结果将返回数据库中落入这个圆范围内的所有对象。本文中将采用圆查询来产生某个聚类的候选集。对于当前聚类 C 中的一个新成员 p 而言，用一个具有适合半径 r 的圆查询来检索点 p 的邻居，并把这些邻居点作为新的候选成员加入到候选集中。

半径 r 的选择是关键的，因为若半径选择过大，则会产生过多候选，那么检验的次数会过多，这样会降低算法的效率。那么半径 r 该如何选择呢？基于聚类中点同样分布的模型，选择半径的要求是，必须使该半径范围内存在至少一个属于该聚类的点。即，选择半径 r 的必要条件是：聚类 C 中的 N 个点中，不是所有点的最近邻距离都大于所选择的圆查询半径 r，设聚类 C 的面积为 A，C 中的元素个数为 N，则这个必要条件可表达为(5)式：

$$N\times P(\text{Dist\_\_NNC}(p)>r)<1 \tag{5}$$

$$P(\text{Dist\_\_NNC}(p)>r)=1-P(\text{Dist\_\_NNC}(p)\le r) \tag{6}$$

把 (4) 式 $F(x)=P(D\le x)=1-\left(1-\pi x^2/\text{Vol}(R)\right)^N$

代入(6)式中，有：

$$P\left(Dist\_\_NNC\left(p\right)>r\right)=1-P\left(Dist\_\_NNC\left(p\right)\le r\right)=\left(1-\pi r^2/Vol\left(R\right)\right)^N \quad (7)$$

把(12)式代入到(10)式中有：

$$\left(1-\pi r^2/A\right)^N<1/N \quad (8)$$

因此，我们有下式成立：

$$r>\sqrt{A/\pi\cdot\left(1-1/N^{1/N}\right)} \quad (9)$$

当插入一个新点 p 到聚类 C 中时，就立即考虑用点 p 作为种子进行扩展，即，要执行一次以 p 为圆心，r 为半径的圆查询，查询结果中没有被处理过的点被当作是新的候选加入到候选集中。聚类的扩展就是通过对候选集中点的检验来进行的，考虑候选集中的某个候选，若把它放于当前聚类 C 中后，最近邻距离集 DistSet__NN(C)仍服从期望分布，则把该点加入当前聚类 C 中。

SCAND 算法中，一次调用 Candidate_query (C,p)返回以 p 为圆心以 r 为半径的圆形区域内的所有点。诸如 R^*-树这样的空间存取方法能够高效地支持很多种区域查询。对于有 n 个点的数据库而言，R^*-树的高度为 logn，所以一次区域查询的平均时间复杂度为 O(logn)。对于数据库中的每一个点，我们最多进行一次区域查询，因此，SCAND 算法的平均时间复杂度为 O(nlogn)。

4.4. 检验候选

本算法中可采用文献[9]中的 χ^2-检验来对候选进行检验，检验当把某个候选插入到当前聚类 C 中后，被扩展了的聚类 C 的最近邻距离集 DistSet__NN (C)是否还服从所期望的距离分布。

之前提到过，SCAND 是一种增量的方法，这就意味着所产生和所检验的候选点的顺序对聚类结果是有影响的，因此，检验候选的顺序也是很关键的。我们把第一次检验失败的候选称为“失败的候选(Fail Candidate)”。对于失败的候选而言，可能存在这样一种情况，之前检验为失败的候选，最后却服从经过扩展后的聚类中的距离分布。为了将检验候选的顺序对聚类所产生的影响减至最小，SCAND 算法考虑了下面特征：对于当前聚类扩展失败了的候选，我们暂不丢弃，先把它们保存起来。

综上所述，检验候选的步骤是：首先，通过候选来扩展当前聚类，然后，采用 χ^2-检验来进行假设检验，检验这个被扩展了的聚类的最近邻距离集是否服从所期望的分布。

4.5. 优化的 SCAND 算法

优化后的 SCAND 算法描述如表 1 所示，算法流程如图 2 所示。

5. 算法评价

在 2.3 节中我们提到了对大型空间数据库上的聚类算法所提出的两个关键要求，一是能够发现任意形状的聚类；二是在大型数据库上效率好。已有的那些较有名的聚类算法中，CLARANS 算法是第一个专门针对大型空间数据库所提出的算法，但该算法只能处理凸面形状或球状聚类，即不能发现任意形状的聚类。DBSCAN 算法能在具有噪声的空间数据库中发现任意形状的聚类，但是 DBSCAN 算法需要输入两个参数 Eps 及 MinPts，算法对于参数非常敏感,而且这两个参数通常是很难确定。本文中的 SCAND 算法最大的优点是不需要输入参数及能够发现任意形状的聚类。因此，在下面的有效性和有用性评价中，我们选择了 CLARANS 算法、DBSCAN 算法来和 SCAND 算法进行比较分析。

Table 1. SCAND algorithm description
表 1. SCAND 算法描述

输入：未被聚类的具有 N 个对象点的队列 SetofPoint 输出：聚类{$C_1,C_2,\cdots$}
step1：判断对象集 SetofPoint 是否为空，如果为空则转到 step6；否则依次取出队列 SetofPoint 中点 Point，判断 Point 是否已被分类，如果是则转到 step1。
Step2：为点 Point 创建一个新类 C_i，并寻找点 Point 的至少 49 个最近邻加入到该类中，计算类 C_i 的最近邻距离分布函数 F(x)。
Step3：依次把 C_i 中每一个点都作为类扩张种子 seed，转到 Step4；如果 C_i 中所有的点都被遍历过，则转到 step5。
Step4：查询以 seed 为圆心 r 为半径的圆形区域内的所有点，把未分类的点加入到 C_i 的候选近邻队列 Candidate(C_i)中，转到 Step3。
Step5：取出队列 Candidate(C_i)中对象 p 加入到类 C_i 中，判断加入新邻居 p 后类 C_i 的最近邻距离集 DistSet__NN(C_i)是否服从期望分布 F(x)，如果是，就把点 p 作为扩张种子 seed:=p，转到 step4；否则把点 p 从类 C_i 中删除，并把点 p 插入到失败候选队列 Fail-Candidate 中。依次取出队列 Candidate(C_i)中每个对象进行考察，直到 Candidate(C_i)为空时转到 step1。
Step6：程序终止。

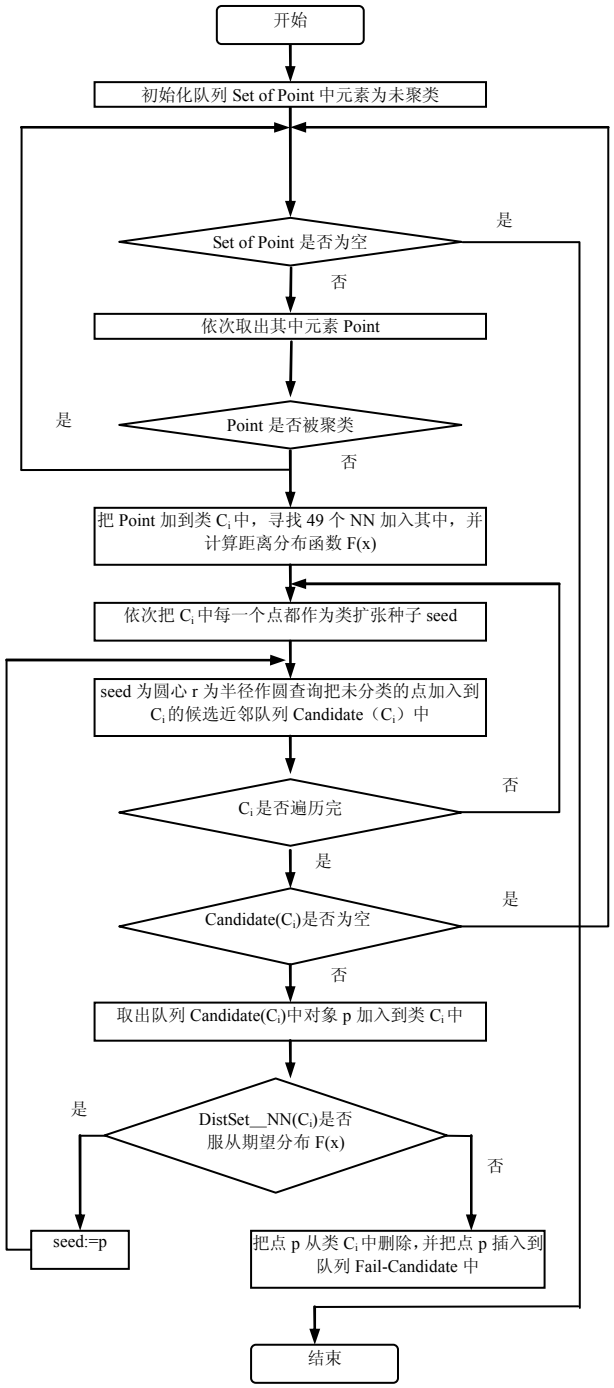

Figure 2. Flow char of SCAND
图 2. SCAND 算法流程图

5.1. 能发现任意形状的聚类

空间数据库中的聚类可能是多种多样的，因此，能够发现任意形状的聚类是决定聚类质量的一个重要因素。我们实验的模拟数据如下：1) 假设如图 3 所示的三种聚类形状；2) 针对三个形状分别合成 500 个、200 个及 150 个均匀分布的点；3) 在数据库中插入 100 个噪声点。SCAND 算法的聚类结果如图 3 所示，不同的聚类用不同的符号表示，噪声用“+”号表示。

结果表明，SCAND 算法的聚类结果和合成数据情况基本一致，只有少数边界点被聚到了错误的类中。DBSCAN 的聚类结果和 SCAND 结果很相似，在此就不再做描述。而 CLARANS 算法处理如图 3 中的任意形状聚类时出现了不正确的聚类结果。

5.2. 算法效率

我们在模拟数据集上对 SCAND、DBSCAN 及 CLARANS 算法效率进行比较，为了保证运行时间，我们把数据量控制在 5000 至 25,000 之间，所有试验在 Pentium IV 2.4 GHz，512 MB 的 PC 机上用 Visual C^{++}实现，三个算法的运行时间如表 2 所示。

由表 2 可以看出，SCAND 算法的运行时间大概为 DBSCAN 运行时间的两倍，但 CLARANS 算法运行时间却是 SCAND 算法的大约 60 倍。

6. 结束语

空间数据挖掘对应用于大型空间数据库中的聚类算法提出以下要求：一是能够发现任意形状的聚类；二是在大型数据库上效率好。针对已有空间聚类算法的不足，提出一种新的聚类算法，基于最近邻距离

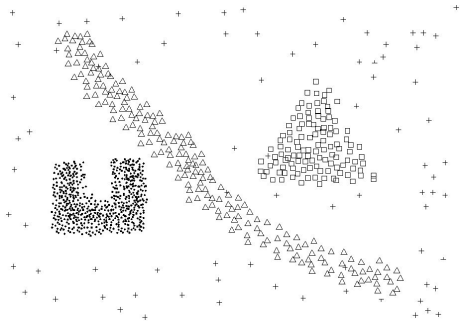

Figure 3. SCAND discover clusters of arbitrary shape
图 3. SCAND 发现任意形状的聚类

Table 2. Run time comparison (sec)
表 2. 运行时间比较(秒)

数据点	5000	10,000	15,000	20,000	25,000
SCAND	82	150	220	350	458
DBSCAN	50	80	135	180	220
CLARANS	5020	22,600	64,700	138,000	250,800

分布的空间聚类算法——SCAND 算法。SCAND 算法的思想是基于一个聚类中的点到它们的最近邻之间的距离，并分析该聚类中这些距离的分布。本文中的分析是基于这样一个假设，即，假设在数据空间的某个特定部分，一个聚类内部的点是同一分布的。SCAND 算法通过邻居点增量地扩展一个初始聚类，扩展条件是结果聚类的最近邻距离分布依然服从期望分布。邻居点的检索是基于圆查询，诸如 R^*-树这样的空间存取方法能够高效地支持圆查询。

实验分析表明，与划分算法 CLARANS 相比，SCAND 动态地决定聚类数目且能发现任意形状的聚类；与密度聚类算法 DBSCAN 相比，SCAND 算法无输入参数。而且 SCAND 算法具有较好的伸缩性，即对于大型数据库是有效的。

7. 致谢

本文获得云南省教育厅科学研究基金资助项目“空间数据挖掘技术在土地利用中的应用研究”(项目编号：2011Y074)的支持，在此深表感谢。

参考文献 (References)

[1] 李德仁, 王树良, 李德毅等. 空间数据挖掘理论与应用[M]. 北京: 科学出版社, 2008: 26-28.

[2] J. Han, M. Kamber. Spatial clustering methods in data mining: A survey. Geographic Data Mining and Knowledge Discovery, 2001: 211-221.

[3] R. T. Ng, J. Han. Efficient and effective clustering method for spatial data mining. Santiago: Proceedings of the 20th International Con- ference on Very Large Data Bases, 1994: 144-155.

[4] M. Ester, H. P. Kriegel, J. Sander, et al. A densidy-based algorithm for discovering clusters in large spatial databases with noise. Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining. AAAI/MIT Press, 1996: 225-231.

[5] 周毅. 基于 Realms 的空间分析数据库系统 SADBS 的实现[D]. 南京航空航天大学, 2001.

[6] 刘国俭. 基于 SADBS 的空间聚类分析系统 SDCAS 的研究与实现[D]. 南京航空航天大学, 2005.

[7] 李新运, 郑新奇, 闫弘文. 坐标与属性一体化的空间聚类方法研究[J]. 地理与地理信息科学, 2004, 20(2): 38-40.

[8] N. Bechmann. The R^*-tree: An efficient and robust access method for points and rectangle. Atlantic: Proceedings of ACM SIGMOD Internationa1 Conference on Management of Data. ACM Press, 1990: 322-331.

[9] 盛骤, 谢式千. 概率论与数理统计(第四版)[M]. 北京: 高等教育出版社, 2009: 80-89.

