

文章编号:1001-9081(2005)08-1818-03

## 一种改进的带障碍的基于密度和网格的聚类算法

严 馨<sup>1</sup>,周丽华<sup>2</sup>,陈克平<sup>2</sup>,徐广义<sup>3</sup>

(1. 昆明理工大学 计算机科学技术系,云南 昆明 650051;

2. 云南大学 计算机科学与工程系,云南 昆明 650091; 3. 南天电子信息股份有限公司,云南 昆明 650041)

(kg\_yanxin@sina.com)

**摘 要:**提出了一个改进的带障碍的网格弥散聚类算法 DCellO1;以网格为基础,将基于密度的聚类算法与图形学种子填充算法相结合。该算法能进行任意形状的带障碍聚类,并且在对象分布不均匀时也能获得较好的聚类结果。实验证明了该算法的有效性与优越性。

**关键词:**聚类;网格;密度;障碍

**中图分类号:** TP18;TP301.6 **文献标识码:** A

## Improved clustering algorithm based on density and grid in the presence of obstacles

YAN Xin<sup>1</sup>, ZHOU Li-hua<sup>2</sup>, CHEN Ke-ping<sup>2</sup>, XU Guang-yi<sup>3</sup>

(1. Department of Computer Science and Technology, Kunming University of Science and Technology, Kunming Yunnan 650051, China;

2. Department of Computer Science and Engineering, Yunnan University, Kunming Yunnan 650091, China;

3. Nantian Electronics Information Corporation Limited, Kunming Yunnan 650041, China)

**Abstract:** An improved grid diffusant clustering algorithm in the presence of obstacles called DCellO1 was proposed. Based on grid, it combined density-based clustering algorithm with seed-filling algorithm of graphics. It could construct arbitrary shape clustering in the presence of obstacles, and could obtain good clustering results when the objects distributed unevenly. The experiments prove the superiority and effectiveness of DCellO1.

**Key words:** clustering; grid; density; obstacles

## 0 引言

聚类是对物理的或抽象的对象集合分组的过程,聚类分析在统计学、数据挖掘等方面的研究较为活跃,出现了很多有效的、可扩展性好的算法,包括:分类算法<sup>[1,2]</sup>、层次算法<sup>[3]</sup>、基于密度的算法<sup>[4]</sup>等。但这些算法未考虑有障碍存在的情况。Han Jiawei 在 1999 年提出一种带障碍的聚类问题:如需要在一个地区确定一些加油站或 ATM,就需要考虑该地区河流、高速公路等约束情况,为聚类算法的研究开辟了一个新方向。Han Jiawei 在这篇文章中提出了一个带障碍的聚类算法 COD-CLARANS<sup>[5]</sup>,它能有效地处理这种带障碍的聚类问题,但存在占用内存过多及障碍的形状影响算法效率等问题。

文献[6]提出了网格弥散带障碍的聚类算法 DcellO,主要是以一个随机抽取的核心网格为基础,不断寻找邻接网格,向四周扩散;然后,再以找到的包含对象的网格作为核心网格,继续向四周弥散。DCellO 算法是基于网格的,相当于将对象进行了一次微聚类,可显著提高算法的执行效率;可以方便地处理光栅图像,通过颜色易于区分对象与障碍物;易于并行化。

DCellO 算法将基于网格与基于密度的算法相结合,因计算范围被限制在局部空间,可以削减计算量。但 DCellO 算法规定:某一聚类是指一个包含对象的网格集合,其成员间带障碍的距离小于某一聚类半径  $\varepsilon$ ,而该集合的对象总数大于某

一阈值  $minPts$ ,则如图 1 所示的对象集合会归并为一个聚类,本应划分为三个甚至更多的聚类成为一个聚类,原因是 DCellO 算法不是严格的基于密度的算法,它考虑相互距离(带障碍)小于  $\varepsilon$  的对象的总数,对于单个对象不管它的  $\varepsilon$  邻域内的对象数目是多少,均将之并入  $S_i$  的集合,这样将会通过一些孤立点将多个聚类并为一个,得到不正确的聚类结果。在对象分布不均匀时此问题尤为突出。



图 1 数据点分布

本文提出一个改进的算法 DCellO1,沿用了 DCellO 算法基于网格、利用图形学种子填充算法及动态规划中单源点最短路径算法等高效率地计算两个对象间的带障碍距离,对 DCellO 算法中核心网格的概念

及聚类模型提出了改进,有效地克服了 DCellO 算法上述缺点,根据密度进行任意形状的带障碍聚类,在对象分布的各种情况下,能有效识别例外与背景噪声,在与 DCellO 算法时间数量级相同的情况下,大大改善了聚类质量。

## 1 原算法主要思想及主要概念

**定义 1** 核心网格(Kerneled Grid):规定用于计算带障碍距离的起始网格,即计算带障碍距离时的初始网格。

**定义 2** 8 邻接网格(Adjacent Grid):如图 2 所示,8 邻接

收稿日期:2005-01-26;修订日期:2005-04-22

基金项目:国家自然科学基金资助项目(60463004);云南省教育厅科学研究基金资助项目(03Y173D)

作者简介:严馨(1969-),女,重庆人,副教授,硕士,主要研究方向:数据挖掘、数据仓库;周丽华(1968-),女,云南丽江人,副教授,硕士,主要研究方向:数据挖掘、数据仓库;陈克平(1970-),男,福建人,硕士研究生,主要研究方向:数据挖掘、数据仓库;徐广义(1965-),男,云南宣威人,高级工程师,硕士,主要研究方向:数据仓库、数据挖掘。

网格指的是与指定网格毗邻的8个网格A..H。

**定义3** 不可见网格 (Non-Visual Grid): 如图3所示, 将A与B的连线光栅化后, 如果在这些离散的光栅化网格中存在有障碍, 则称A与B相互不可见。



图2 8邻接网格

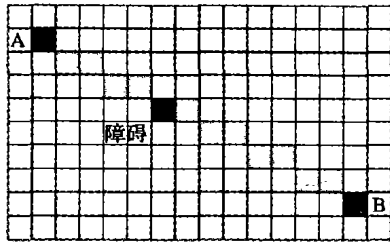


图3 不可见网格

**定义4** 带障碍距离 (Distance in the Presence of Obstacle, DPO): 指待计算网格与核心网格间的以网格中心为基准的最短距离。DCello算法以可见网格为基础, 将待计算的网格与核心网格间用两两相互可见的网格连接起来, 并使可见网格间的距离之和为最小, 即为带障碍的距离。如图3, A与核心网格O间带障碍的距离为 $AB + BC + CO$ 。

**定义5** 离核心网格最近的可见网格 (Visual Grid Closet to O, VGCO): O为一个核心网格, A为一个待计算的网格, 若 $B = \{X | \forall X \in \text{网格空间}, \exists \text{dist}(X, O) \text{ 为最小值且 } \text{vis}(A, X) = 1\}$ , 则B为A的离O最近的可见网格, 如图4。

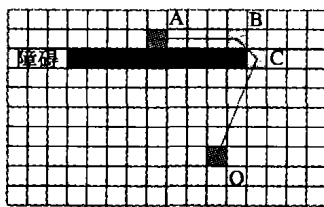


图4 带障碍的距离

**定义6** 聚类的下限阈值 (Lower Bound): 如果聚类中对象数量小于此值, 则当作例外。

**定义7** 聚类半径 (Clusterring Rradius): 待计算网格与核心网格间的带障碍距离的最大允许值, 即一个聚类内部, 任意两个对象间允许的最大带障碍距离。

DCello算法的基本步骤如下: 1) 划分网格, 将对象微聚类映射到网格中; 2) 从包含对象的网格中随机选取一个作为核心网格; 3) 在聚类半径内计算: 寻找核心网格的邻接网格, 在计算其邻接网格与核心网格的DPO的基础上, 不断向外弥散, 找到其聚类半径之内所有DPO小于聚类半径的对象作为候选核心网格集合; 4) 分别以候选核心网格集合中的对象作为核心网格出发, 继续向四周弥散, 重复3), 直到不能再找到候选核心网格时为止; 5) 将所有寻找到的包含对象的网格计数, 如大于聚类下限阈值, 则归并为一个聚类, 否则视为例外; 6) 从S中随机抽取一个包含对象的网格, 重复3)~5), 进行下一轮聚类, 直到S中所有包含的对象全部聚为聚类或例外时为止。

DCello算法计算某网格与核心网格间的DPO, 即GetCurrentDist子算法, 利用了通用Bresenham算法, 将两网格间的连线光栅化, 通过判断光栅化网格中是否包含障碍, 来决定两网格是否可见。具体计算距离时, 将单源最短路径算法与Bresenham算法相结合, 以邻接网格为基础, 迭代计算邻接网格与核心网格间的DPO。首先, 以核心网格O为计算DPO的起点, 先在主程序中将核心网格的二元组设定为(0,0), 即O的VGCO是它本身, 与核心网格距离为零, 然后在GetCurrentDist中对被计算的网格A寻找其已计算过的邻接网格。

A到O的距离 =

$$\begin{cases} A \text{ 与 } A \text{ 的邻接网格的 VGCO 的距离} + \text{此 VGCO 到 } O \text{ 的距离} \\ \quad (\text{如 } A \text{ 与其邻接网格的 VGCO 可见}) \\ A \text{ 到其邻接网格的距离} + \text{其邻接网格到它的 VGCO 的距离} + \\ \quad \text{此 VGCO 到 } O \text{ 的距离} \\ \quad (\text{如 } A \text{ 与其邻接网格的 VGCO 不可见}) \end{cases}$$

## 2 改进算法 DCellO1

针对DCello算法未真正考虑到密度对聚类结果的影响, 我们提出一个改进的DCellO1算法, 其基本思想是: 以一个随机抽取的包含对象的网格为基础, 寻找其邻接网格, 向四周扩散, 得到起始网格 $\epsilon$ 邻域(带障碍)内的所有对象, 即与起始网格直接密度相连的对象, 再从这些对象出发利用邻接网格得到与它们直接密度相连的对象, 如此重复, 直到找到所有与O密度相连的对象即聚为一类。以下是算法用到的基本概念。

**定义8** 核心网格 (Kerneled grid): 一个包含对象的网格O, 其 $\epsilon$ 邻域内包含的对象的数目须大于下限阈值 $\text{minPts}$ 。

**定义9** 聚类半径 $\epsilon$  (Clustering Radius): 指待计算网格与核心网格间DPO最大值。

**定义10** 下限阈值 (Lower Bound)  $\text{minPts}$ : 一个包含对象的网格O, 与O的带障碍距离 $\text{DPO} < \epsilon$ 的对象的个数大于 $\text{minPts}$ , O才是核心网格。

**定义11** 非核心网格标志 $\text{tag}$ : 每一包含对象的网格均有一个标志 $\text{tag}$ , 如某网格不满足核心网格的条件, 则其标志 $\text{tag} = 1$ 。初始化时, 将所有包含对象的网格的标志置零。

其余概念除了用起始网格替换核心网格外, 类似于DCello算法中的定义。

DCello1算法的基本思想:

1) 划分网格, 将对象微聚类映射到网格中, 并记下每个网格中包含对象的数目。将所有包含对象的网格核心标志置为零, 表示其核心状态未知。

2) 随机选取一个包含对象且 $\text{tag} = 0$ 的网格O作为起始计算的网格。

3) 将寻找到的当前未计算过的邻接网格, 记为候选邻接网格, 累加器 $\text{sum}$ 置零。

4) 计算候选邻接网格与起始网格间的DPO。

5) 从候选网格中寻找与起始网格的距离小于聚类半径 $\epsilon$ 的网格, 将其并入待计算的网格集合中; 然后再从中寻找包含对象的网格, 如包含对象, 将此网格中包含对象的数目累加到 $\text{sum}$ 中, 并将其并入集合 $S_x$ ,  $S_x$ 是一个待检验的候选聚类集合。

6) 以上一步寻找到的待计算网格集合为基础, 从中继续寻找新的候补邻接网格集合, 重复5), 直到寻找到的邻接网格与起始网格O的距离均大于 $\epsilon$ 为止。

7) 如 $\text{sum} < \text{minPts}$ , 则起始网格不是核心点, 将O点的标志 $\text{tag}$ 置1; 如 $\text{sum} \geq \text{minPts}$ , 将 $S_x$ 中所有的对象并入候选起始网格集合 $\text{Subject}$ 中, 且将 $S_x$ 中所有对象从S中删去。 $S_x$ 中所有的对象并入第i个聚类集合 $S_i$ 中。

8) 将 $S_x$ 置空, 从 $\text{Subject}$ 中另选一个 $\text{tag} = 0$ 的对象出发, 重复3)~8), 直到 $\text{Subject}$ 为空为止。

9) 从S中另选一个 $\text{tag} = 0$ 的对象作为起始点, 进行第i+1轮聚类, 重复3)~8), 直到S中再也找不到 $\text{tag} = 0$ 的对象为止。此时S中剩下的所有 $\text{tag} = 1$ 的对象均为例外。

由此可见, DCellO1算法与DCello算法的主要不同之处: DCello算法将任意起始网格看作核心点, 通过邻接网格不断向外弥散、延伸, 在此过程中只要任意两对象的带障碍距离小于 $\epsilon$ 则并入集合 $S_i$ , 其间不考虑核心网格 $\epsilon$ 邻域内对象的个

数,只对  $S_i$  中对象总数计数,如大于  $\minPts$  则认为可归为一个聚类,表现在算法中:计数是在第 5) 步即认为第  $i$  轮聚类完成后进行,DCello 算法并未真正考虑密度,因为它只计数而不管聚类所占“面积”。DCello1 算法在寻找某一起始网格  $O$  的  $\varepsilon$  邻域内对象后马上对其计数,即在第 5), 6) 步完成后计数,从而判断出是否应将所有与  $O$  的 DPO 小于  $\varepsilon$  的对象归入聚类,有效地避免了将例外归入聚类及通过孤立点搭桥将多个聚类聚为一类的情况。伪码算法如下:

输入:  $S$ , 所有包含对象的网格集合;  $\varepsilon$ , 聚类半径;  $\minPts$ , 下限阈值输出: 聚类  $S_i$ , 例外 outlier

```

i = 1;
对 S 中所有元素置 tag = 0;          //所有对象核心点状态未知
while (S 中 tag = 0 的对象 isn't null)
{ c1 = extract_random_grid(S 中 tag = 0 的对象);
  Subject = {c1}; Sx = ∅;
  while (Subject 中 tag = 0 的对象 isn't null)
  { c2 = extract_random_grid(Subject 中 tag = 0 的对象); sum = 0;
    delete(Subject, c2);
    将区域 [c2.x - ε...c2.x + ε, c2.y - ε...c2.y + ε] ∈ 网格区间
    标为未计算, 并将网格 c2 标为已计算
    O = c2; Sabut = {c2}; Stemp = ∅;
    while (Sabut isn't null)
    { 对 Sabut 中每一 grid A
      判断 A 的 8 个邻接网格, 如果未计算过且不是障碍, 则将
      之并入 Stemp, delete(Sabut, A);
      while (Stemp isn't null)
      { c3 = extract_random_grid(Stemp); delete(Stemp, c3);
        E[c3] = Get_CurrentDist(c3, O) 并将 c3 标记为已计算
        DistCurrent = E[c3]. CurrentDistance + c3 与其 VGCO
        间
        欧式距离?
        if (DistCurrent < ε)
        { Sabut = Sabut ∪ {c3};
          if (c3 in S)
          { Sx = Sx ∪ {c3}; sum ++;
            }
          }
        }
      }
    }
    if (sum < minPts) c2.tag = 1;
    else
    { 将 Sx 中所有对象并入 Subject 中, 并将 Sx 中所有对象从
      S 中删去, 将 Sx 中所有元素并入  $S_i$  中
    }
    Sx = ∅;
  }
  i ++;          //进行读 i + 1 轮聚类
}

```

$S$  中剩余的对象为例外

由此可见,DCello 算法从某一起始网格出发,向外弥散,只要任意两个对象间  $DPO < \varepsilon$  则并入某一集合  $S_i$ , 如  $S_i$  所含对象数目  $> \minPts$  则形成一个聚类,否则  $S_i$  中所有对象为噪声。DCello1 算法沿用了其网格弥散及采用 Bresenham 及动态规划中单源最短路径算法计算带障碍距离,真正考虑了密度的概念,从任一起始网格  $O$  出发,通过计算  $O$  的  $\varepsilon$  邻域内包含的对象数目求得与  $O$  直接密度相连的点,进而求得所有与  $O$  密度相连的点,即得到一个关于  $\varepsilon$  和  $\minPts$  的聚类;如  $O$  不是核心点,就另选一个核心状态未知的对象做同样的处理,从而避免将噪声并入聚类及通过噪声对象得到不正确的聚类结果。

### 3 DCellO1 算法分析

#### 3.1 时间性能分析

本算法将基于密度与网格的算法结合起来,只在  $\varepsilon$  邻域内计算 DPO,大幅降低了计算量;利用邻接网格加速算法执行,利用 Bresenham 及单源最短路径算法快速计算带障碍距离 DPO。如以某一个网格  $O$  为起始网格,  $\varepsilon$  邻域内无障碍(此为最坏情况),面积为  $\pi\varepsilon^2$ ,如网格边长为 1,则共有  $\pi\varepsilon^2$  个网格,需调用 Bresenham 算法  $\pi\varepsilon^2$  次,而 Bresenham 算法时间复杂度为  $O(n)$ ,故从  $O$  出发求得与  $O$  的带障碍距离小于  $\varepsilon$  的对象所需的时间复杂度为  $\pi\varepsilon^2$ ,如得到的  $\varepsilon$  邻域内对象数目小于下限阈值  $\minPts$ ,则  $O$  不是一个核心网格,将它的 tag 置 1;此时另选一个 tag  $\neq 1$  的网格  $O'$  作为起始网格计算,如  $O'$  的  $\varepsilon$  邻域内对象数目大于  $\minPts$ ,且  $O$  也在  $O'$  的  $\varepsilon$  邻域内,即  $O$  与  $O'$  直接密度相连,由于已知  $O$ . tag = 1,则不需要再从  $O$  出发求与  $O$  直接密度相连的对象,只需求出其余与  $O'$  密度相连的对象,亦即包含对象的网格作为起始网格进行  $\varepsilon$  邻域的搜索仅为一次,故 DCellO1 算法总的时间复杂度为  $O(n) = n\varepsilon^2$ ,在  $\varepsilon$  固定的情况下与  $n$  成线性关系,与 DCellO 算法的时间性能在数量级上相同。

#### 3.2 空间复杂度分析

如网格空间大小为  $M \times N$ ,则需要用一标志矩阵存放每一网格的状态,还需在计算距离时使用二元组矩阵存储网格的 VGCO 及距离,另外还需要一个单链表存放包含对象的网格及其数量、标志,故空间复杂度为  $O(MN + n)$ 。

#### 3.3 实验结果

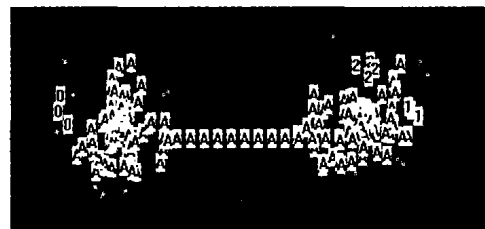


图 5 DCellO 算法聚类结果

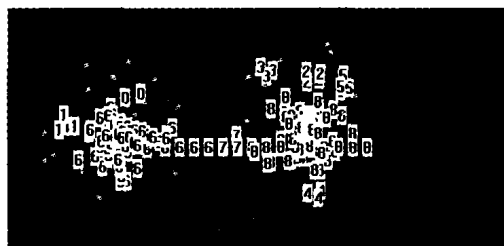


图 6 DCellO1 算法聚类结果

聚类质量的测试详见图 5、图 6,位于相同聚类中的点用相同的数字或字母代表,例外用单点表示,数据为随机生成的二维正态分布的相同点簇。由图可知,在存在孤立点且孤立点距两个类边界点的距离均小于  $\varepsilon$  的情况下,DCello 算法将之归为一个聚类,而 DCellO1 算法能根据密度识别孤立点,避免不恰当的聚类结果。由于 DCellO1 算法对每个起始网格均要对其  $\varepsilon$  邻域内对象计数,其间多次涉及到集合的归并、删除,故两个算法尽管时间复杂度数量级相同,但从系数来说,DCello1 算法的更大。

### 4 结语

DCello 算法是用基于网格的算法,实现有一定约束条件即在有山川、河流等障碍存在情况下的聚类。本文针对

(下转第 1823 页)

结果中可以看到,选取的特征项太多也会对属性降维和规则降维产生不好的影响,我们认为这是由于某些特征项引入了一些干扰信息。

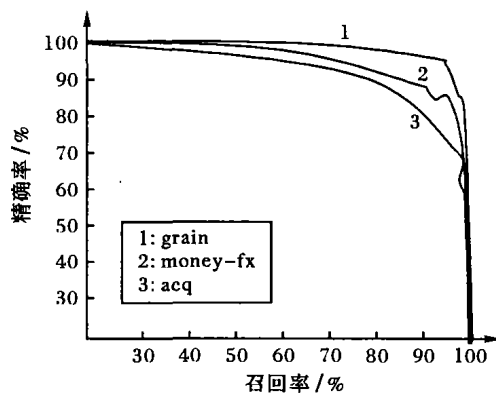


图1 类别 acq, Money-fx, grain 的分类精确率与召回率

表1 类别 grain 在不同特征项维数下的属性降维度和样本降维度

	100	500	1000	2000
$R_a$	0.785	0.681	0.632	0.574
$R_s$	0.55	0.582	0.617	0.630

## 4 结语

本文提出的方法总体而言进行了三次维度消减:词频离散化降维、条件属性(特征项)降维和分类规则降维。实验结果说明 DSM 方法可以应用于文本分类,并能获得相当好的结果。它能够同时消减文集中文档特征属性的维度以及分类规则所需的样本维度,从而提高分类的效率。然而,由于我们只针对 Reuter-21578 的三个子类别进行了实验,有关维度的选择阈值如何确定、相似度和重要度如何折中等很多问题尚需要深入探讨。另外,我们还将在以后的工作中引入基于 DSM

的增量式学习方法、动态修订和产生规则,进一步改善分类器的性能。

### 参考文献:

- [1] YANG Y, PEDERSEN JP. A Comparative Study on Feature Selection in Text Categorization[A]. Proceedings of the Fourteenth International Conference on Machine Learning[C]. Tennessee, USA: Vanderbilt University, 1997.
- [2] 孙建军, 成颖, 丁芹, 等. 信息检索技术[M]. 北京: 科学出版社, 2004. 168 - 170.
- [3] DUIN RPW, LOOG M, HAEB - UMBACH R. Multi - class Linear Feature Extraction by Nonlinear PCA[A]. Proceedings of 15th International Conference on Pattern Recognition[C]. Barcelona, Spain: IEEE Computer Science Press, 2000. 398 - 401.
- [4] NGUYEN, SON H. Scalable classification method based on rough sets[A]. Rough Sets and Current Trends in Computing, Lecture Notes in Computer Science[C]. PA, USA: Springer, 2002. 433 - 440.
- [5] 夏德麟, 晏蒲柳. 一种新的信息系统知识约简方法——DSM 方法[D]. 武汉: 武汉大学大学电子信息学院, 2001.
- [6] ZHOU JG, XIA DL, YAN PL. Incremental Machine Learning Theorem and Algorithm Based on DSM Method[A]. Proceedings of the Third International Conference on Machine Learning and Cybernetics[C]. Shanghai: IEEE, 2004. 2202 - 2207.
- [7] AIZAWA A. The Feature Quantity: An Information Theoretic Perspective of TfIdf-like Measures[A]. Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval[C]. Tarrytown, NY, USA: Pergamon Press, Inc, 2000. 104 - 111.
- [8] Reuters - 21578 Text Categorization Collection [DB/OL]. <http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html>, 2004.
- [9] 江昊, 晏蒲柳. 基于 DSM 的数据约简[J]. 武汉大学学报(理学版), 2003, 49(3): 378 - 382.

(上接第 1820 页)

DCello 算法不能根据密度识别出孤立点及通过孤立点搭桥将若干个聚类聚为一类的不足,提出了改进算法 DCello1:沿用了 DCello 算法邻接网格向四周弥散及使用 Bresenham 及最短路径算法计算带障碍距离的思想,引入核心网格、聚类半径、下限域值及密度相连的概念,真正考虑了密度对聚类的影响,能够处理任意形状的对象、任意形状的障碍,可方便处理光栅图像,易于并行化,在对象分布不均匀时聚类效果尤其好,其时间复杂度与空间复杂度在数量级上与 DCello 算法相同。

### 参考文献:

- [1] MACQUEEN J. Some methods for classification and analysis of multivariate observations[A]. 5th Berkeley symposium on mathematics, statistics and probability[C], 1967, 1. 281 - 296.
- [2] KAUFMAN L, ROUSSEEUW P. Finding groups in data: an introduction to cluster analysis[M]. New York: John Wiley & Sons, 1990.
- [3] KARYPIS G, HAN E - H, KUMAR V. Chameleon: a hierarchical clustering algorithm using dynamic modeling[J]. Computer, 1999, 32(1): 32 - 68.
- [4] ESTER M, KRIEGER H-P, SANDER J, et al. A density-based algorithm for discovering clusters in large spatial databases with noise [A]. Second International Conference on Knowledge Discovery and Data Mining [C]. Portland: AAAI Press, 1996. 226 - 231.
- [5] TUNG AKH, HOU J, HAN J. Spatial Clustering in the Presence of Obstacles[A]. Proceedings of 2001 International Conference on Data Engineering[C], 2001.
- [6] 陈克平, 周丽华, 王丽珍, 等. DCellO——网格弥散聚类算法[J]. 计算机研究与发展, 2004, 41(增刊): 205 - 212.
- [7] HAN J, KAMBER M. 数据挖掘——概念与技术(影印版)[M]. 范明, 孟小峰, 译. 北京: 机械工业出版社, 2001.
- [8] NG R, HAN J. Efficient and effective clustering methods for spatial data mining[A]. Twentieth International Conference on Very Large Databases[C]. Morgan Kaufmann, 1994. 144 - 155.
- [9] NG R, HAN J. Clarans: A method for clustering objects for spatial data mining[J]. IEEE Transactions on Knowledge and Data Engineering, 2002, 14(5): 1003 - 1016.
- [10] ZHANG T, RAMAKRISHNAN R, LIVNY M. Birch: An efficient clustering method for very large databases[A]. ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery[C]. Montreal, 1996. 103 - 114.
- [11] GUHA S, RASTOGI R, SHIM K. Cure: an efficient clustering algorithm for large databases[A]. ACM SIGMOD International Conference on the Management of Data[C]. Seattle, WA, USA, 1998. 73 - 84.
- [12] KAUFMAN L, ROUSSEEUW P. Finding Groups in Data: an Introduction to Cluster Analysis[M]. John Wiley & Sons, 1990.
- [13] WANG W, YANG J, MUNTZ R. STING: A statistical information grid approach to spatial data mining[A]. Proceedings of International Conference on Very Large Data Bases (VLDB'97)[C]. Athens, Greece, 1997. 186 - 195.
- [14] BRESENHAM JE. Algorithm for computer control of a digital plotter[J]. IBM Systems Journal, 1965, 4(1): 25 - 30.