# 空间离群点的模型与跳跃取样查找算法

黄添强[1),2)] 秦小麟[1)] 王钦敏[3)]

[1)](南京航空航天大学计算机科学与工程系，南京 210016)

[2)](福建师范大学数学与计算机学院计算机科学与工程系，福州 350007) [3)](福州大学空间信息工程研究中心，福州 350002)

**摘 要** 目前无论是查找一般的离群点，还是空间离群点，都强调非空间属性的偏离，但在图像处理、基于位置的服务等许多应用领域，空间与非空间属性要综合考虑。为此，首先提出了一个综合考虑两者的空间离群点定义，然后提出了一种新的基于密度的空间离群点查找方法——基于密度的跳跃取样空间离群点查找算法 DBSODLS。由于已有的基于密度的离群点查找方法对每一点都要求进行邻域查询计算，故查找效率低，而该算法由于可充分利用已知的邻居信息，即不必计算所有点的邻域，从而能快速找到空间离群点。分析与试验结果表明，该算法时间性能明显优于目前已有的基于密度的算法。

**关键词** 数据挖掘 空间离群点 空间数据库 影响域

**中图法分类号**：TP311.13 TP392 TP18 **文献标识码**：A **文章编号**：1006-8961(2006)09-1230-07

# Spatial Outlier Model and Detection Algorithm with Leapingly Sampling

HUANG Tian-qiang[1),2)], QIN Xiao-lin[1)], WANG Qin-min[3)]

[1)](*Department of Computer Science and Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing* 210016)

[2)](*Department of Computer Science and Engineering, College of Mathematics and Computer Science, Fujian Normal University, Fuzhou* 350007)

[3)](*Spatial Information Research Center in Fujian Province, Fuzhou University, Fuzhou* 350002)

**Abstract** Existing work in outlier detection emphasizes the deviation of non-spatial attribute not only in outlier detecting in statistical database but also in spatial outlier detecting in spatial database. However, both spatial and non-spatial attributes must be synthetically considered in many applications, such as image processing, position-based service. We defined outlier in respect of taking account of both spatial and non-spatial attributes and proposed a new density-based spatial outlier detecting approach with leapingly sampling(DBSODLS). Existing density-based outlier detection approaches must calculate neighborhoods of every object, which are time-consuming. This method makes the best of neighbor information that have been detected, leapingly selects the next object, but not every object, which reduces many neighborhood queries. Theoretical comparison shows this method is better than other density-based methods in efficiency, and the experimental results also show that the approach outperforms the existing density-based methods in efficiency.

**Keyword** data mining; spatial outliers; spatial database; impact neighborhood

## 1 引 言

当今，数据挖掘与知识发现的研究主要集中于发现常规模式或频繁的事件，但在一些应用中，异常的模式、稀有的事件比常规的模式、频繁的事件更有价值。离群点查找就是旨在发现偏离常规模式的小部分异常模式。离群点的查找在许多的应用领域有重要的应用，如 影像处理、信用卡欺诈识别、社会热点识别、股票分析、金融审计、恶劣天气预报等。

目前存在多种的离群点定义，如 Hawkin 认为，离群点是严重偏离其他对象的观察点，以至于让人怀疑它是由不同的机制产生的[1]，而且，离群点的定义随着用户需求与应用领域的变化而变化[2]。

基金项目：国家“863”计划资助项目(2001AA633010-04)；国家自然科学基金项目(49971063)；江苏省自然科学基金项目(BK2001045)
收稿日期：2005-01-31；改回日期：2005-06-07

第一作者简介：黄添强(1971～ )，男。2006 年获南京航空航天大学计算机科学与工程系博士学位，现为福建师范大学讲师。研究方向为空间数据挖掘。E-mail：tianqianghuang@163.com

已有的多维离群点的查找方法可分为均一多维的方法与二分多维的方法两类[3]。其中均一多维的方法由于不区分空间维(地学空间)与非空间维,故不适合用来查找空间离群点;二分多维的方法虽可区分空间与非空间属性,可适合用来查找空间离群点,但是,已有的这种方法在定义空间离群点时仍然强调非空间属性的偏离——“空间离群点是指非空间属性与空间邻域中其他的对象明显不同的空间对象,它虽是局部不稳定的,且对邻域其他的点具有极端的值,但是对总体来说并不异常”[3,4]。

在某些应用领域,该领域专家要通过研究对象在空间属性与非空间属性两方面与其他对象的关系来识别离群点。例如 在遥感影像处理中,当查找某一类型植被的空间分布异常时,其植被类型是非空间属性,而植被的分布位置则是空间属性。又如 政府要调查中等收入住户的分布时,其收入是非空间属性,而住户的位置则属于空间属性。在这些例子中,空间与非空间属性要综合考虑。图 1 表示了不同非空间属性的两类对象,圆圈与圆点各代表一种属性的对象。如果不考虑非空间属性,则图中所有的对象就是一个聚类,但在考虑非空间属性时,将会有不同的结果。如对于圆点的对象,它们有两个聚类 $C_1$、$C_2$ 与两个离群点 $a$、$b$。

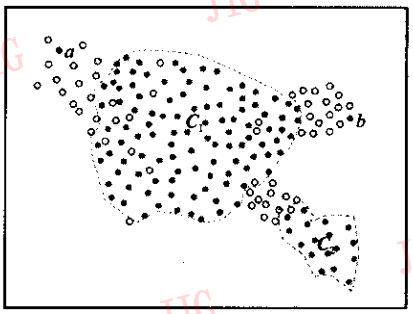


图 1　一个解析性例子

Fig. 1　An illumination example

上述例子说明,只有通过综合考虑空间与非空间属性来定义离群点,才可以为此类问题提供正确的模型,也才能区分正确的空间聚类与空间离群点。

已有的均一多维的方法不区分属性维(非空间维)与空间维。这种方法面向统计数据库,它是把各种多维数据看成同一度量的多维空间,同时用所有的维来定义邻域,并进行比较。Knorr 等提出了一个基于距离的离群点的概念[5,6],以后又有 $k$-最邻近邻域的基于距离的方法以及 Top-k 离群点等类似的概念[7]。Arning 等提出了利用基于偏离的技术来查找离群点。这种方法首先检查对象的特征,然后将偏离这些特征的对象作为离群点[8]。Breunig 等提出了局部离群点的概念 LOF(local outlier factor)[9],它是通过数据空间的所有维度来计算对象的距离,进而计算对象的可达密度,最后通过局部的偏离度来判断离群点。在计算几何中,有人提出了基于深度的方法,即把对象组织成不同深度的凸面来查找离群点[10,11]。有人用聚类的方法来发现离群点,如 DBSCAN[12](density based spatial clustering of applications with noise),ROCK[13](robust clustering using links),C2P[14](clustering based on closest pairs)等方法,聚类后,除了聚类,剩下的就是离群点。He Zeng-you 等提出了一种用基于聚类的局部离群点因子的方法来发现聚类[15]。He Zeng-you 等还提出了一个分类离群点的概念[16],他们认为离群点应该考虑分类标记。Aggarwal 与 Yu 用投影的方法来发现聚类[17],其主要是把高维空间数据投影到低维空间,而局部区域中的极低密度的点即为离群点。Hu Tian-ming 等提出了基于模式的方法[18],并认为离群点有偏离簇的稀疏点和偏离常规的密集点两种模式。这些面向统计数据的方法,由于只考虑非空间属性的偏离,因此不适合在空间数据中直接应用。

另一类多维离群点查找方法是二分多维的方法。这类方法因面向空间数据库,故适合查找空间离群点,其又可分为基于图形的方法与定量测试的方法两种。其中,图形的方法是用可视化的方法来查找空间离群点,如变量云与散点图[19,20]。定量的方法是用测试来区分离群点与非离群点的方法,如 Scatterplot 方法[21,22] 和 Moran scatterplot 方法[23]。但这些方法有如下的缺点:①由于它们是单变量的,因此不适合多维空间;②变量云要求大量的后处理,这将造成查找效率低;③基于图形的方法没有确定的标准用来区分离群点等等。如今这些方法已不实用。Shekhar 等提出了一个新的空间偏离点的定义[2,24],该查找方法区分空间属性与非空间属性,并可用空间属性来定义邻域,而用非空间属性来识别离群点。但是,这些方法仍然是强调非空间属性的偏离。

## 2　空间离群点及其相关定义

本文用非空间属性来区分对象的类别,而用空

间属性(空间位置)来定义对象的偏离。由于空间对象一般可用一个点来代表,所以在下文中点和对象是一个概念。

若给定一个空间数据库 $D$ 和一个距离函数 $d$,$X_{\mathrm{Eps}}$为实数参数,$P_{\min}$为正整数参数,$q.\ attr$ 表示非空间属性变量,则可给出以下定义:

**定义 1** 以对象 $p$ 为圆心,以 $X_{\mathrm{Eps}}$为半径的区域内,满足一定非空间属性的对象集,即 $\{q \in D \mid d(p, q) \leqslant X_{\mathrm{Eps}}$ and $q.\ attr\ satisfy\ C\}$ 称为点 $p$ 的影响域,表示为 $N_{\mathrm{Eps}}(p)$。

**定义 2** 点 $p$ 的影响域中,除自己以外的满足一定非空间属性的对象称为点 $p$ 的邻居。$|N_{\mathrm{Eps}}(p)|$表示影响域的基,即影响域内的邻居数。

**定义 3** 如果一个点的影响域内有至少 $P_{\min}$ 个满足一定非空间属性的点,则称影响域是密集的,这个对象称为核心点。

**定义 4** 如果点的影响域包含少于 $P_{\min}$ 个满足一定非空间属性的点,则称这个影像域是稀疏的。如果一个点是核心点的邻居,而且这个点的邻域是稀疏的,则称这个点为边缘点。

**定义 5** 如果一个点是核心点或边缘点,并且是某个边缘点的邻居,则称这个点为邻近边缘点。

**定义 6** 如果两个点满足下面条件之一:

(1) $p \in N_{\mathrm{Eps}}(q)$, $|N_{\mathrm{Eps}}(q)| \geqslant P_{\min}$;

(2) $q \in N_{\mathrm{Eps}}(p)$, $|N_{\mathrm{Eps}}(p)| \geqslant P_{\min}$。

则称这两个点是直接密度可达。这个定义与 DBSCAN 算法中的定义类似。

**定义 7** 如果存在一个点链 $p_1, \cdots, p_n$ 对于 $1 \leqslant i \leqslant n-1$,$p_{i+1}$ 是 $p_i$ 的直接密度可达点,且 $p = p_n$,$q = p_1$,则称点 $p$ 和 $q$ 相互密度可达[12]。

**定义 8** 一个聚类 $C$ 是一个数据库 $D$ 中的满足下面条件的非空子集:对于点 $p, q \in D$,如果 $p \in C$ 并且点 $p, q$ 密度可达,则 $q \in C$。

**定义 9** 不是核心点或边缘点并满足一定非空间属性的对象称为空间离群点,即点 $p$ 为离群点应满足下面条件:$p \in D$,$p.\ attr\ satisfy\ C$,$|N_{\mathrm{Eps}}(p)| < P_{\min}$,并且 $\forall q \in D$,如果 $|N_{\mathrm{Eps}}(q)| > P_{\min}$,则 $p \notin N_{\mathrm{Eps}}(q)$。

## 3 空间离群点查找算法 DBSODLS

为了查找离群点,已有的基于密度的离群点查找方法必须计算每个点的邻域,但由于邻域查找是费时的操作,所以本文应该尽量避免这种操作。当一个点的邻域是密集的,则这个点以及它所有的邻居不可能是离群点,所以计算一个邻域后,如果发现它是密集的,则就不必对它的邻居进行邻域计算,这样就可以节省大量的时间。

下面给出基于密度的跳跃取样空间离群点查找算法(density-based spatial outlier detecting with leapingly sampling, DBSODLS),算法步骤如下:

```
Algorithm DBSODLS(D, X_Eps, P_min)
(1) CandidateSet = Empty;
(2) ClusteringSet = Empty;
(3) While(!D.isClassified())
(4)   {p = Select_unclassified_point(D);
(5)   NeighborhoodSet = D.Neighbors(p, X_Eps);
(6)   if(|NeighborhoodSet| > P_min)
(7)     ClusteringSet = ClusteringSet ∪ NeighborhoodSet ∪ {p}
(8)   else
(9)     CandidateSet = CandidateSet ∪ NeighborhoodSet ∪ {p}
(10)  endif;
(11)  }  // While !D.isClassified
(12) Borders = Empty;
(13) While(!CandidateSet.isLabel)
(14)   {Select one point q from CandidateSet;
(15)   q.isLabel;
(16)   CSNB = CluseringSet.Neighbors(q, X_Eps);
(17)   Borders = Borders ∪ (CSNB ∩ ClusteringSet);
(18)  }  // While !CandidateSet.isLabel
(19) While(!Borders.isLabel)
(20)     {Select one point b from Borders;
(21)     b.isLabel;
(22)     Bord_NB = D.Neighbors(b);
(23)     if(|Bord_NB| > P_min)
(24)       CandidateSet.deleate(Bord_NB);
(25)     }  //While !Borders.isLabel
(26) OutlierSet = CandidateSet;
```

这个算法分为以下 3 部分:

(1)划分阶段(第 3 ~ 11 步) 该阶段把所有的数据点划分成聚类点与候选离群点两部分。划分时,首先任选一个点,并计算它的邻域,若邻域为密集的,则此邻域内所有的点(包括这点本身与它的所有邻居)为聚类点,否则邻域内所有的点为候选离群点,然后选择下一个点继续扩张,下一个点为邻域内离圆心最远的刚刚被访问的点(即以前没有被访问过),接着继续计算,如果找不到下一个符合条件的点,则在数据库中随机选择未被访问的点。这样一直到把所有的点划分成聚类点与候选离群点两类。

在这个阶段因为有可能把一些边缘点也划入到候选离群点集中,故算法要进一步的求精。这就需要查找这些点的邻近边缘点,以便用这些点来识别误划入的点。

(2)邻近边缘点查找阶段(第 13 ~ 18 步)　该阶段完成对邻近边缘点的查找,即检查候选离群点邻域里的每一个邻居,若有邻居在聚类点集里,则把这个邻居放入邻近边缘点集。邻近边缘点集将用来把误选入候选离群点的边缘点剔除,这将在第 3 阶段执行。

(3)求精阶段(第 19 ~ 25 步)　如果邻近边缘点的邻域是密集的,并且它包含了某个候选离群点,则把这个候选离群点剔除,因为它实际上是一个边缘点,而不是离群点。

DBSODLS 算法的第 1、2 步是用来初始化集合变量 *CandidateSet* 与 *ClusteringSet*,它们分别用来存储候选离群点与聚类点;第 3 步 ~ 第 17 步是划分阶段,其中,第 3 步是一个循环,当所有的点划分结束时,则循环结束,在第 4 步按上文提出的策略选择一个点,第 5 步计算它的邻域;第 6 步 ~ 第 10 步用于对邻域进行判断,若邻域是密集的,则为聚类点,否则为候选离群点;第 13 步 ~ 第 18 步为邻近边缘点查找阶段,其中,第 16 步的 *ClusteringSet. Neighbors* 为计算邻域函数,第 17 步是把邻域中属于聚类点的邻居并入邻近边缘点集;第 19 步 ~ 第 25 步是用第 3 阶段收集到的邻近边缘点来剔除误选入候选离群点集的边缘点。对候选离群点集中的每个点,用第 22 步来计算它的邻域,若为密集的,则它为边缘点,不是离群点;第 26 步输出离群点。

为了解析 DBSODLS 算法,本文给出一个示例(如图 2 所示)。图中的两个类型对象分别用两种不同的符号表示。本文关注圆点的对象。显然,圆点对象有两个聚类和两个离群点。聚类在图中部与图的右下角,离群点为点 $a$ 与 $d$。当算法运行到第 3 步 ~ 第 11 步时,则把空间对象分为两部分——聚类点与候选离群点。假设算法选择点 $a$,则计算点 $a$ 的邻域;假设对象 $b$ 与 $c$ 没有被任何邻域标记,因为点 $a$ 的邻域只包含邻居 $b$、$c$,是稀疏的(设 $P_{\min}=5$),故对象 $a$、$b$、$c$ 被标为候选离群点,放入候选离群点集,但实际上,由于对象 $b$、$c$ 是边缘点,而不是离群点,故算法必需进一步对候选离群点进行求精。算法执行第 2 阶段时,会计算对象 $a$、$b$、$c$ 的邻域,若对象 $b$、$c$ 的邻域含有聚类集的点(如图中多边形所包含的点,即对象 $b$、$c$ 为邻近边缘点),则这些点将被放入邻近边缘点集。在第 3 步中,这些点用来帮助剔除对象 $b$、$c$。因为多边形中的点一定有一些点的邻域是密集的,并且包含对象 $b$、$c$,故对象 $b$、$c$ 将被剔除。

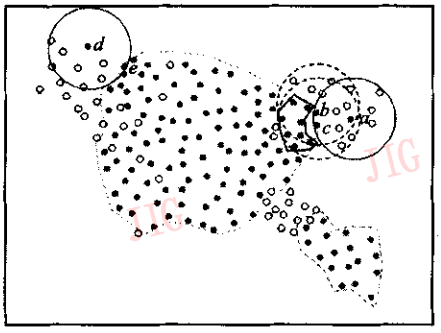


图 2　DBSODLS 算法解析图

Fig. 2　DBSODLS algorithm illumination

## 4　DBSODLS 算法的时间复杂性

本算法最复杂的操作是影响域查询操作,此时,邻域查询需要扫描数据库,复杂性为 $O(n)$。若使用空间索引,如果用 R * -trees[24] 或 SR-trees[25],则对于一个点数为 $n$ 的区间,其影响域查找的时间复杂性为 $O(\log n)$。假设每个影响域中对象个数为 $k$,初选点位 $x$ 个,稀疏的影响域有 $y$ 个($x \ll n, y \ll n$),且对于密集的影响域内的邻居算法不计算它们的邻域,则第 1 与第 2 阶段计算的影响域为 $(x+(k-1)\times y)$ 个。设邻近边缘点有 $z$ 个($x \ll n$),则第 3 阶段计算的影响域有 $z$ 个。由此可见,算法的时间复杂性为 $O((x+(k-1)\times y+z)\log n$,令 $m=(x+(k-1)\times y+z)$,则算法的时间复杂性为 $O(m\log n)$。$m$ 的大小与影响域的半径与数据的密集度有关。在一定的范围内,若 $X_{\mathrm{Eps}}$ 值越大,则查询的影响域越少;若数据越密集,则查询的影响域越少。通常,因 DBSODLS 算法查询的影响域数只有其他基于密度的算法的几分之一,故 DBSODLS 算法比其他算法快几倍。

## 5　DBSODLS 算法与其他基于密度的挖掘算法理论比较

DBSCAN 算法[12] 与 LOF 算法[9] 是基于密度的离群点挖掘算法的两个代表算法。但是,这些算法最坏的时间性能并没有得到明显的改进,其中最重要的原因是这些算法都要对所有对象进行一种非常耗时的操作——影响域查找操作。

### 5.1 DBSODLS 算法与 GDBSCAN 算法性能比较

GDBSCAN 算法[26]是 DBSCAN 算法的扩展,以便用于空间数据库。这个算法可以查找聚类与离群点。

GDBSCAN 算法中的 Eps-邻域计算相当于 DBSODLS 中的影响域查询计算,这是算法中最费时的操作。由于 GDBSCAN 算法是通过计算 Eps-邻域来计算可达距离,以查找聚类点与离群点,故 GDBSCAN 算法必须计算数据库中每一点的 Eps-邻域。但是,DBSODLS 算法却不需要计算每个点的影响域,因为当一个点的邻域是密集的,则它的所有邻居将不要再计算影响域,所以就大大地减少了这个耗时的计算,可见 DBSODLS 算法时间性能优于 GDBSCAN 算法。

### 5.2 DBSODLS 算法与 LOF 算法性能的比较

LOF 算法是通过计算每一点离群因子来识别离群点。离群因子 $LOF(p)$ 是指对象 $p$ 的局部可达密度与对象 $p$ 的邻居的局部可达密度之平均值的比值。由于计算局部可达密度必须计算 $k$-邻域,$k$-邻域的计算相当于影响域的计算,因此是最耗时的计算。由于 LOF 算法必须计算所有点的离群因子,故必须对所有点的邻域进行查询,而本文算法却不必对所有对象进行查询,因此 DBSODLS 算法时间性能优于 LOF 算法。

DBSODLS 算法要进行查询的邻域数和 $X_{\mathrm{Eps}}$ 的值与数据密集的程度有关。在一定的范围内,若 $X_{\mathrm{Eps}}$ 值越大,则本算法查询的邻域越少;若数据越密集,则本算法查询的邻域越少。下面的实验将验证这些性质。

## 6 实验评价

为了对算法的有效性与效率进行实验评价,本文采用影响算法时间性能的两个主要因素来进行评价。算法有效性是指能正确区分离群点与聚类点;效率意味着时间性能是否优于已有的方法。第 1 个实验先用少量的具有说明性的合成数据来说明本方法的有效性,然后用大量的合成数据来研究算法的效率;最后,本文对影响邻域查找次数的参数,即对算法的效率有重大影响的两个参数进行了实验分析。实验中,算法用 Vc6.0 编程,所有的实验是在 CPU 为 Pentium-4-2.2G,内存为 256MB,系统为 Windows XP professional 的个人电脑上进行的。

### 6.1 有效性实验

为了说明本算法的有效性,本文用图 3 所示的

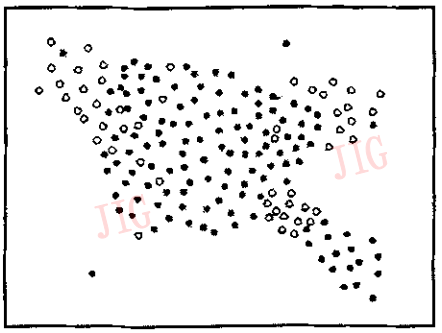

图 3 人工合成数据库

Fig. 3 Artificial database

合成数据来验证算法的有效性。在这些数据中,对象两种不同的非空间属性用圆圈与圆点来表示。该实验查找圆点对象的离群点,DBSODLS 算法中的 $q.attr$ 设为“圆点”;DBSODLS 算法与 GDBSCAN 算法查找的邻域的半径设为 30,邻域包含的最少对象的数目为 3。LOF 算法中 $MinPts$(即本文正整数参数 $P_{\min}$)也为 3,$LOF > 1.5$。图 4 是 DBSODLS 算法找到的离群点的示意图;图 5 是 GDBSCAN 算法找到的聚类与离群点的示意图;图 6 是 LOF 算法找到的离群点的示意图。由图 4、图 5、图 6 可见,DBSODLS 算法与 GDBSCAN 算法都能正确识别离群点,而 LOF 算法则不能正确识别全部离群点,因为它的离群点定义没有区分空间与非空间属性。

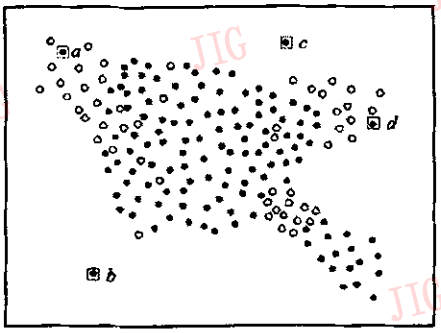


图 4 DBSODLS 算法找到的全部 4 个离群点

Fig. 4 Outliers identified by DBSODLS

### 6.2 算法效率比较

为了比较 GDBSCAN 算法、LOF 算法与 DBSODLS 算法的查找效率。本文用模拟数据发生器产生了具有各种密度分布的对象数分别为 2 000、4 000、8 000、20 000、50 000、100 000、150 000、200 000 的数据来进行实验。GDBSCAN 算法与 DBSODLS

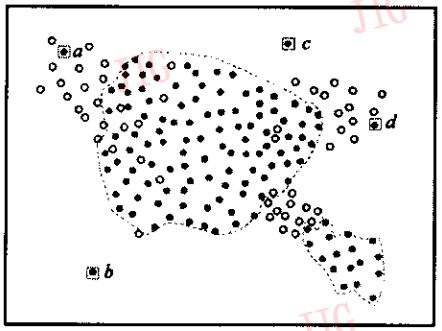


图 5　GDBSCAN 算法查找到的全部 4 个离群点以及 2 个聚类

Fig. 5　All four outliers and two clusters identified by GDBSCAN algorithm

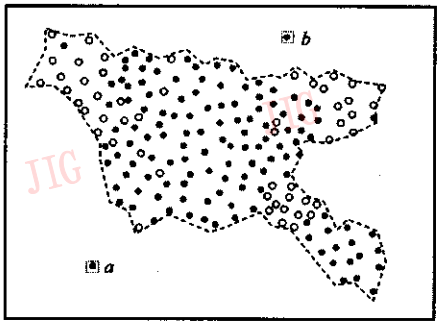


图 6　LOF 算法查找到的 2 个离群点并把中间两类对象当作一个聚类

Fig. 6　Two outliers identified by LOF algorithm and two clusters in middle taken as one cluster by it

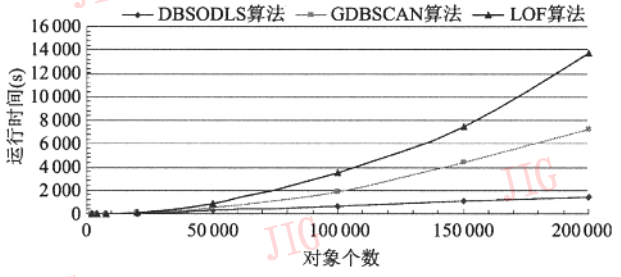


图 7　算法 DBSODLS 与 GDBSCAN、LOF 3 个算法运行时间比较

Fig. 7　Time efficiency comparison between GDBSCAN, LOF and DBGORS

算法查询邻域的半径设为 50,邻域包含的最少对象的数目为 10。LOF 算法的参数 $MinPts = 30$,$LOF > 1.5$。图 7 显示了它们的运行时间与数据库大小的关系。从图 7 可以看出,DBSODLS 算法运行的时间随数据量增加,以接近线性的比例增加,且运行时间明显少于其他两个算法。

### 6.3　影响时间性能的两个主要因素评价

邻域查询是最耗时的操作,而影响邻域查询次数的主要是如下两个因素:第 1 个是邻域半径的取值,图 8 显示 DBSODLS 算法、GDBSCAN 算法与 LOF 算法在一个 15 000 个点的含有各种密度的数据库中进行邻域查询的次数与邻域半径取值的关系。因为 GDBSCAN 算法与 LOF 算法要查询所有点的邻域,故邻域查询次数与邻域半径的关系呈一直线。对于 DBSODLS 算法,$X_{Eps}$ 在 1 到 13 的范围中,随着 $X_{Eps}$ 值的增加,邻域查询次数减少;但当 $X_{Eps}$ 大于 13 时,邻域查询的次数不再减少。从图 8 可以看出,DBSODLS 算法查询的邻域数目明显少于其他两个算法。

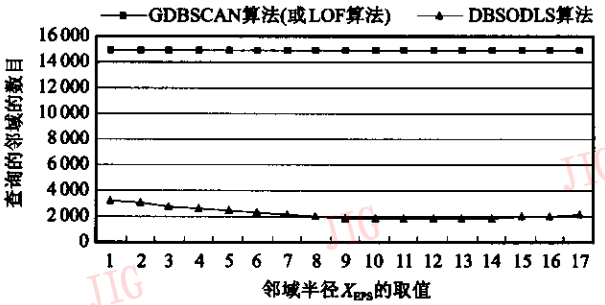


图 8　邻域半径的取值与需要查找邻域数目的关系

Fig. 8　Relation between value of radius of neighborhood and the number of neighborhood required

另一个主要的影响因素是离群点的数目,即聚类点的密度。图 9 显示了具有 10 000、20 000、40 000、80 000、100 000 个对象的数据库,其在包含各种不同的离群点份额的数据中查询邻域的次数。图 9 表明了邻域查询次数随离群点数目的增加而增加。

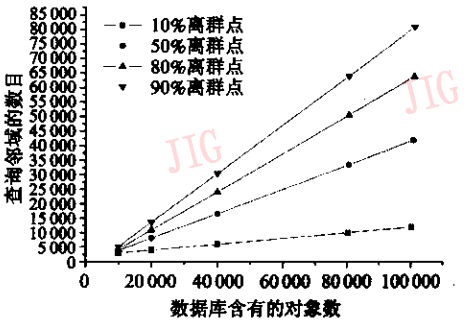


图 9　DBSODLS 算法查询的邻域数与含有离群点的百分比的关系

Fig. 9　Relation between the number of neighborhood required by DBSODLS algorithm and the percentage of outlier in database

## 7 结 论

随着遥感、多媒体等空间数据的积累与空间数据库在各部门的广泛应用，查找空间离群点等空间数据挖掘已成为数据挖掘与知识发现的重要任务之一。由于已有的离群点查找方法强调非空间属性的偏离，其在某些应用领域中，不能很好地查找离群点，为此本文提出了综合考虑空间属性与非空间属性的离群点定义，从而为这些应用领域中的空间数据挖掘提供了新的方法。已有的基于密度的离群点查找方法，由于要求计算所有点的邻域，因此时间效率低，本文提出一种基于密度的跳跃地计算邻域的方法，由于该方法可尽可能减少邻域的查找次数，因此可提高时间效率。理论分析与实验结果表明，该方法时间性能优于已有的基于密度的方法。

## 参考文献(References)


1 Hawkins D. Identification of outliers [M]. London：Chapman and Hall, 1980.

2 Dai H, Srikant R, Zhang C. OBE：Outlier by example [A]. In：Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD) [C], Sydney, Australia, 2004：222～234.

3 Shekhar S, Lu C T, Zhang P. A unified approach to detecting spatial outliers [J]. GeoInformatica, 2003, **7**(2)：139～166.

4 Lu C T, Chen D, Kou Y. Algorithms for spatial outlier detection [A]. In：Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM) [C], Melbourne, Florida, USA, 2003：597～600.

5 Knorr E M, Ng R T. A unified notion of outliers：properties and computation [A]. In：Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD) [C], Newport Beach, CA, USA, 1997：219～222.

6 Knorr E M, Ng R T. Algorithms for mining distance-based outliers in large datasets [A]. In：Proceedings of 24th Very Large Data Bases (VLDB) Conference [C], New York, USA, 1998：392～403.

7 Ramaswamy S, Rastogi R, Kyuseok S. Efficient algorithms for mining outliers from large data sets [A]. In：Proceedings of the ACM International Conference on Management of Data (SIGMOD) '00 [C], Dallas, Texas, USA, 2000：93～104.

8 Arning A, Agrawal R, Raghavan P. A linear method for deviation detection in large databases [A]. In：Proceedings of Conference on Knowledge Discovery and Data Mining (KDD) '96 [C], Portland OR, USA, 1996：164～169.

9 Breunig M M, Kriegel H P, Ng R T, *et al*. LOF：Identifying density-based local outliers [A]. In：Proceedings of the ACM International Conference on Management of Data (SIGMOD) '00 [C], Dallas, Texas, USA, 2000：427～438.

10 Preparata F, Shamos M. Computatinal geometry：an introduction [M]. New York：Springer Verlag, 1998.

11 Ruts I, Rousseeuw P. Computing depth contours of bivariate point clouds [J]. Computational Statistics and Data Analysis, 1996, **23**(1)：153～168.

12 Ester M, Kriegel H P, Sander J, *et al*. A densitybased algorithm for discovering clusters in large spatial databases [A]. In：Proceedings of Conference on Knowledge Discovery and Data Mining (KDD) '96 [C], Portland OR, USA, 1996：226～231.

13 Guha S, Rastogi R, Kyuseok S. ROCK：A robust clustering algorithm for categorical attributes [A]. In：Proceedings of 15th International Conference on Data Engineering [C], Sydney, Australia, 1999：512～521.

14 Nanopoulos A, Theodoridis Y, Manolopoulos Y. C2P：Clustering based on closest pairs [A]. In：Proceedings of Very Large Data Bases (VLDB) '01 [C], Rome Italy, 2001：331～340.

15 He Zeng-you, Xu Xiao-fei, Deng S. Discovering cluster-based local outliers [J]. Pattern Recognition Letters, 2003, **24** (9～10)：1642～1650.

16 He Zeng-you, Xu Xiao-fei, Huang Joshua-zhexue, *et al*. Mining class outliers：concepts, algorithms and applications in CRM [J]. Expert Systems with Applications, 2004, **27**(4)：681～697.

17 Aggarwal C, Yu P. Outlier detection for high dimensional data [A]. In：Proceedings of the ACM SIGMOD '01 Internation Conference on Management of Data [C], Santa Barbara, CA, USA, 2001：37～46.

18 Hu Tian-ming, Sung Sam-Y. Detecting pattern-based outliers [J]. Pattern Recongition Letters, 2003, **24**(16)：3059～3068.

19 Haslett J, Brandley R, Craig P, *et al*. Dynamic graphics for exploring spatial data with application to locating global and local anomalies [J]. The American Statistician, 1991, (45)：234～242.

20 Panatier Y. Variowin, software for spatial data analysis in 2D [M]. New York：Springer-Verlag, 1996.

21 Haining R. Spatial Data Analysis in the Social and Environmental Sciences [M]. Cambridge, UK：Cambridge University Press, 1993.

22 Luc A. Exploratory Spatial Data Analysis and Geographic Information Systems [A]. In：Painho M, editor：New Tools for Spatial Analysis [M], Lisbon, Portugal：ISEGI, EUROSTAT, 1994：45～54.

23 Luc A. Local indicators of spatial association：LISA [J]. Geographical Analysis, 1995, **27**(2)：93～115.

24 Beckmann N, Kriegel H-P, Schneider R, *et al*. The R*-Tree：an efficient and robust access method for points and rectangles [J]. SIGMOD Record, 1990, **19**(2)：322～331.

25 Katayama N, Satoh S. The SR-tree：an index structure for high-dimensional nearest neighbor queries [J]. SIGMOD Record, 1997, **26**(2)：369～380.

26 Sander J, Ester E M, Kriegel H, *et al*. Density-based clustering in spatial databases：the algorithm GDBSCAN and its applications [J]. Data Mining and Knowledge Discovery, 1998, **2**(2)：169～194.