南京航空航天大学

论文集

（二〇〇五年） 第16册

信息科学与技术学院

（第2分册）

南京航空航天大学科技部编

二〇〇六年三月

信息科学与技术学院

042 系

目　　录

序号	姓名	职称	单位	论文题目	刊物、会议名称	年、卷、期	类别
001	黄添强 秦小麟	博士生 教授	042 042	Density-Based Spatial Outliers Detecting	Lecture Notes in Computer Science	2005.3514.	
002	黄添强 秦小麟	博士生 教授	042 042	Quick Spatial Outliers Detecting with Random Sampling	Lecture Notes in Computer Science	2005.3501.	
003	黄添强 秦小麟	博士生 教授	042 042	一种新的空间多维关联规则模型与算法	南京航空航天大学学报	2005.37.03	
004	包 磊 秦小麟	博士生 教授	042 042	用于运动趋势预测的灰色时空数据模型	中国图象图形学报	2005.10.02	
005	包 磊 秦小麟	博士生 教授	042 042	基于灰集的不确定性时空数据模型	遥感学报	2005.09.06	
006	包 磊 秦小麟	博士生 教授	042 042	The Discrete Representation of Continuously Moving Indeterminate Objects	Chinese Journal of Aeronautics	2005.18.01	
007	包 磊 秦小麟	博士生 教授	042 042	Approximate Spatiotemporal Relations Between Indeterminate Evolving Regions, ICMLC 2005	International Conference on Machine Learning and Cybernetics会议	2005	
008	钟 勇 秦小麟 包 磊	博士生 教授 博士生	042 042 042	基于应用查询模式的挖掘算法及其在入侵检测中的应用	应用科学学报	2005.23.05	
009	钟 勇 秦小麟	博士生 教授	042 042	基于时态的多级数据访问控制方法	系统工程与电子技术	2005.27.03	
010	赵宝献 秦小麟	硕士生 教授	042 042	数据库访问控制研究综述	计算机科学	2005.32.01	
011	常 远 秦小麟	硕士生 教授	042 042	基于CNSDTF和OpenFlight的空间数据互操作研究	中国图象图形学报	2005.10.12	
012	郑吉平 秦小麟	博士生 教授	042 042	数据挖掘中采样技术的研究	系统工程与电子技术	2005.27.11	
013	皮德常	副教授	042	计算机专业开展“双语教学”的实践与探讨	计算机教育	2005.18.06	
014	皮德常	副教授	042	Radio Monitoring System	Journal of Communication and Computer	2005.02.00	
015	皮德常 秦小麟 谷汪锋 程 冉	副教授 教授 硕士生 硕士生	042 042 042 042	STBAR:A MORE EFFICIENT ALGORITHM FOR ASSOCIATION RULE MINING	Proceedings of the Fourth International Conference on Machine Learning and Cybernetics会议	2005	
016	皮德常 王 强 李伟奇 吕 军	副教授 硕士生 高工 高工	042 042 外 外	APA:An Interior-Oriented Intrusion Detection System Based on Multe-agents	LNCS	2005.3619	
017	皮德常	副教授	042	“程序设计语言”课程设计新方法的实践与探讨	南京航空航天大学学报(社科版)	2005.07.01	

目 录

目　录

序号	姓名	职称	单位	论文题目	刊物、会议名称	年、卷、期	类别
031	曹汝鸣 毛宇光	硕士生 副教授	042 042	用于不完全信息系统的四值逻辑	扬州大学学报	2005.08.08	
032	黄　慧 毛宇光	硕士生 副教授	042 042	基于多重集的次协调数据库的研究	计算机应用	2005.25.12 (增刊)	
033	王艳磊 毛宇光 武立福	硕士生 副教授 硕士生	042 042 042	基于多版本快照的多级安全事务调度算法	计算机应用	2005.25.12 (增刊)	
034	刘正涛 毛宇光 吴　庄	硕士生 副教授 硕士生	042 042 042	一种新的流数据模型及其扩展	计算机科学	2005.32.07 (增B)	
035	刘正涛 毛宇光 吴　庄	硕士生 副教授 硕士生	042 042 042	持续SPJ查询的有限内存可计算性研究	计算机科学	2005.32.07 (增A)	
036	刘正涛 毛宇光 应　毅	硕士生 副教授 硕士生	042 042 042	基于Web服务的分布式Web应用框架研究	第一届全国Web信息系统及其应用会议	2004	
037	应　毅 毛宇光 刘正涛	硕士生 副教授 硕士生	042 042 042	基于ADO.NET技术的Wed访问数据库研究与实现	计算机与现代化	2005.21.04	
038	应　毅 毛宇光	硕士生 副教授	042 042	可信度在次协调关系数据库中的应用	计算机科学	2005.32.07 (增B)	
039	杨　宁 毛宇光	硕士生 副教授	042 042	基于Vague集的广义模糊关系数据模型	计算机工程与应用	2005.41.11	
040	宋卫东 毛宇光 张玲东	硕士生 副教授 硕士生	042 042 042	数据流挖掘技术研究	微机发展	2005.15.08	
041	李旭帅 毛宇光	硕士生 副教授	042 042	SQL语言的形式语义	微机发展	2005.15.03	
042	武立福 毛宇光	硕士生 副教授	042 042	多级安全数据库保密性和数据完整性研究	计算机工程与应用	2004.40.08	
043	徐　敏 张丽萍	讲师 讲师	042 080	基于Fisher线性判别式的层次文档分类	南京理工大学学报	2005.29.04	
044	杜国平	博士生	042	反正法与归谬法的现代分析	自然辩证法研究	2005.21.03	
045	谭晓阳	副教授	042	Weighted SOM-face:Selecting Local Features for Recognition from Individual Face Image	LNCS	2005.3578	
046	谭晓阳	副教授	042	Feature Selection for High Dimensional Face Image Using Self-organizing Maps	LNAI	2005.3518	
047	谭晓阳	副教授	042	基于“SOM脸”的选择性单训练样本人脸识别	南京航空航天大学学报	2005.37.01	
048	谭晓阳	副教授	042	Recognizing Partially Occluded, Expression Variant Faces From Single Training Image per Person With SOM and k-NN Ensemble	IEEE Transactions on Neural Networks	2005.16.04	

目　录

目　录

序号	姓名	职称	单位	论文题目	刊物、会议名称	年、卷、期	类别
067	王 敏 陈松灿	博士生 教授	042 042	基于小世界体系的指数自联想记忆模型研究	计算机学报	2005.28.12	
068	田永军 陈松灿	硕士生 教授	042 042	面向矩阵模式的正则化Ho-Kashyap算法	计算机研究与发展	2005.42.09	
069	谭可人 陈松灿	硕士生 教授	042 042	Adaptively weighted sub-pattern PCA for face recognition	Neurocomputing	2005.64.00	
070	李 开 陈松灿	硕士生 教授	042 042	用包含运动轮廓特征的水平集方法实现人的轮廓跟踪	江苏工业学院学报	2005.17.04	
071	赵秋荷 陈松灿	硕士生 教授	042 042	一种基于背景减除的运动检测新算法	江苏工业学院学报	2005.17.04	
072	刘 俊 陈松灿	博士生 教授	042 042	Resampling LDA/QR and PCA+LDA for Face Recognition	Australian conference on Artifillal Intellig-ence会议	2005	
073	戴 群 陈松灿	讲师 教授	042 042	Chained DLS-ICBP Neural Networks with Multiple Steps Time Series Prediction	Neural Processing Letters	2005.21.00	
074	黄元元	讲师	042	Binary trademaek retrieval using shape and spatial feature	Proceed ings of SPIE The Internatiomal Society for Optical Engineering	2003.5286	
075	黄元元	讲师	042	Binary trademaek retrieval using entropy and moments	Proceed ings of SPIE The Internatiomal Society for Optical Engineering	2003.5286	
076	张道强	讲师	042	(2D)2PCA:2-directional 2-dimensional PCA for efficient face representation and recognition	Neurocomputing	2005.69.00	
077	张道强 陈松灿	讲师 教授	042 042	Two-dimensional non-negatove matrix factorization for face representation and recognition	LNCS	2005.3723	
078	张道强 陈松灿	讲师 教授	042 042	Representing image matrices: Eigenimages vs.Eigenvectors	LNCS	2005.3497	
079	张道强 陈松灿	讲师 教授	042 042	Fast image compression using matrix K-L transform	Neurocomputing	2005.68.00	
080	张道强 陈松灿	讲师 教授	042 042	A New Face Recognition Method on SVD Perturbation for Single Example Image per person	Applied Mathematics and Computation	2005.163.02	
081	张道强 陈松灿	讲师 教授	042 042	Improving the robustness of' online agglomerative clustering method' based on kernel-induce distance measures	Neural Processing Letters	2005.21.01	
082	张道强 陈松灿	讲师 教授	042 042	核双向联想记忆框架及鲁棒人脸识别	计算机研究与发展	2005.42.00	

目　　录

序号	姓名	职称	单位	论文题目	刊物、会议名称	年、卷、期	类别
083	陈 蕾 张道强 周 鹏 陈松灿	硕士生 讲师 硕士生 教授	042 042 042 042	基于SWA的核自联想记忆模型及其人脸识别应用	应用科学学报	2005.05.00	
084	陈海燕 万麟瑞	助教 副研	042 042	UML-OOPN集成建模技术研究	计算机工程与科学	2005.27.12	
085	王荣培 万麟瑞	硕士生 副研	042 042	多专家AHP的算法改进及其在供应商选择模型中的应用	计算机应用与软件	2005.22.07	
086	袁立罡 万麟瑞	硕士生 副研	042 042	XUML/ACME集成建模方法与VMI构架研究	计算机工程与设计	2005.26.08	
087	王传栋 黄志球 张江涛 张 静	硕士生 教授 硕士生 硕士生	042 042 042 042	面向工程试验的元数据管理模型研究	计算机工程与设计	2005.26.04	
088	张 静 黄志球 王传栋 张江涛	硕士生 教授 硕士生 硕士生	042 042 042 042	面向对象耦合性度量工具的设计与实现	计算机应用研究	2005.10.00	
089	张江涛 黄志球 王传栋 张 静	硕士生 教授 硕士生 硕士生	042 042 042 042	基于Web高可用性PDM体系结构	计算机工程与设计	2005.26.02	
090	侯 萍	硕士生	042	Some Representation Theorems for RecoveringContraction Relations	Journal of Computer Technology	2005.20.04	

Density-Based Spatial Outliers Detecting

Tianqiang Huang[1], Xiaolin Qin[1], Chongcheng Chen[2], and Qinmin Wang[2]

[1] Department of Computer Science and Engineering,
Nanjing University of Aeronautics and Astronautics, Nanjing, 210016, China
tianqianghuang@163.com
[2] Spatial Information Research Center in Fujian Province,
Fuzhou, 350002, China
http://www.sirc.gov.cn/

Abstract. Existing work in outlier detection emphasizes the deviation of non-spatial attribution not only in statistical database but also in spatial database. However, both spatial and non-spatial attributes must be synthetically considered in many applications. The definition synthetically considered both was presented in this paper. New Density-based spatial outliers detecting with stochastically searching approach (*SODSS*) was proposed. This method makes the best of information of neighborhood queries that have been detected to reduce many neighborhood queries, which makes it perform excellently, and it keeps some advantages of density-based methods. Theoretical comparison indicates our approach is better than famous algorithms based on neighborhood query. Experimental results show that our approach can effectively identify outliers and it is faster than the algorithms based on neighborhood query by several times.

1 Introduction

A well-quoted definition of outliers is the Hawkin-Outlier [1]. This definition states that an outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism. However, the notion of what is an outlier varies among users, problem domains and even datasets[2]: (i) different users may have different ideas of what constitutes an outlier, (ii) the same user may want to view a dataset from different "viewpoints" and, (iii) different datasets do not conform to specific, hard "rules" (if any).

We focus on outlier in spatial database, in which objects have spatial and non-spatial attributions. Such datasets are prevalent in several applications. Existing work of Multidimensional outlier detection methods can be grouped into two sub-categories, namely homogeneous multidimensional and bipartite multi- dimensional methods [3]. The homogeneous multidimensional methods model data sets as a collection of points in a multidimensional isometric space and provide tests based on concepts such as distance, density, and convex hull depth. These methods do not distinguish between spatial dimensions and attribute dimensions (non-spatial dimensions), and use all dimensions for defining neighborhood as well as for comparison. Another multidimensional outlier detection method is bipartite multidimensional test which is designed to detect spatial outliers. They differentiate between spatial and non-spatial

V.S. Sunderam et al. (Eds.): ICCS 2005, LNCS 3514, pp. 979 – 986, 2005.

attributes. However, they defined outlier as "spatial outlier is spatially referenced objects whose non-spatial attribute values are significantly different from those of other spatially referenced objects in their spatial neighbor- hoods [3,4]", which emphasizes non-spatial deviation and ignores spatial deviation.

In some application, domain specialist needs detect the spatial objects, which have some non- spatial attributes, deviation from other in spatial dimension. For example, in image processing, detecting a certain type vegetable is anomaly in spatial distribution. The vegetable type is non-spatial attribute, and the vegetable location means spatial attributes. As another example, government wants to know middle incoming residents distribution in geo-space. To detect outliers in these instances, spatial and non-spatial attributes may be synthetically taken into account. For example, there are two type objects in Fig. 1. The solid points and rings respectively represent two objects with different non-spatial attribute, such as the solid objects represent one vegetable and the rings are the other. All objects in Fig. 1 are one cluster when we didn't consider non-spatial attribute, but they would have different result when we took spatial and non-spatial attribute into account. Apparently, when we focus solid objects, the solid objects in *C1* and *C2* are clusters, and object *a* and *b* are outliers.

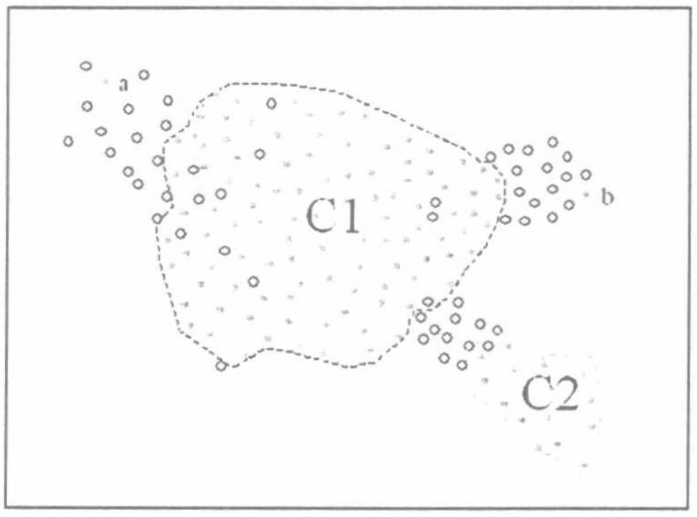

Fig. 1. An illumination example

We took into account of spatial and non-spatial attributes synthetically to define the outliers. If the objects that have some non-spatial attributes are keep away from their neighbor in spatial relation. We defined them outliers.

The main contributions of this paper are: (1) we propose a novel density-based algorithm to detect it, which is the quicker than existing algorithms based on neighborhood query. (2) We evaluate it on both theory and experiments, which demonstrate that algorithm can detect outlier successfully with better efficiency than other algorithms based on neighborhood query.

The remainder of the paper is organized as follows: In section 2, we discuss formal definition of outliers. Section 3 presents the *SODSS* algorithm. Section 4 evaluates performance of *SODSS*. Section 5 reports the experimental evaluation. Finally, Section 6 concludes the paper.

2 Density-Based Notion of Spatial Outliers

In this section we present the new definition of outlier, in which spatial and non-spatial attributes were synthetically taken into account.

Given a dataset *D*, a symmetric distance function *dist*, parameters *Eps* and *MinPts*, and variable *attrs* indicates the non-spatial attributes.

Definition 1. The **impact neighborhood** of a point p, denoted by $IN_{Eps}(p)$, is defined as $IN_{Eps}(p) = \{q \in D \mid \mathrm{dist}(p, q) \leq Eps$ and $q.attrs$ satisfy C$\}$.
Definition 2. The **Neighbor** of *p* is any point in impact neighborhood of *p* except *p*.
Definition 3. If a point's impact neighborhood has at least *MinPts* points, the impact neighborhood is **dense**, and the point is **core point**.
Definition 4. If a point's impact neighborhood has less than *MinPts* points, the impact neighborhood is **not dense**. If a point is a neighbor of core point, but his neighborhood is not dense, the point is **border point**.
Definition 5. If a point is core point or border point, and it near a border point p, the point is **near-border point** of *p*.
Definition 6. A point *p* and a point *q* are **directly density-reachable** from each other if (1) $p \in IN_{Eps}(q)$, $|IN_{Eps}(q)| \geq MinPts$ or (2) $q \in IN_{Eps}(p)$, $|IN_{Eps}(p)| \geq MinPts$.
Definition 7. A point *p* and a point *q* are **density-reachable** from each other, denoted by $DR(p, q)$, if there is a chain of points $p_1,\ldots,p_n$, $p_1=q$, $p_n= p$ such that p_{i+1} is directly density-reachable from p_i for $1 \leq i \leq n$-1.
Definition 8. A **cluster** *C* is a non-empty subset of *D* satisfying the following condition: $p, q \in D$: if $p \in C$ and $DR(p, q)$ holds, then $q \in C$.
Definition 9. **Outlier** *p* is not core object or border object, i.e., *p* satisfying the following conditions: $P \in D$, $|IN(p)| < MinPts$, and $\forall\, q \in D$, if $|IN(q)| > MinPts$, then $p \notin IN(q)$.

3 *SODSS* Algorithm

In *DBSCAN* [5] or *GDBSCAN* [6], to guarantee finding density-based clusters or outliers, determining the directly density-reachable relation for each point by examining the neighborhoods is necessary. However, performing all the region queries to find these neighborhoods is very expensive. Instead, we want to avoid finding the neighborhood of a point wherever possible. In our method, the algorithm discards these dense neighborhoods in first, because these objects in it are impossibly outliers. The algorithm stochastically researched in database but not scan database one by one to find the neighborhood of every point like *DBSCAN*, so the algorithm outperform famous algorithms based on neighborhood query, such as *DBSCAN* [5], *GDBSCAN* [6], *LOF* [7].

In the following, we present the density-based Spatial Outlier Detecting with Stochastically Searching (*SODSS*) algorithm. *SODSS* is consisted of three segments. The first (lines 3~17) is *Dividing Segment*, which divide all object into three parts, cluster set, candidate set or outlier; The second (lines 19~23) is *Near-border Detecting*

Segment, which detect and record the near-border objects of candidate, i.e., the neighbors of these border objects that may be labeled candidate, which would be used to detect these border objects in the third segment; The third (lines 24~31) is *Fining Segment*, using the near-border objects to find these border objects and remove them.

SODSS starts with an arbitrary point *p* and Examine its impact neighborhood *NeighborhoodSet* with *D.Neighbors(p, Eps)* in line 5. If the size of *NeighborhoodSet* is at least *MinPts*, then *p* is a core point and its neighbors are belong to some clustering, to put them into clustering set list; otherwise, if the size is 0, *p* is outlier, so put them into outlier set; or else *p* and his neighbor may be outliers, so put them into candidate set. Lines 19~23 detect neighbors of these that were labeled candidates in *Dividing Segment* and include them into candidate set. These objects would be used to detect border objects that are not outliers from candidate set. Lines 24~31 check every object in candidate set to remove the border objects.

SODSS algorithm

```
Algorithm SODSS(D, Eps, MinPts)
1. CandidateSet = Empty;
2.ClusteringSet = Empty;
3.While (!D.isClassified( ) )
4.   {Select one unclassified point p from D;
5.    NeighborhoodSet = D.Neighbors(p, Eps);
6.    if ( | NeighborhoodSet | > MinPts )
7.       ClusteringSet = ClusteringSet ∪ NeighborhoodSet
8.    else
9.         if( | NeighborhoodSet | > 0 )
10.            {NeighborhoodSet.deleateCluserLabledPoit;
11.             CandidateSet = CandidateSet ∪
                                     NeighborhoodSet ∪ p
12.             }
13.         else
14.             OutlierSet = OutlierSet ∪ p
15.         endif;
16.     endif;
17.    }  // While !D.isClassified
18. Borders = Empty;
19. While ( !CandidateSet.isLabel )
20.      { Select one point q from CandidateSet;
21.        q.isLabel;
22.        Borders = Borders ∪ CluseringSet.Neighbors(q,
                          Eps);
23.      }  //  While !CandidateSet.isLabel
24. While ( !Borders.isLabel )
25.        { Select one point b from CandidateSet;
26.        b.isLabel;
27.        Bord_NB = D.Neighbors( b );
28.        if  ( | Bord_NB | > MinPts )
29.            CandidateSet.deleate (Bord_NB);
30.        OutlierSet = OutlierSet ∪ CandidateSet;
31.        } // While !Borders.isLabel
```

To understand this algorithm, we give example as Fig. 2. There are two type objects in Fig. 2. The solid point represented one-type objects and the ring represented the other type objects. Supposing we focus on solid objects. Apparently, there are two clusters and two outliers in solid objects in the figure. Clusters are located in center and right down, and outliers are object *a* and object *d*. when algorithm run lines 3~17 to divide spatial objects to three parts, cluster set, outlier or candidate set. Algorithm may select object *a*, and calculate neighborhood *A*. Supposing object *b* and *c* have not been labeled in any dense neighborhood. They are the neighbors in neighborhood *A*, and neighborhood *A* is sparse, so they are labeled to candidate. When object *b* and *c* is included in candidate set, the near-border objects near *b* and *c*, which include in the red polygon *P* in Fig. 2., are also included in candidate set through the *Near-border Detecting Segment* in line 19~23. Some of near-border objects in red polygon *P* are dense, so object *b* and *c* would be removed from candidate set. So *SODSS* can identify real outlier.

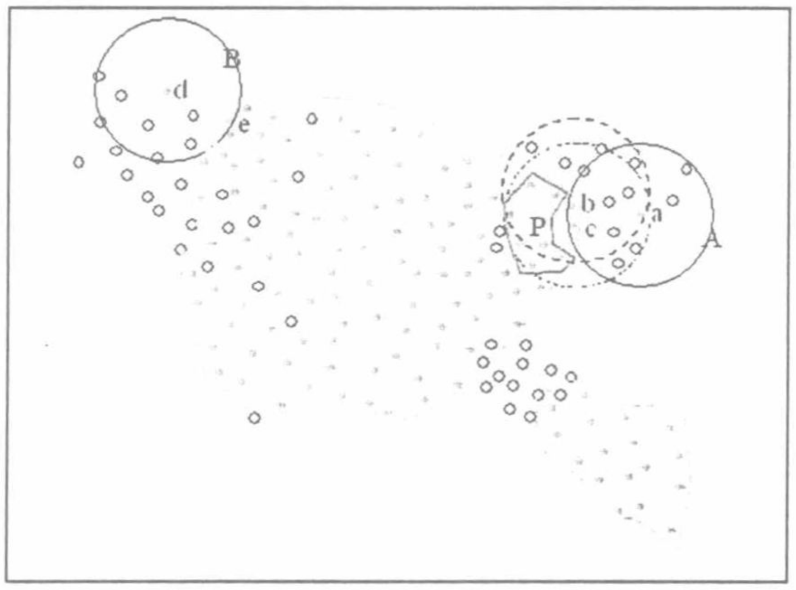

Fig. 2. Object *a* and *d* are outliers. Object *b* and *c* would be labeled candidates. The objects in red polygon *P* are border objects that are put into candidate set in *Near-border Detecting Segment*

4 Theoretical Performance Comparison of *SODSS* and the Other Density-Based Algorithm

There are many density-based algorithm that were proposed to detect outliers, but calculation efficiency is not obviously improved. In the worst case, the time cost of the algorithms are O(n^2). *SODSS* outperform existing algorithms in calculation efficiency.

The neighborhood query *D.Neighbors*(*p*, *Eps*) in line 5 is the most time-consuming part of the algorithm. A neighborhood query can be answered in O(*logn*) time using spatial access methods, such as R*-trees [8] or SR-trees [9]. When any clusters are found, their neighbors would not be examine by *SODSS* again, so *SODSS* will perform fewer neighborhood queries and save much time. Clustering objects must much more than outlier objects, so *SODSS* can reduce much neighborhood query and then have good efficiency. Supposing *SODSS* performs *k* neighborhood queries, its time complexity is O(*klogn*), which *k* is much smaller than *n*. In the second and third segment algorithm must query neighborhood again, but these operation are in candidate set and the number of candidate is very few. The *k* is related to *Eps*, so the time complexity is related to *Eps*. With increasing of *Eps* time cost decreases in certain range, however, the candidates would increase greatly when *Eps* exceeds the threshold and the time cost would increase obviously.

4.1 Performance Comparison of *SODSS* and *GDBSCAN*

GDBSCAN [6] extended the famous algorithm *DBSCAN* to apply to spatial database. *GDBSCAN* identify spatial outlier through detecting cluster, i.e., the noises are outliers. This algorithm scans database and examine all objects neighborhoods.

Eps-Neighborhood of *GDBSCAN* corresponds to impact neighborhood of *SODSS*, which is expensive operation. One crucial difference between *GDBSCAN* and *SODSS* is that once *SODSS* has labeled the neighbors as part of a cluster, it does not examine the neighborhood for each of these neighbors. This difference can lead to significant time saving, especially for dense clusters, where the majority of the points are neighbors of many other points.

4.2 Performance Comparison of *SODSS* and *LOF*

LOF [7] calculates the outlier factor for every object to detect outliers. It is the average of the ratio of the local reachability density of *p* and those of *p*'s *MinPts*-nearest neighbors. The local reachability density is based on *MinPts*-nearest neighbors. *LOF* must calculate *k*-distance neighborhoods of all objects, which time costs are equal to impact neighborhoods query. Calculating *k*-distance neighborhoods is the main expensive operation. *SODSS* detect outlier by removing cluster objects with stochastically researching. All neighbors in dense neighborhood would not calculate their neighborhood again, so the region query of *SODSS* must be less than *LOF*'s. Accordingly, *SODSS* have better efficiency than *LOF*.

5 Experimental Evaluation

The criteria evaluating outlier detection approaches can be divided into two parts: efficiency and effectiveness. Good efficiency means the technique should be applicable not only to small databases of just a few thousand objects, but also to larger databases with more than hundred thousand of objects. As for effectiveness, a good approach should have ability to divide exactly outliers from clusters. We have done many experiments to examine the efficiency and effectiveness, but here limiting to extension we only presented two. In first, we use synthetic data to explain effectiveness of our approach. Secondly, we use large database to verify the efficiency. Experiments showed that our ideas can be used to successfully identify significant local outliers and performance outperforms the other density-based approaches. All experiments were run on a 2.2 GHz PC with 256M memory.

5.1 Effectiveness

To compare *SODSS* with *GDBSCAN* [6] and *LOF* [7] in terms of effectiveness, we use the synthetic sample databases which are depicted in Fig. 3. In these datasets, the non-spatial property for the points is depicted by different symbol, rings and solid points. Experiment focus on solid objects, and set *q.attrs = solid*. Fig. 4 shows the outliers and clusters identified by *SODSS*. The radius set to 2.1 in *SODSS*, MinPts set to 3. *SODSS* and *GDBSCAN* can identify outliers correctly, because they consider non-spatial attribute. As shown in Fig. 5, *LOF* does not find the outliers because it ignores non-spatial attributes and considers all objects are cluster.

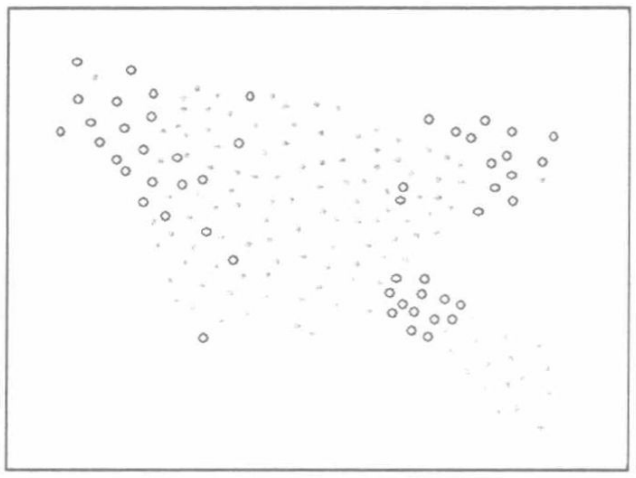

Fig. 3. Synthetic sample databases

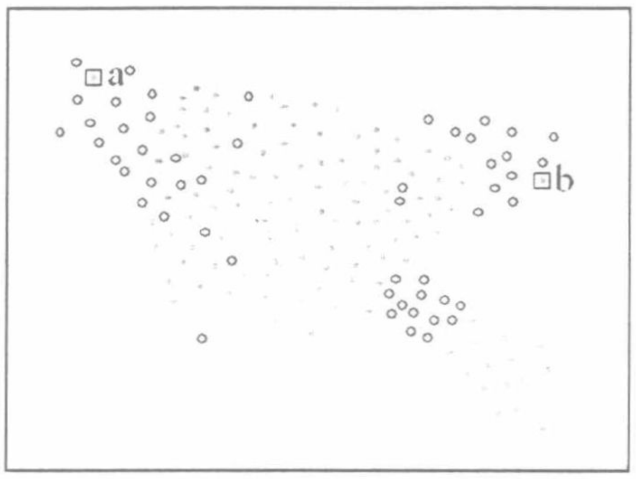

Fig. 4. Outlier *a* and *b* identified by *SODSS* or *SDBDCAN*

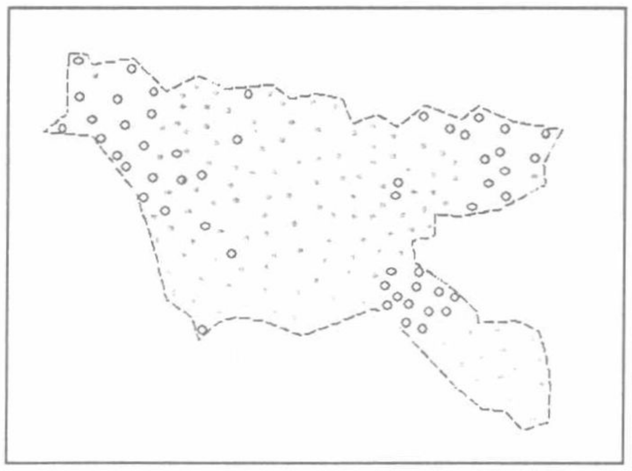

Fig. 5. *LOF* can't identify outliers

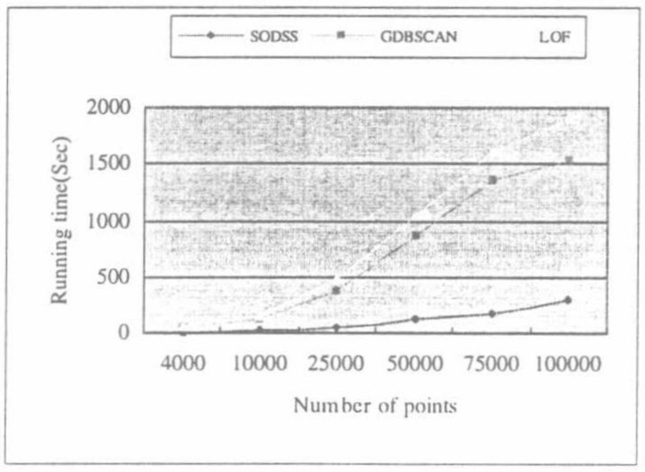

Fig. 6. Time efficiency comparisons between *GDBSCAN*, *LOF* and *SODSS*

5.2 Efficiency

For comparison computational efficiency of *SODSS* and *GDBSCAN* and *LOF*, we used synthetic datasets that are consisted of points from 4000 to 100,000. The *Eps* is 5, and *MinPts* is 10, when *SODSS* query the neighborhood. They are the same when

GDBSCAN run. We set $MinPts = 30$ and $LOF > 1.5$. Fig. 6. shows the running time for *SODSS* increases with the size of the datasets in an almost linear fashion, and the performance is obviously better than the other two.

6 Conclusion

In this paper, we formulated the problem of one-type spatial outlier detection and presented effective and efficient *SODSS* algorithms for spatial outlier mining. This algorithm does not calculate neighborhood of very objects but stochastically research. It discards much region query of cluster, and gained good efficiency.

Acknowledgements. This research supported by the National Nature Science Foundation of China (No. 49971063), the National Nature Science Foundation of Jiangsu Province (BK2001045), the High-Tech Research of Jiangsu Province (BG2004005) and the National High-Tech Research and Development Plan of China (No. 2001AA634010-05).

References

1. D. Hawkins. Identification of Outliers. Chapman and Hall, London, 1980
2. H. Dai, R. Srikant, and C. Zhang. OBE: Outlier by Example. In: Proceedings of PAKDD 2004, Sydney, Australia, May 26-28, 2004, LNAI 3056, pages: 222-234, 2004
3. S. Shekhar, C.T. Lu, and P. Zhang. A unified approach to detecting spatial outliers. GeoInformatica, 7(2): 139-166, 2003
4. C.T. Lu, D. Chen, and Y. Kou. Algorithms for spatial outlier detection. In Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM 2003), December 19-22, 2003, Melbourne, Florida, USA, pages: 597-600. IEEE Computer Society, 2003
5. M. Ester, H.P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases. In: Proceedings of KDD'96, Portland OR, USA, pages: 226-231, 1996
6. J. Sander, M. Ester, H. Kriegel, and X. Xu. Density-based Clustering in Spatial Databases: the algorithm GDBSCAN and its applications. Data Mining and Knowledge Discovery, val. 2, no. 2, pages: 169-194, 1998
7. M.M. Breunig, H.P.Kriegel, R.T.Ng, and J. Sander. LOF: Identifying density-based local outliers. In: Proceedings of SIGMOD'00, Dallas, Texas, pages: 427-438, 2000
8. N. Beckmann, H.P. Kriegel, R. Schneider, and B. Seeger. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. SIGMOD Record, vol. 19, no. 2, pages: 322-331, 1990
9. N. Katayama and S. Satoh. The SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries. SIGMOD Record, vol. 26, no. 2, pages: 369-380, 1997

Quick Spatial Outliers Detecting with Random Sampling

Tianqiang Huang[1], Xiaolin Qin[1], Qinmin Wang[2], and Chongcheng Chen[2]

[1] Department of Computer Science and Engineering,
Nanjing University of Aeronautics and Astronautics, Nanjing, 210016, China
Tianqianghuang@163.com
http://www.nuaa.edu.cn/
[2] Spatial Information Research Center in Fujian Province,
Fuzhou, 350002, China
http://www.sirc.gov.cn/

Abstract. Existing Density-based outlier detecting approaches must calculate neighborhood of every object, which operation is quite time-consuming. The grid-based approaches can detect clusters or outliers with high efficiency, but the approaches have their deficiencies. We proposed new spatial outliers detecting approach with random sampling. This method adsorbs the thought of grid-based approach and extends density-based approach to quickly remove clustering points, and then identify outliers. It is quicker than the approaches based on neighborhood queries and has higher precision. The experimental results show that our approach outperforms existing methods based on neighborhood query.

1 Introduction

The definition of spatial outlier varies with user needs and problem domain etc. Shekhar and Lu et al. [1,2] defined spatial outlier as spatially referenced object whose non-spatial attribute values are significantly different from those of other spatially referenced objects in their spatial neighborhoods. This definition emphasizes non-spatial deviation and ignores spatial deviation. In some application, domain specialist needs detecting the spatial objects, which have some non-spatial attributes, deviate from other in spatial dimension. For example, scientists researched the patients with a certain disease lived in different places. They would consider various kinds of situations which include abnormity of spatial attribute. We took into account of spatial and non-spatial attributes synthetically to define outlier. If the objects that have some non-spatial attributes are keep away from their neighbor in spatial relation. We defined them outliers.

There are many outlier-detecting algorithm. Existing approaches can be broadly classified into the following categories: Distribution-based approach [3], Depth-based approach [4], Clustering approach [5], Distance-based approach [6], Density-based approach [7] and Model-based approach [8,9]. There are many advantages in density-based algorithm but these approaches have poor efficiency. The grid-based approaches that are used to detect clusters of outliers calculate quickly but they have "dimension curse" and have poor precision. We absorb the thought of grid-based algorithm

B. Kégl and G. Lapalme (Eds.): AI 2005, LNAI 3501, pp. 302 – 306, 2005.