The Journal of GIS Association of Korea, Vol. 12, No. 4, pp. 339-349, December 2004

Analysis of Outlier Effects on Spatial Indices

Si-Wan Kim* · Kyoung-Sook Kim* · Ki-Joune Li*

ABSTRACT

Outliers in spatial databases influence on the performance of spatial indexing methods including R-tree. They enlarge the size and overlapping area of MBRs in R-tree which are important factors in determining the performance. In this paper, we give an analysis of outlier effects on R-tree by analytical and experimental work, and propose a method for properly handling outliers. Our experimental results show that our method improves about 15 percents of the performance.

Keywords : Outliers, spatial database, R-tree, MBR

요 약

공간 데이터베이스에서 예외자는 R-tree 계열의 공간색인의 성능에 많은 영향을 미친다. 즉, 예외자로 인하여 R-tree 계열의 공간색인에서 최소경계사각형의 넓이가 불필요하게 넓어지고 겹침 현상이 심해지게 되고 이로 인해 질의처리 시 더 많은 디스크 접근을 필요하게 된다. 따라서, 본 논문에서는 예외자가 공간색인에 주는 영향을 분석하여, 예외자를 미리 처리할 경우, 얼마만큼의 성능을 향상시킬 수 있는지 비용모델과 적절한 예외자의 처리방법을 제안한다. 그리고 실험을 통해 예외자를 미리 처리함으로써 어느 정도의 공간색인의 질의처리 성능을 향상시킬 수 있는지 보여준다. 실험결과에 따른면, 본 논문에서 제안된 예외자의 처리방법이 기존의 공간색인의 성능을 평균 15%정도 향상시킬 수 있음을 보여준다.

주요어 : 예외자, 공간 데이터베이스, 공간색인, 최소경계사각형

* Department of Computer Science, Pusan National University swkim@isel.cs.pusan.ac.kr

1 Introduction

The distribution of spatial objects affects the performance of spatial indexing. It is however difficult to describe the distribution of real data by a parametric model, such as Gaussian model, since objects in real application do not respect any statistical distribution model due to many exceptional factors. Outliers, which exist in the real world, are an important factor in complementing the description of data distribution. Outlier is an object apparently isolated from others and influences on the performance of spatial indexing methods.

In this paper, we particularly focus on the effects of outliers on R-tree[3], since R-tree has been considered as the most robust and efficient spatial indexing method. Outliers enlarge the size of MBR and increase the overlapping area between MBRs in R-tree. By properly handling them, we reduce the size of MBR and overlapping area, and consequently improve the performance of R-tree. In this paper, we will first analyze the effects of outliers on the performance of R-tree, which have been ignored by previous works. And we will propose a method to control outliers in efficient way for enhancing the performance of R-trees. According to our experiments, it improves about 15 percents of the performance for small query. This paper is organized as follows. In the next section, we will present related work. And we will give an analysis of R-tree performance and outliers in section 3. In section 4, we will propose a method to control outliers based on the analysis presented in section 3. Experimental results will be shown in section 5 and the conclusion will be given in the next section.

2. Related Work

Analysis of the relation between properties of spatial objects and the performance of spatial indexing methods is a crucial task in improving the performance of spatial indexing. For this reason, a number of studies have been done to find the factors determining the performance[4][5][7][13]. The performance of R-tree family including R*-tree is mainly determined by the number, size and shape of MBR, which are highly related with the splitting strategy and the distribution of spatial objects.

Most researches on R-tree performance in the early 1990s assumed the uniformity of data distribution[4][5]. However, spatial data in real world are rarely uniformly distributed. Some researches have been done to analyze the performance of R-tree under non-uniform distribution. For example[6], proposed a cost model with several parameters including the density of spatial objects, which is in fact a non-parametric way of describing the distribution of spatial data in 2-dimensional space. This work was extended to high dimensional space

by[13]. In[7], a cost model based on fractal geometry was proposed, and it provides relatively accurate estimation results for 2-dimensional space. An elaborate cost model was proposed by[5] based on hyperbolic power law for describing properties of MBR, such as area of MBR.

These researches give comprehensive analysis on the performance of R-tree and the properties of spatial data. But it still remains a number factors to consider for explaining the performance of R-tree. In this paper, we assume that outlier is another factor to explain the distribution of data and consequently the performance as well.

In our paper, we first investigate the relationship between the performance of R-tree and outliers. Outliers can be defined as objects apparently isolated from others. But we need a more precise definition and a detecting method for outliers. Recently some remarkable researches have been done for outliers in knowledge discovery and data mining domains. Each work gives a definition of outlier, which is slightly different from others, and provides a detecting method. In[10], a distance based outlier is defined as follows,

"An object *O* in a data set *T* is a *DB(p,D)*-outlier of at least fraction *p* of the objects in *T* lies greater than distance *D* from *O*".

And based on this definition, an algorithm with $O(kN^2)$ time complexity was proposed[11]. proposed a different outlier detecting method by introducing LOF (Local Outlier Factor) to consider local densities. These two methods have a serious drawback. In order to apply them, we must have a priori knowledge about the least fraction p or LOF, which can be only found by trial and error or heuristic approach.

In [12], the definition of outliers is slightly modified as "Given a *k* and *n*, a point *p* is an outlier if $card(q \mid D_k(q) > D_k(p), q \neq p) < n$, that is, if no more than *n* -1 other points in the data set have a higher value for D_k than *p*." In fact, the fraction *p* of the definition in [10] is replaced with *k*-nearest in this definition. And they also proposed an efficient outlier detection algorithm. This approach has several advantages over the previous methods. First no a priori knowledge is required in contrast with the others. Second, we can control the number of outliers with ease depending on the system parameters such as buffer size, which will be discussed later.

In this paper, we will apply the definition and detection algorithm for outliers given by [12], since it is efficient and we can control the number of outliers by this method. The control of the number of outliers is an important requirement for our method, which will be proposed in section 4.

3. Outliers and Performance of R-tree

In this section, we will investigate the relationship between outliers and the performance

of R-tree. For the reason of simplicity, we assume the space as $[0, 1]^2$.

3.1 Performance of R-tree

The performance of R-tree is mainly determined by MBR. The size, the shape, the overlapping area of MBR greatly influence on the performance of R-tree. A good R-tree should satisfy the following conditions; 1) small MBR, 2) small overlapping area, and 3) squareness.

The first condition is evident and well explained in [4][7][13]. It is known that the performance of R-tree is proportional to the inverse of its MBR area. And the second condition on the overlapping area between MBRs is discussed in [2] and [3]. In fact, the second condition depends on the first one. In other words, an R-tree results in small overlapping area, when it has small MBRs. The third condition can be intuitively understood, and a brief proof is given as below.

Theorem Among MBRs with the same area, squared MBR results in the least number of disk accesses for intersection query.

Proof Suppose that the width and height of a rectangle *R* are *w* and *h* respectively and the side length of a square *S* with the same area is l. According to [4], the number of a R-tree node accesses for an intersection query Q is approximately given as

$$DA(Q) = \sum_{i=1}^{d}\sum_{k=1}^{N_i}(w_{ij}+q)(h_{ij}+q)$$

where d is the height of the R-tree, N_i is the number of nodes in the i -th level of R-tree and q is the side length of query. Let $r_{ij} = \frac{h_{ij}}{w_{ij}}$ and $T(r_{ij}) = (w_{ij}+q)(r_{ij}w_{ij}+q)$ where $r_{ij} \geq 1$. Then

$$DA(Q) = \sum_{i=1}^{d}\sum_{k=1}^{N_i}(w_{ij}+q)(h_{ij}+q) = \sum_{i=1}^{d}\sum_{k=1}^{N_i}T(r_{ij})$$

When $T(r_{ij})$ is minimum, *DA(Q)* is also minimum. Suppose that S_{ij} is a rectangle whose area is identical with R_{ij}. Then $l_{ij}^2 = w_{ij}h_{ij} = r_{ij}w_{ij}^2$. That is $w_{ij} = \frac{l_{ij}}{\sqrt{r_{ij}}}$. Thus

$$T(r_{ij}) = (w_{ij}+q)(r_{ij}w_{ij}+q) = (\frac{l_{ij}}{\sqrt{r_{ij}}}+q)(l_{ij}\sqrt{r_{ij}}+q)$$

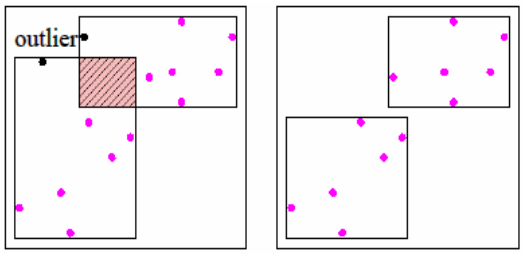
Since $T(r) > 0$ for $r_{ij} \geq 1$, $T(r_{ij})$ is a monotonic increasing function when $r_{ij} \geq 1$, and it is minimum when $r_{ij} = 1$, that is $w_{ij} = h_{ij}$. It means that *DA(Q)* becomes minimum, when every MBR of R-tree node is a square.

In the following sections, we will analyze the relationship between outliers and the performance of R-tree.

3.2 Outliers and Area of MBR

Outliers are related with the three conditions mentioned in the previous subsection. As illustrated by figure 1(b) and figure 1(a), we can reduce the area of MBR by removing outliers and consequently the

overlapping area between MBR as well. We give a brief analysis of outlier effect on the area of MBR.



(a) MBR before removing outliers (b) MBR after removing outliers

Figure 1. Examples of MBR and outliers

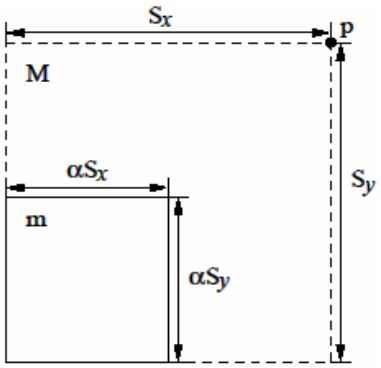


Figure 2. Reduced MBR by removing outliers

As shown by figure 2, suppose that point *p* is an outlier and M is MBR containing *p*. And *m* is the reduced rectangle by removing *p*. Let us investigate the number of leaf node accesses for region query after reducing MBR. According to [4], the number of leaf node accesses for a square region query *Q* is given as

$$DA_{M,leaf}(Q) = \sum_{j=1}^{N_{leaf}} (s_{x,j} + q)(s_{y,j} + q)$$

where $s_{x,j}$ and $s_{y,j}$ are the side length of the *j*-th MBR along *x*-axis and *y*-axis respectively and *q* is the side length of the query. And the number of leaf node accesses after the reduction becomes

$$DA_{M,leaf}(Q) = \sum_{j=1}^{N_{leaf}} (\alpha_{x,j} s_{x,j} + q)(\alpha_{y,j} s_{y,j} + q)$$

where $\alpha_{x,j}$ and $\alpha_{y,j}$ are the reduction rates of the *j*-th MBR along *x*-axis and *y*-axis respectively ($0 < \alpha_{x,j}, \alpha_{y,j} \leq 1$. For the reason of simplicity, we assume that the reduction rates for the MBRs are identical and the average side lengths of MBRs along *x*-axis and *y*-axis are same. Then the ratio of the leaf node accesses before removing outliers over that after removing outliers is approximately

$$\begin{aligned} r_{leaf}(\alpha) &= \frac{DA_{m,leaf}}{DA_{M,leaf}} \\ &= \frac{N_{leaf}(\alpha\bar{s_x} + q)(\alpha\bar{s_y} + q)}{N_{leaf}(\bar{s_x} + q)(\bar{s_y} + q)} \\ &= \frac{(\alpha\bar{s_x} + q)(\alpha\bar{s_y} + q)}{(\bar{s}_x + q)(\bar{s_y} + q)} \end{aligned} \quad (1)$$

If $\bar{s_x} = \bar{s_y}$, the above equation is reduced to

$$r_{leaf}(\alpha) = \left(\frac{\alpha s + q}{s + q}\right)^2 \quad (2)$$

Note that $\alpha = 1$ if there is no reduction. According this equation, the performance gap between two cases becomes more significant, as decreases the reduction ratio α. It means that outliers give a negative influence on the

area of MBR and the performance of R-tree. And it is evident that the overlapping area between MBRs becomes large due to outliers.

And this equation also means that the reduction ratio α does not influence significantly on the performance, when the query size is relatively large compared to the size of MBR. We will see it via experimental results in section 5. We performed experiments to validate the relationship between outliers and the performance of R-tree given by equation (2) with two real data sets on Seoul and Long Beach County containing and a synthetic data. Figure 3 shows that the performance of R-tree is improved by removing outliers. And it also shows the measured improvement of performance by removing outliers and their estimated values by equation (2) and the accuracy is within 5 percents for most cases.

3.3 Outliers and Squareness of MBR

We have seen that the squareness of MBR influences on the performance of R-tree. While outliers are strongly related with the area of MBR, it seems not evident that outliers influence on the squareness of MBR. But some experiments show that the squareness of MBR in R-trees is also affected by outliers. Results of experiments concerning outliers and the squareness of MBR is shown in figure 6(b) in section 5. This figure shows that the squareness for Seoul data is considerably improved by removing outliers, but it is not evident for LBCounty data.

4. Controlling Outliers

We have shown that we can improve the performance of R-tree by removing outliers. We propose a method to handling them. The basic idea of this method is very simple, that is to separate outliers from other objects and manage them on main memory independently. This method is composed with two steps,

1. detect k outliers from data set,
2. remove them from R-tree and store them in an outlier bucket $B(k)$.

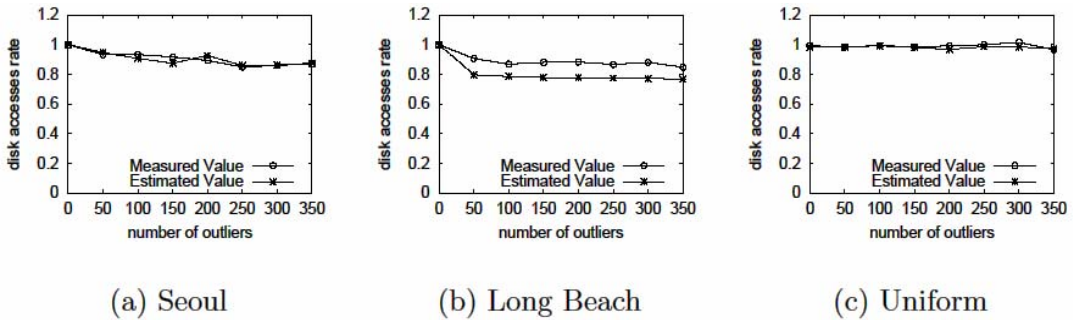


Figure 3. Number of outliers and the rate of leaf node accesses

In order to apply our method, we need an outlier discovery method. When we select an algorithm among the methods that we have explained in section 2, an important requirement must be considered. That is the controllability of the number of outliers, since the size of outlier bucket is limited. And we have chosen the method proposed by [12], since it satisfies this requirement and gives a good performance.

By this method, we first decide the number of outliers noutlier to detect. And we compute $D_k(p)$ distance of each object (see [12] for the definition of $D_k(p)$ distance), and noutlier objects with largest $D_k(p)$ distance are selected as outliers.

This algorithm shows a relatively good performance, since its time complexity is $O(n \log n)$. Note that this algorithm can be used only for point object. In order to extend this algorithm to non-point objects, we should compute their centroid to convert them to point objects.

The outlier bucket is to be managed independently from other objects. It is stored on secondary memory and loaded into main memory before query processing. To process a spatial query, we must check the outlier bucket loaded in main memory before the access to R-tree.

Since the size of main memory is limited, we allocate a small number of disk pages for the outlier bucket in most cases. Experiments show that we can considerably improve the performance by allocating only one disk page for outlier bucket. Suppose that we allocate one disk page(for example, 4K bytes) for outlier bucket. Then we can store about 341 outliers, which is a sufficient number for most cases. Experiments show that the gain of performance improvement becomes trivial when the number of outliers exceeds a certain threshold.

5. Experiments

For experiments, we prepared real data and synthetic data sets. The real data sets include Seoul and LBCounty data, which contain road intersections, and other important objects in the region, such as buildings. Figure 4 shows the distributions of the real data sets. The numbers of objects of these data sets are 68,736 and 36,548 respectively. And we have also prepared a synthetic data set for uniform distribution consisting of 50,000 points.

We carried out several experiments with these data sets. First, we observed the MBR reduction rate by removing outliers and the resulting performance improvement. Figure 5 shows reduction rates for each data set, where *x*-axis represents the number of outliers removed and *y*-axis indicates the reduction rate. As we expect, no reduction rate is found in the uniform data set, while remarkable reduction rates are achieved in the real data sets. As we increase the number of outliers, the reduction rate increases to a certain limit, but no further improvement can be obtained even we increase the number of

(a) Seoul (b) Long Beach (c) Uniform

Figure 4. Data sets for experiments

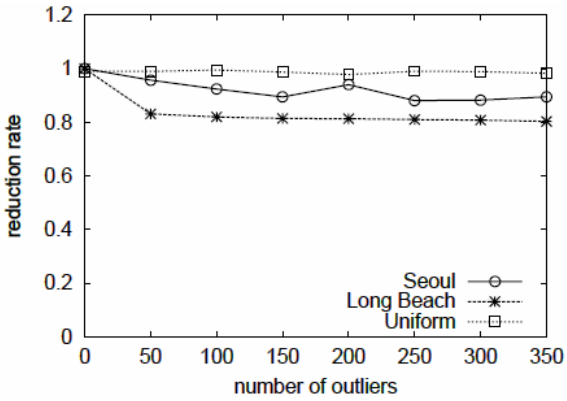
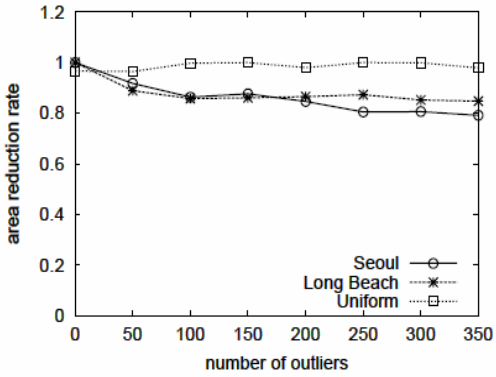


Figure 5. Number of outliers and reduction rates

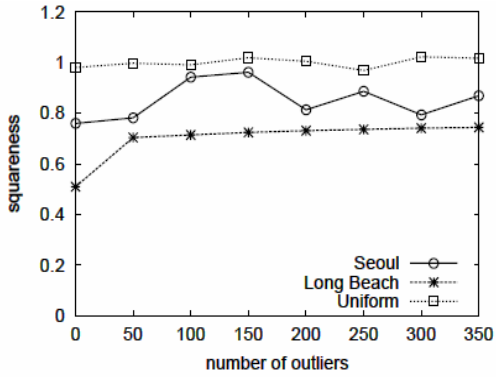
outliers This means that it is recommended to increase the number of outliers to a certain threshold value, $n_{A\,Optimal}$(=250 and 150 for Seoul and LBCounty respectively). According to our experiments, the threshold value is between 0.36 and 0.41 percents of the total objects.

Figure 6(a) shows the relation between the number of outliers and the average area of leaf nodes in R-tree. The area decreases as we increase the number of outliers to a certain threshold, $n_{BOptimal}$(=250 and 150 for Seoul and LBCounty respectively as well) and no further improvement is obtained even though we store more than $n_{BOptimal}$ outliers on outlier bucket. It is because the area of leaf node MBR is strongly related with outliers and reduction rate, as we have expected.

And the relation between the squareness of MBR and outliers is shown by figure 6(b). It is not evident that we could improve the squareness by removing outliers. While the squareness for LBCounty data is apparently



(a) Number of Outlier and Area of MBR



(b) Number of Outliers and Squareness of MBR

Figure 6. Number of outliers, area, and squareness of MBR

improved, it is not evident for Seoul data. We need further studies to analyze this phenomenon.

We counted the number of leaf node accesses to measure the performance of R-tree instead of node accesses. If we would count the number of node accesses including internal nodes, we should take into account other factors, such as splitting strategy, but they are beyond the scope of this paper. The number of leaf node accesses is strongly related with the performance of R-tree and clearly reveals the relationship between the reduction rate and the performance of R-tree.

The tendency of the graphs does not differs from the precedent graphs. The improvement of the performance reaches to a certain limit as we increase the number of outliers to $n_{COptimal}$ and no important improvement is observed. We see 15 percents of performance improvement for small query. However, small improvement (about 3.4 percents) is achieved for large query. But as we have explained in section 3, the outlier effect becomes trivial when the size of query is relatively large compared to the size of MBR.

An interesting observation of the experiments is that the values of $n_{AOptimal}$, $n_{BOptimal}$, and $n_{COptimal}$ are almost identical, which are 250 and 150 for Seoul data and LBCounty data respectively. It means that there is a threshold value which optimizes reduction rate, the area of MBR and the performance of R-tree at the same time. It is however not a surprising fact, since the performance of R-tree is determined by the area of MBRs, which is influenced by reduction rate.

6. Conclusion

Among several factors that determine the performance of R-tree, outliers must be carefully handled to improve the performance. In this paper, we analyzed the relationship between outliers and the performance of R-tree. The conditions for a good R-tree can be summarized as 1) small MBR, 2) small overlapping area and 3) the squareness of MBR. Outliers however enlarge the size of MBR and consequently the overlapping area as well. And they tend to prevent the squareness of MBR. We can improve the performance of by properly handling outliers.

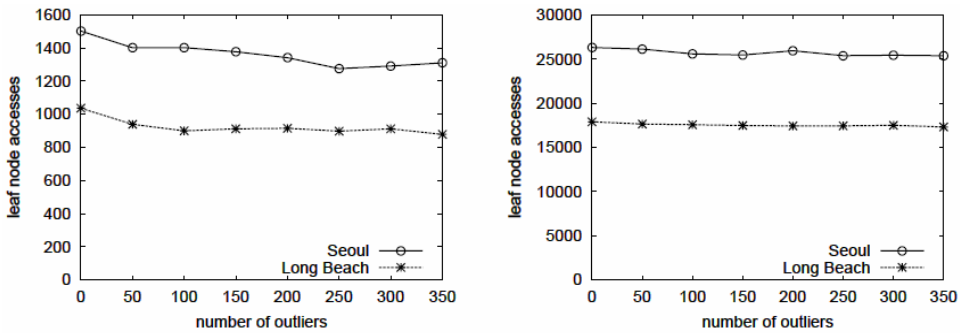
In this paper, we proposed a method for improving the performance of R-tree by properly controlling outliers, which consists of two steps. First we detect k outliers, which can be stored on a prepared buffer, and build R-tree without them. Second, we process spatial query by checking the outlier buffers before accessing R-tree. This simple method enhances the performance. Experiments show that it improves the performance about 15 percents over the original R-tree for small queries.

Acknowledgement

This work was supported by KOSEF grants number 01-2002-000-10014-0 and KOSEF grants number R05-2002-000-01288-0, and by the Program for the Training of Graduate Students in Regional Innovation which was conducted by the Ministry of Commerce, Industry and Energy of the Korean Government.

References

A. Guttman, 1984, R-trees a dynamic index structure for spatial searching, In Proc. SIGMOD Conf, pp.47-57



(a) Query Size(Side Length)=1 % (b) Query Size=(Side Lenth)=10 %

Figure 7. Number of outliers, and number of leaf node accesses

N. Roussopoulos T. Sellis and C. Faloutsos, 1987, The R+-tree: A dynamix index for multi-dimensional objects, In Proc. VLDB Conf, pp.507.518

R. Schneider N. Bechmann, H.-P. Kriegel and B. Seeger, 1990, The R*-tree: An efficient and robust access method for points and rectangles, In Proc. SIGMOD Conf, pp.322-331

B.-U. Pagel, H.-W. Six, H. Toben and P. Widmayer, 1993, Towards an Analysis of Range Query Performance in Spatial Data Structures, In Proc. PODS Conf, pp.214-221

I. Kamel and C. Faloutsos, 1993, On Packing R-trees, In Proc. CIKM Conf, pp.490-499

Y. Theodoridis and T. K. Sellis, 1994, Optimization Issues in R-tree Construction (Extended Abstract), In Proc. IGIS Conf, pp.270-273

C. Faloutsos and I. Kamel, 1994, Beyond Uniformity and Independence: Analysis of R-trees Using the Concept of Fractal Dimension, In Proc. SIGMOD Conf, pp.4-13

M. Barnett and J. Watts, 1994, Building a high-performance collective communication library, In Proc. SC Conf, pp.107-116

S.T. Leutenegger, M.A. Lopez and J.M. Edgington, 1997, STR: A simple and efficient algorithm for R-tree packing, In Proc. ICDE Conf, pp.497-506

E. Knorr and R. Ng, 1998, Algorithms for mining distance-based outliers in large datasets, In Proc. VLDB Conf, pp.392-403

Raymond T. Ng and M. M. Breunig, H.-P. Kriegel and J. Sander, 2000, LOF: Identifying density-based local outliers, In Proc. SIGMOD Conf, pp.93-104

R. Rastogi S. Ramaswamy and K. Shim, 2000, Efficient algorithms for mining outliers from large data sets, In Proc. SIGMOD Conf, pp.427-438

E. Stefanakis Y. Theodoridis and T. K. Sellis, 2000, Efficient cost models for spatial queries using R-trees, In Proc. TKDE Conf, pp.19-32