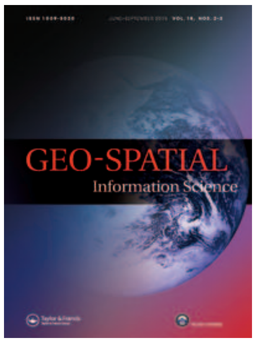

Geo-spatial Information Science

ISSN: 1009-5020 (Print) 1993-5153 (Online) Journal homepage: http://www.tandfonline.com/loi/tgsi20

An accurate selectivity estimation method for window queries and an implementation thereof

Changxiu Cheng, Jing Yang, Xiaomei Song, Shanli Yang & Lijun Wang

To cite this article: Changxiu Cheng, Jing Yang, Xiaomei Song, Shanli Yang & Lijun Wang (2015) An accurate selectivity estimation method for window queries and an implementation thereof, Geo-spatial Information Science, 18:2-3, 81-89, DOI: 10.1080/10095020.2015.1065047

To link to this article: http://dx.doi.org/10.1080/10095020.2015.1065047

Published online: 11 Sep 2015.

Geo-spatial Information Science, 2015
Vol. 18, Nos. 2–3, 81–89, http://dx.doi.org/10.1080/10095020.2015.1065047

An accurate selectivity estimation method for window queries and an implementation thereof

Changxiu CHENG[a,b]*, Jing YANG[a], Xiaomei SONG[b], Shanli YANG[a] and Lijun WANG[c]

[a]*Academy of Disaster Reduction and Emergency Management, Beijing Normal University, Beijing 100875, China;* [b]*State Key Laboratory of Resources and Environmental Information Systems, Institute of Geographic Science and Natural Resources Research, Chinese Academy of Sciences, Beijing 100101, China;* [c]*China Internet Network Information Center, Beijing 100101, China*

(*Received 2 March 2015; final version received 7 April 2015*)

Spatial selectivity estimation is crucial to choose the cheapest execution plan for a given query in a query optimizer. This article proposes an accurate spatial selectivity estimation method based on the cumulative density (CD) histograms, which can deal with any arbitrary spatial query window. In this method, the selectivity can be estimated in original logic of the CD histogram, after the four corner values of a query window have been accurately interpolated on the continuous surface of the elevation histogram. For the interpolation of any corner points, we first identify the cells that can affect the value of point (x, y) in the CD histogram. These cells can be categorized into two classes: ones within the range from $(0, 0)$ to (x, y) and the other overlapping the range from $(0, 0)$ to (x, y). The values of the former class can be used directly, whereas we revise the values of any cells falling in the latter class by the number of vertices in the corresponding cell and the area ratio covered by the range from $(0, 0)$ to (x, y). This revision makes the estimation method more accurate. The CD histograms and estimation method have been implemented in INGRES. Experiment results show that the method can accurately estimate the selectivity of arbitrary query windows and can help the optimizer choose a cheaper query plan.

Keywords: cumulative density (CD) histogram; selectivity estimation; window queries; spatial database; spatial query optimization

1. Introduction

With the increasing accuracy and volume of spatial data needed to be stored and managed, it has become more important to execute queries efficiently on spatial data (*1*). Many improved methods outside the query optimization kernel of the database management system (DBMS) have been explored in recent years, such as compression and storage of spatial data (*2*), multi-scale spatial database (*3–5*), and progressive transmission of spatial data (*6,7*). However, it is also necessary to support spatial query optimization in the kernel of DBMS, requiring the query optimizer to estimate the selectivity and cost of spatial operations, so that it can choose the query execution plan with the least estimated cost (*8*).

Spatial selectivity estimation is crucial in a query optimizer to choose a good execution plan in a given query (*9*). Selectivity estimation attempts to ascertain how many data items will be retrieved (the selectivity) and what the I/O complexity will be in servicing the query. However, the I/O processing costs are usually more dominant than the CPU processing costs in a query (*10*), because I/O processing is a critical bottleneck in the performance of a computer system. Therefore, selectivity is one of the most important parameters in the cost model.

Histograms are the most popular for estimating the query result size in a relational database (*11*). Histogram-based technique works by partitioning the dataset into a small number of subsets called "buckets," and then using approximations for each bucket to model the distribution of the tuples within. Query result estimations are then obtained by processing the query against the buckets and the approximations used therein.

Of the various spatial histograms, the cumulative density (CD) histogram performs the best and gives fairly accurate results for selectivity estimation of window queries (selecting items that overlap a given query window) (*10*), bearing in mind that it can only be adapted to window queries. In recent years, many spatial histograms have been used for spatial selectivity estimation. Most spatial histograms are used for spatial selection operators, i.e., MinSkew (*11*), SQ (*8*), CD (*10*), and Euler (*12–14*); however, some have been used for spatial join operators, i.e., GH (*9*). The CD histogram is the most widely used of these histograms, despite only being applicable to window queries. The first reason for this is that it performs the best and gives fairly accurate results for selectivity estimation. The second reason is that the window query is the most common, most basic, and has also been widely used as the subject of analysis in the other related studies (*15–17*). This article focuses on estimating (analyzing) the selectivity of window queries on spatial databases.

The fairly accurate results from the CD histogram mentioned above are based on the assumption that the query window aligns with the boundary of a grid cell. In

*Corresponding author. Email: chengcx@bnu.edu.cn most applications, query windows rarely align with the boundary of a grid cell, in which case the estimated value may be incorrect (*18,19*). Chi et al. proposed the generalized cumulative density (GCD) (*18*) and generalized cumulative density method based on the Intersection area ratio (GCID) (*19*) algorithms to estimate the selectivity of a query window not aligning with the boundary of a grid cell. The GCID algorithm is finer than the GCD algorithm, because the latter method takes account of the distribution density of spatial data via an array *iArea*(i, j). However, the *iArea*(i, j) array cannot accurately reflect the distribution density of the coverage vector data. This article proposes a more accurate spatial selectivity estimation method based on a CD histogram and gives an implementation thereof in INGRES, which is an open source DBMS. Besides the implementation in this article, PostGIS also integrates a spatial selectivity estimation module in its query optimizer. However, as shown in the experiments in Section 5.1, the selectivity estimation error in PostGIS is high, because of the multiple count problems in its spatial histogram.

The rest of this article is organized as follows. In Section 2, we review the selectivity estimation problem for query windows not aligning with the boundary of a grid cell, and discuss the limitations of the two methods from (*18*). In Section 3, we present a more accurate selectivity estimation method for an arbitrary query window based on a CD histogram and discuss the advantages and disadvantages thereof. Section 4 presents the proposed design and implementation in INGRES. In Section 5, we describe two experiments, the first of which verifies that the proposed method is more accurate than the other five methods mentioned above, while the other confirms that a cheaper query plan can be found using the proposed method implemented in INGRES. Section 6 contains the concluding remarks and further work.

2. Related work

2.1. *CD histogram and its selectivity estimation*

The CD histogram is one of the most important techniques for approximating range query selectivity in spatial databases. Assuming all objects are represented by their MBRs, the CD algorithm can be adapted for browsing applications as follows. Given a grid of R^2 with resolution c, construct four histograms H_{ll}, H_{lr}, H_{ul}, and H_{ur}. The size of each histogram is N with each bucket corresponding to a grid cell. A bucket of H_{ll} keeps the number of lower-left vertices that fall within the bucket. Similarly, H_{lr}, H_{ul}, and H_{ur} keep the counts of the lower-right, upper-left, and upper-right vertices of the objects, respectively. To improve the query efficiency, all the histograms are cumulative, in the sense that a bucket $H(i, j)$ stores the number of vertices in the region(0 0, i j), which covers the range from *cell*(0, 0) to *cell*(i, j). Therefore, for a query (x_a y_a, x_b y_b), the number of intersecting objects can be calculated as follows. Figure 1(a) shows an example of three objects and a query at (x_a y_a, x_b y_b), while Figure 1(b) depicts the four histograms constructed by the CD algorithm. The number of objects intersecting the query is given by: $H_{ll}(x_b, y_b) - H_{lr}(x_a - 1, y_b) - H_{ul}(x_b, y_a - 1) + H_{ur}(x_a - 1, y_a - 1) = 3 - 0 - 1 + 0 = 2$.

However, the fairly accurate results from the CD histogram are based on the assumption that the query window aligns with a boundary of the grid cells. In most applications, the query window rarely aligns with a boundary of the grid cells, in which case, a large error may occur (*18*). Figure 2 shows a query window not aligning with the boundary of a grid cell as depicted by Q. According to the CD estimation algorithm, the window Q should be adjusted to coincide with window Q', giving an estimated selectivity of 2, whereas the real selectivity is 1.

2.2. *Two improved estimation algorithms*

Chi et al. presented the GCD algorithm based on the query window area ratio in Ref. (*18*), as given by Equation (1). This algorithm is perhaps better than the algorithm in Section 2.1. However, there is an implicit assumption in this method that the spatial data is uniformly distributed in the adjusted query window (Q'). This assumption is so unrealistic that it may result in a high selectivity estimation error.

$$S(Q) = (H_{ll}(x_b, y_b) - H_{lr}(x_a - 1, y_b) - H_{ul}(x_b, y_a - 1) + H_{ur}(x_a - 1, y_a - 1)) \times \frac{Area(Q)}{Area(Q')} \tag{1}$$

where $Area(Q)$ and $Area(Q')$ are the areas of the original query window and the adjusted query window, respectively.

To eliminate the assumption mentioned above, Ref. (*19*) proposed the GICD algorithm. It introduces a fifth histogram (*iArea*) in addition to the H_{ll}, H_{lr}, H_{ul}, and H_{ur} histograms. The value of *iArea*(i, j) indicates the area ratio that spatial objects cover in *cell*(i, j). Then, the selectivity of any query window can easily be estimated by Equation (2). Experiments in Ref. (*19*) showed that the estimation results using GICD are more accurate than those using GCD, because *iArea*(i, j) can approximately describe the distribution density of spatial objects.

$$S(Q) = (H_{ll}(x_b, y_b) - H_{lr}(x_a - 1, y_b) - H_{ul}(x_b, y_a - 1) + H_{ur}(x_a - 1, y_a - 1)) \times \frac{\sum_{i=k}^{l}\sum_{j=m}^{n}(iArea(i,j) \times Area_{i,j}(Q))}{\sum_{i=k}^{l}\sum_{j=m}^{n} iArea(i,j)}, \tag{2}$$

where k and l are the cell position values of the x-axis intersecting the query window; n and m are the cell position values of the y-axis intersecting the query window; *iArea*(i, j) is the area ratio of the intersection region between *cell*(i, j) and the objects, and $Area_{i,j}(Q)$ is the

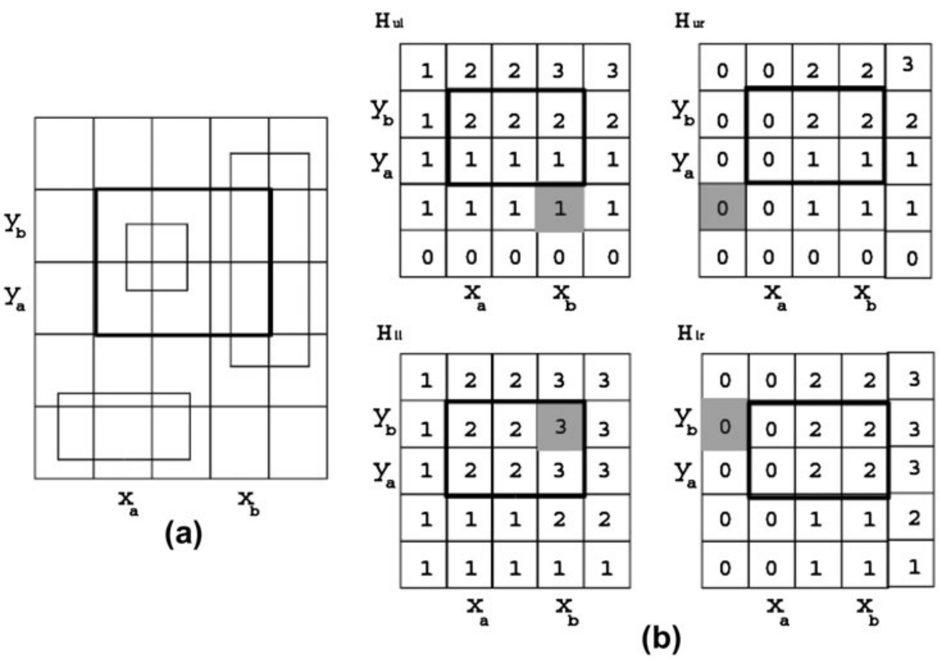

Figure 1. CD histograms (a) and the estimation algorithm (b) (*14*).

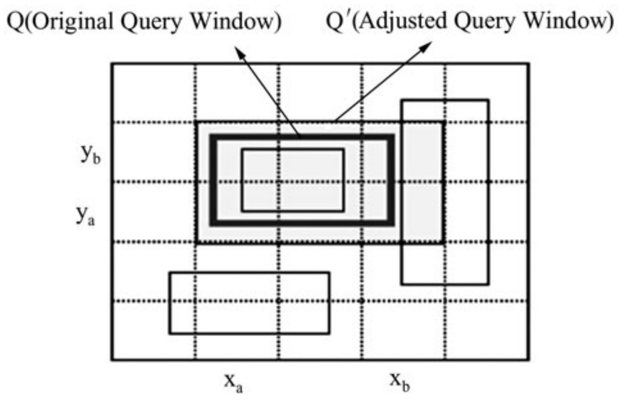

Figure 2. Problem with the CD estimation algorithm (*18*).

area of the intersection region between *cell*(i, j) and the query window.

In fact, it may be better to use the *iArea*(i, j) to represent the spatial distribution density of polygon data, only if their features are discrete distributions and the feature sizes are preferably within the cell size. However, if their features continuously cover most of the area of a map with almost no holes (for example, land-use data), the value of *iArea*(i, j) may not accurately reflect the spatial distribution of the features, because most of the values of *iArea*(i, j) within the coverage data are 1, those disjoint from the coverage data are 0, and the values of *iArea*(i, j) overlapping the boundary of the coverage data may be any value between 0 and 1. Since coverage data is very common in most applications, the GICD histogram is not well suited to coverage vector data, line data, or point data.

3. Accurate selectivity estimation algorithm based on CD histogram

3.1. Accurate spatial selectivity estimation algorithm

An accurate selectivity estimation algorithm can be developed by subtly revising the histogram values of the four corner points of the query window, rather than revising the selectivity of an adjusted query window, as proposed in Ref. (*18*). In past research, the CD histogram was often regarded as discrete raster data to be processed, as shown in Figure 1(b). In fact, every CD histogram should have a continuous elevation surface according to its generation algorithm. Therefore, the value of each cell in Figure 1(b) is not the actual value of the cell, but the value of the upper-right corner of the cell. In such a fine continuous elevation surface, if the histogram values of the four corners of a query window can be obtained by an accurate interpolation algorithm, the selectivity of the query window can be calculated accurately by Equation (3).

$$S(Q) = H_{ll}(x_{\max}, y_{\max}) - H_{lr}(x_{\min}, y_{\max}) - H_{ul}(x_{\max}, y_{\min}) + H_{ur}(x_{\min}, y_{\min}) \tag{3}$$

where ($x_{\min}$, $y_{\min}$) are the coordinates of the lower-left corner of the query window, and ($x_{\max}$, $y_{\max}$) are the coordinates of the upper-right corner of the query window.

According to certain concepts of CD histograms, the value of any point (x, y) in the histogram should be the number of corresponding vertices that fall within the region (0 0, x y). Taking H_{ll} and point (2.7, 4.5) in Figure 3 as an example, the value of H_{ll}(2.7, 4.5) should be the number of lower-left vertices that fall within the

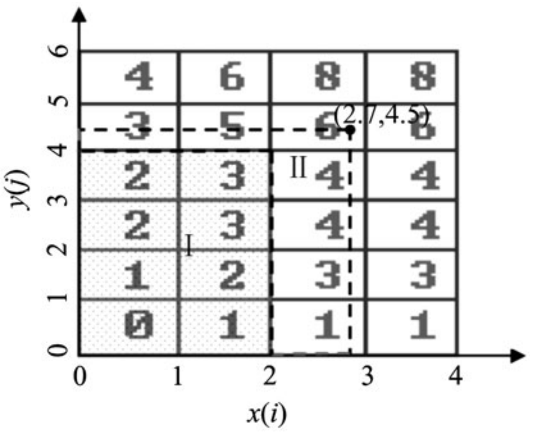

Figure 3. Example of how to estimate the value of point (2.7, 4.5) in H_{ll} histogram.

region(0 0, 2.7 4.5). If the region (0 0, 2.7 4.5) is divided into two parts, part I and part II as indicated in Figure 3, the value of $H_{ll}(2.7, 4.5)$ would be the sum of the number of vertices falling within part I and part II, respectively. The number of vertices in part I is known, $H_{ll}(\lfloor 2.7 \rfloor, \lfloor 4.5 \rfloor) = 3$. The number of vertices in part II, however, is uncertain, but can be estimated by the number of vertices falling within every cell covered by part II and its corresponding area ratio occupied by part II, as given by the last three terms in Equation (4). Thus, the value of any point (x, y) in the H_{ll} histogram could be interpolated by Equation (4). Equation (5) expresses how to deduce the number of lower-left vertices falling within $C_{ll}(\lceil x \rceil, \lceil y \rceil)$, and can be accurately calculated according to the cumulative concepts of the CD histogram. According to Equations (4) and (5), $H_{ll}(2.7, 4.5)$ in Figure 3 should be $3 + (4 - 3) \times (2.7 - 2) + (5 - 3) \times (4.5 - 4) + (6 - 5 - 4 + 3) \times (2.7 - 2) \times (4.5 - 4) = 4.7$. The other values of any point (x, y) in H_{lr}, H_{ul}, or H_{ur} can be obtained by similar logic to that given in Equations (4) and (5).

3.2. *Advantages and disadvantages of the proposed method*

The proposed method is more accurate than the two probability model methods proposed in Ref. (*18*). The accuracy of the method relies on the fact that the CD histograms are taken to have smooth elevation surfaces, rather than ladder-like surfaces. Moreover, in a smooth histogram surface, the value of any point can be divided into a very certain value corresponding to part I and some uncertain values in part II of Figure 3. In the method, every uncertain value is estimated by the number of vertices within its cells and the corresponding area ratio occupied by part II. It is thus obvious that the method maintains accuracy at every step without any additional assumptions and information.

This method can be adapted to many types of vector data and requires less storage than the GICD algorithm. The method directly deduces a point's distribution density from the CD histogram rather than introducing some other variable to record it. Thus, additional storage is unnecessary in this method. In addition, the method converts the distribution density of spatial data into the distribution density of some of the vertices. Furthermore, the concept of the distribution density of some vertices is the same for discrete polygon data, continuous polygon data, line data, or point data. Conversely, $iArea(i, j)$ in the GICD method is a concept tied to polygon data, and is most useful for small polygons with discrete distributions.

The time complexity of the proposed estimation method is a little higher than the original CD estimation method and the GCD method, but it is sometimes more efficient than the GICD method. Since there is no loop clause in the original CD method and the GCD method, their complexities are almost $O(1)$. The loop in the GICD method is focused on accumulating the probability of the area, as shown in Equation (2), and thus its complexity should be $O((\lfloor x_{\max} \rfloor - \lfloor x_{\min} \rfloor) \times \lfloor y_{\max} \rfloor - \lfloor y_{\min} \rfloor)$. However, the complexity of this method could be maintained on a constant level according to the last part of Equation (4). Therefore, its complexity is better than the GICD algorithm.

$$\begin{aligned} H_{ll}(x,y) &= H_{ll}(\lfloor x \rfloor, \lfloor y \rfloor) + \sum_{i=1}^{\lfloor y \rfloor} C_{ll}(\lceil x \rceil, i) \times (x - \lfloor x \rfloor) + \sum_{j=1}^{\lfloor x \rfloor} C_{ll}(j, \lceil y \rceil) \times (y - \lfloor y \rfloor) + C_{ll}(\lceil x \rceil, \lceil y \rceil) \times (x - \lfloor x \rfloor) \times (y - \lfloor y \rfloor) \\ &= H_{ll}(\lfloor x \rfloor, \lfloor y \rfloor) + (H(\lfloor x \rfloor + 1, \lfloor y \rfloor) - H(\lfloor x \rfloor, \lfloor y \rfloor)) \times (x - \lfloor x \rfloor) \\ &\quad + (H_{ll}(\lfloor x \rfloor, \lfloor y \rfloor + 1) - H_{ll}(\lfloor x \rfloor, \lfloor y \rfloor)) \times (y - \lfloor y \rfloor) + C_{ll}(\lceil x \rceil, \lceil y \rceil) \times (x - \lfloor x \rfloor) \times (y - \lfloor y \rfloor); \end{aligned} \tag{4}$$

$$\begin{aligned} C_{ll}(\lceil x \rceil, \lceil y \rceil) = H_{ll}(\lceil x \rceil, \lceil y \rceil) - H_{ll}(\lceil x \rceil - 1, \lceil y \rceil) \\ - H_{ll}(\lceil x \rceil, \lceil y \rceil - 1) \\ + H_{ll}(\lceil x \rceil - 1, \lceil y \rceil - 1); \end{aligned} \tag{5}$$

where $\lfloor x \rfloor$ and $\lfloor y \rfloor$ are the floor of the x and y values, respectively; $\lceil x \rceil$ and $\lfloor y \rfloor$ are the ceilings of the x and y values, respectively; $C_{ll}(\lceil x \rceil, \lceil y \rceil)$ is the number of lower-left vertices falling within $Cell(\lceil x \rceil, \lceil y \rceil)$.

4. Implementation of the selectivity estimation algorithm

The CD histogram and the accurate estimation method have been implemented and integrated into the optimizer facility (OPF) module of INGRES. Previously, the OPF module gave a default selectivity ratio (such as 50%) to all nodes without data statistics including spatial prediction node. This implementation improved the selectivity ratio of spatial prediction is close to the real value.

Based on the INGRES object-relation extension mechanism – object management extension (OME), some related functions for spatial prediction cost estimation have been developed. Before embarking on this work, our team developed some spatial data types (i.e., *ST_Geometry*) and corresponding functions (i.e., *ST_Intersect*, *ST_MBRIntersect*, *ST_GeomFromText*) through the user definition mechanism of the OME. Using this framework, we coded some functions to manage a CD histogram and to obtain the selectivity estimation. *ST_BldHist*, *ST_ExpHist* and *ST_DelHist* were used to build, export, and delete a certain CD histogram for a spatial table, while *ST_QESTHist* was used to estimate the selectivity in a certain query window. In addition, we created a system catalog named *IISTSTATISTICS* to store CD histograms built by *ST_BldHist SQL function*. The SQL to create *IISTSTATISTICS* is given below.

CREAT TABLE *IISTSTATISTICS*

(SHTABBASE	INT4 NOT NULL,	// the internal identity of a base table with spatial histogram
SHTABINDEX	INT4 NOT NULL,	// the internal index identity of the base table
SHATTRNO	INT4 NOT NULL,	// the internal identify of special attribute with spatial histogram
SHNULL	FLOAT8,	// whether spatial histogram is NULL or not
SHSAMPLE	FLOAT8,	// whether spatial histogram is sample or not
SHREDUNDANCY	FLOAT8,	// redundancy rate of the spatial histogram
SHRANGE	ST_GEOMETRY,	// geometric extent of the spatial data to be built as histogram
GEODIMENSION	INT4,	// dimensions of spatial data
CELLPARAM	BLOB,	// parameters to describe histogram, i.e. cell numbers in each different dimension
SHTYPE	INT4,	// type of histogram, i.e. CD, Euler, or others
CELLVALUES	BLOB,	// cell values of the spatial histogram
SHVERSION	CHAR(16))	// version info of the spatial histogram

Having developed these basic histogram functions, we needed to find the correct place to add a switch, callback the *ST_QESTHist* function to estimate the spatial selectivity in that switch, and input the selectivity into the cost estimation module. The cost evaluation procedure of a query plan proceeds from the bottom node to the top node in a query plan tree. Therefore, we needed to insert some code in the nodes' cost evaluation module and equate the estimated selectivity to the corresponding input parameters of the cost evaluation module. In the OPF module, *oph_bfcost* is the function to obtain the selectivity of every node with bool factors. In query parser PSF (Parser Facility), we determined and added some masks to identify whether the prediction is spatial related. Therefore, if the predictions are the right spatial predictions, such as *ST_intersect* and *ST_disjoint* operator, cost-based optimizer find the related parameters from query optimization global structure (*subquery)* and callback the *ST_QESTHist* function. Note that the selectivity of *ST_disjoint* should be 1 minus the selectivity of the corresponding *ST_intersect* operator, because they are complementary. Thus, the selectivity returned by *ST_QESTHist* will be equal to the *bp- > opb_selectivity* variable of the corresponding bool factor in *subquery*, which is a very important parameter in the cost evaluation of the plan tree.

Having made the changes mentioned above, the CD histogram and estimation method for the *ST_Intersects* and *ST_Disjoint* operators could be implemented in the OPF module of INGRES by the logic mentioned in Section 3.

5. Experiments

Two experiments are described in this section. One confirms that the proposed method has high selectivity estimation accuracy after comparing the estimation errors from the five methods discussed in Sections 1–3. The other shows that the accurate selectivity estimation assists the OPF module in making the right decision in selecting a query plan after comparing these spatial query plans and their efficiency with a CD histogram.

5.1. Accuracy experiment

The test data for this experiment was land-use data of a Chinese county on a 1:10,000 scale, stored in an INGRES database. We randomly defined eight query windows on the land-use data to test the accuracy of their selectivity estimation. Figure 4 shows the land-use data with eight query windows demarcated by the dotted lines and appropriately labeled with numbers.

First, we used the *ST_BldHist* function to build CD histograms with 20 rows and 20 columns. The corresponding geographic structure query language (GSQL) is given below.

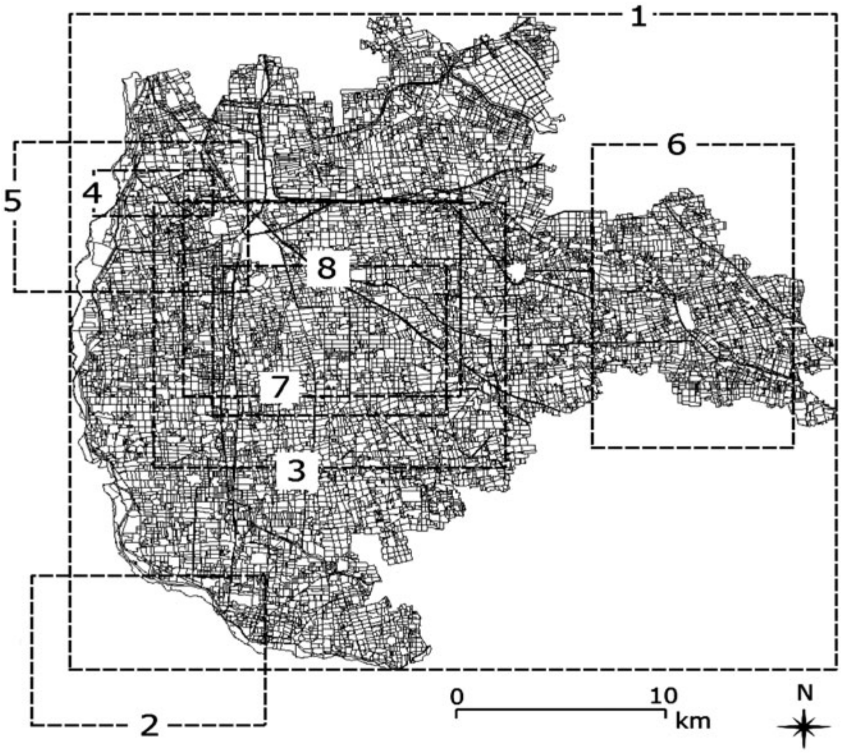

Figure 4. Land-use data and the eight query windows defined randomly.

```
SELECT ST_BldHist ('INGRES', 'landuse', 'geometry',
ST_GeomFromText ('POLYGON((487898.531
253069.641, 531754.938 253069.641, 531754.938
296054.656, 487898.531 296054.656, 487898.531
253069.641))'), 1, '20, 20', 0, 0).
```

Then, we used the ST_*QESTHist* function to obtain the selectivity estimation value. Taking query window 4 as an example, the GSQL to obtain its estimation selectivity is the following:

```
SELECT ST_QESTHist ('INGRES', 'landuse', 'geome-
try', st_GeomFromText ('POLYGON((484695.007
277840.242, 498046.113 277840.242, 498046.113
287640.369, 484695.007 287640.369, 484695.007
277840.242))', intersect, 32631)).
```

After obtaining the selectivity estimation values for the eight query windows, we obtained the actual value by executing the GSQL to find the geometry whose MBR intersects the query window, rather than finding the geometry that intersects the query window. The reason for this is that the query optimizer tries to ensure that the estimation value approximates the number of candidate objects rather than the number of objects in the final results, because the selectivity estimation focuses mainly on the filtering step, and the I/O cost is determined predominantly by the number of candidate objects in the filtering step (*8,9*) rather than the number of objects in the final results. Therefore, using query window 4 as an example, the GSQL to obtain the actual selectivity is the following:

```
SELECT Count (*) FROM landuse

WHERE landuse.geometry ST_MBRIntersects ST_
GeomFromText('POLYGON((484695.007 277840.242,
498046.113 277840.242, 498046.113 287640.369,
484695.007 287640.369, 484695.007 277840.242))').
```

After obtaining the estimation and real selectivity of the eight query windows, we used the relative error ratio ((*Estimation value* − *Real value*) / *Real value*) to indicate the estimation accuracy.

To compare the accuracy of the proposed model with the other four methods, we implemented the original estimation method and the two methods by Chi et al. (*18,19*), obtained the estimation values and actual values of the eight query windows, and calculated their relative error ratios in the same way. In addition, we also repeated the same experiment in POSTGIS, where we used the *build_histogram2d* and *estimate_histogram2d* functions to build a spatial histogram and estimate its selectivity. Figure 5 shows the relative estimation error ratios of the five methods for the eight query windows randomly defined in Figure 4.

5.2. *Discussion about the accuracy*

Figure 5 shows that the proposed method achieves the best estimation accuracy of the five methods. On the whole, the relative error ratio of this method is steady and much less than 10%, which is the lowest of the five methods. It is obvious that the relative error ratio of the

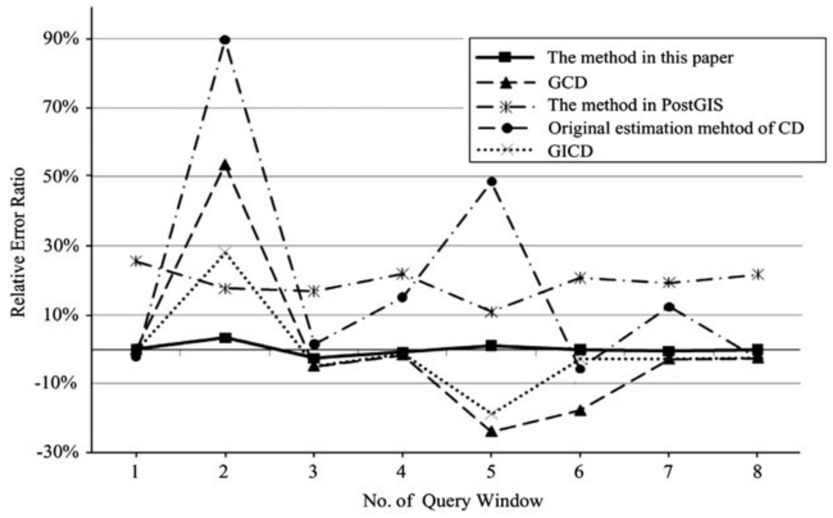

Figure 5. Relative estimation error ratios of the five methods on querying the test data in the eight query windows.

method in PostGIS is significantly higher than that of the proposed method. In addition, the average error ratio of the other three methods is also higher than the ratio of the proposed method.

Statistic results of Chi et al's methods show that the GICD method is better than the GCD method for query windows overlapping the boundary of the land-use data; however, their selectivity estimations are often the same on a query window within the boundary of the land-use data. If the query window overlaps the boundary of land-use data (i.e., windows 2 and 5), the *iArea*(*i*, *j*) within the query window would have different values. In this case, the *iArea*(*i*, *j*) would be work in the selectivity estimation. However, if the query window overlaps the boundary of land-use data (i.e., windows 7 and 8), all the *iArea*(*i*, *j*) within the query window are 1. Therefore, the selectivity estimations of the GCD and GICD method are the same. This is discussed in the last paragraph of Section 2.2.

In addition, we found that the selectivity estimation error may be greater if the query window is very small, because a smaller cardinal number results in a larger relative error.

5.3. *Experiment to investigate the effect of selecting a query plan*

This experiment investigates whether a cheaper query plan is chosen after the selectivity estimation in Section 3 has been implemented in INGRES. To enlarge the enumeration space of the query plan, we downloaded the Alaska data from the Quantum GIS website and chose *airports*, *trees*, and *builtups* tables of Alaska to create a more complex GQL. The *airports* table, as a point layer, stored 76 airports in Alaska, the *trees* table, as a polygon layer, stored 444 tree polygons, while the *builtups* table, as a polygon layer, stored 18 built-up zones. We tried to find some combination of *name* in *airports*, *f_code* in *trees*, and *name* in *builtups*, which simultaneously satisfied the following conditions. First, the *area* of *trees* should be less than 1000 km^2, the *airports* should intersect with a polygon described by the following GSQL and the *builtups*' name should not be "NOME." Furthermore, the distance between *trees* and *builtups* and the distance between *airports* and *builtups* should be less than 50,000 and 40,000 m, respectively. The appropriate GSQL is given below.

```
SELECT * FROM airports, trees, builtups

WHERE trees.area_km2 < 1000

AND airports.geometry ST_Intersects ST_GeomFrom-
Text('POLYGON((500000 3000000, 500000 5000000,
1500000 5000000, 1500000 3000000, 500000
3000000))')

AND builtups.name < > 'NOME'

AND ST_Distance(trees.geometry, builtups.geometry)
< 50000

AND ST_Distance(airports.geometry, builtups.geometry)
< 40000.
```

There are three bool factors in this query: two for the attribute columns (i.e., *trees.area_km*^{2} < 1000 and *builtups.name* < > 'NOME') and one for the geometry column (i.e., *airports.geometry* ST_Intersect ST_GeomFromText('POLYGON((500000 3000000, 500000 5000000, 1500000 5000000, 1500000 3000000, 500000 3000000))')). We use the *Optimizedb* command to build two attribute histograms for the *area_km*^{2} column of the *trees* table and the *name* column of the *builtups* table. This will help the OPF module obtain the correct selectivity of the two bool factors with attributes.

We executed the GSQL given above without a spatial histogram. The INGRES DBMS chose a query plan tree

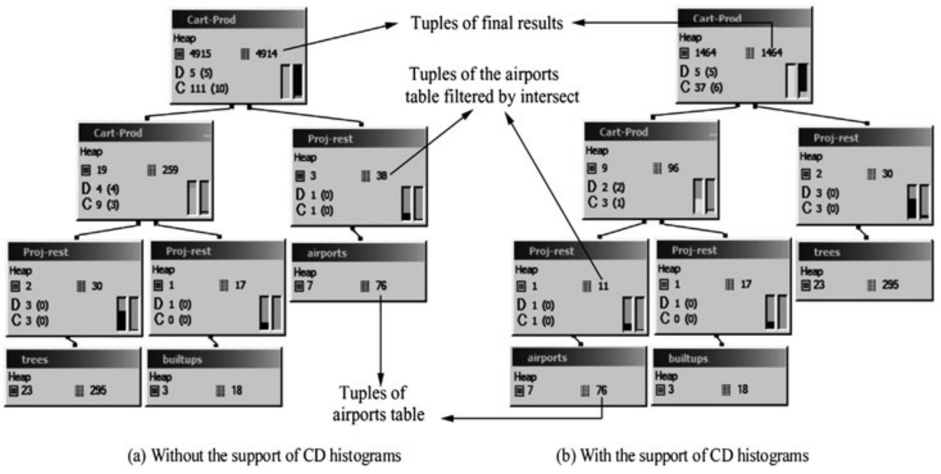

Figure 6. Different query plans to be chosen under different situations.

as shown in Figure 6(a). First, it created a Descartes production between the project restriction results of the *trees* table and the project restriction results of the *builtups* table. Then, it created a Descartes production between the production results in the previous step and the project restriction results of the *airports* table. After obtaining the actual value of every query node by setting a trace point as *QE90*, we found that the selectivity of the two attribute columns, as illustrated by the two project restriction nodes above the *trees* and *builtups* tables in Figure 6(a), is almost the same as their actual values. However, as the project restriction node above the *airports* table shows, the spatial selectivity is 38, which is the product of the tuples of the *airport* table (76) and the default ratio (50%) given by INGRES. Obviously, it is far deviated from the actual value of 11.

To repeat this query and produce statistics of the execution times, we set trace points as *DM421* beforehand to prevent the second execution getting data directly from the table already in the DMF (data manipulation facility) cache. After setting the trace point, we repeatedly executed that GSQL 10 times. The average execution time and standard variance was 27,996.2 ms and 0.08, respectively.

Thereafter, we used the *ST_BldHist* function to create CD histograms with 20 columns and 20 rows for the *airports* table. The corresponding GSQL is the following:

```
SELECT ST_BldHist ('chengcx', 'airports', 'geometry',
ST_GeomFromText ('POLYGON((–4789817 976645
5471489 976645, 5471489 7564794, −4789817
7564794, −4789817 976645))'), 1, '20, 20', 0, 0).
```

With the support of the spatial histogram, the INGRES DBMS chose the query plan tree as shown in Figure 6(b), which is different to that in Figure 6(a). It first created a Descartes production between the project restriction results of the *airports* table and the project restriction results of the *builtups* table. Then, it created a Descartes production between the production results in the previous step and the project restriction results of the *trees* table. It is obvious that the spatial selectivity in the project restriction node above the *airports* table is the same as the actual value 11.

We also set a trace point as *DM421* and repeatedly executed the GSQL 10 times in the same way. The average execution time and standard variance were 19,009.4 ms and 0.078, respectively. Thus, the execution efficiency of INGRES improved significantly with the use of the accuracy estimation.

5.4. *Discussion about the effect*

According to the experiment in Section 5.3, a cheaper query plan is chosen with the support of CD histograms and the accurate selectivity estimation. The important difference between the two query plans lies in the selectivity between the two dotted line rectangles. Without the use of CD histograms, the selectivity was 38. However, with the support thereof, the tuples of the *airports* table filtered by the intersect operator were estimated to be 11, which is the same as the actual value. Accurate selectivity, as an important parameter of the cost estimation model, helps the OPF module choose the cheaper plan tree in Figure 6(b), which ultimately saves the computation time at almost 9000 ms (27,996.2–19,009.4 ms).

This method not only guarantees the selectivity estimation accuracy of a leaf node with intersect or disjoint restriction in the query plan, but also the selectivity estimation accuracy of the final result. Although a window query is the most common and most basic operator in a spatial database, it cannot be ignored that there may be other operators in a spatial query, for example, the spatial join operator based on the *ST_Distance* operator in the Carte-Prod node of Figure 6. At present, the selectivity estimation method of the *ST_Distance*-based spatial join has not been implemented. Thus, its selectivity in Figure 6 was estimated by the traditional Descartes production estimation of INGRES. This is the reason that the tuples in the final result may differ significantly from the actual tuples.

6. Conclusions and future work

In this article, we propose an accurate selectivity estimation for window queries based on CD histograms, and implemented it in INGRES. This method can accurately estimate the selectivity of arbitrary query windows without additional storage. In addition, this method helps the OPF module make the correct decision in choosing a query plan, because it ensures that the query cost estimation closely matches the actual execution situation. The implementation makes a compact integration between the DBMS and the spatial histogram.

This article only discusses the selectivity estimation for window queries, which are the most common. However, there are various other spatial operations in a spatial database. To ensure that the estimation cost of the spatial query close resembles the actual situation, further work needs to be done, such as, the selectivity estimation about a spatial join. The spatial join is an important basic operation in a multi-table spatial query. In addition, it is also very important to estimate the selectivity of finer topology predications. This article only implemented the selectivity of intersect and disjoint predications. However, overlap, contain, within, and so on, may also be commonly used in a complex spatial query.

Funding

This work was supported by the National Natural Science Foundation of China [grant numbers 41222009, 41271405].

Notes on contributors

Changxiu Cheng is a professor at Beijing Normal University. Her past affiliations were with Beijing Institute University and China Agriculture University from where she received a bachelor's degree in computer science in 1997 and a doctor degree in soil science in 2001. Her research interests include multi-scale representation, spatial database query optimization, and GIS application in disaster.

Jing Yang is a visit student of Beijing Normal University. Her past affiliation was with Inner Mongolia Normal University from where she received a bachelor's degree.

Xiaomei Song is an engineer of Yongyou Software Company. His past affiliation was with Institute of Geographic Science and Natural Resources Research from where he received a doctor degree.

Shanli Yang is a visit student of Beijing Normal University. He will graduate from Liaoning Normal University in 2015.

Lijun Wang is an engineer of China Internet Network Information Center. His past affiliation was with China Agriculture University from which he received a master's degree in 2001.

References

(1) Jiang, S.; Lee, B.S.; He, Z. Cost Modeling of Spatial Operators Using Non-parametric Regression. *Inf. Sci.* **2007**, *177* (2), 607–631.
(2) Bjørke, J.T.; Nilsen, S. Wavelets Applied to Simplification of Digital Terrain Models. *Int. J. Geograp. Inf. Sci.* **2003**, *17* (7), 601–621.
(3) Oosterom, P.; Schenkelaars, V. The Development of an Interactive Multi-scale GIS. *Int. J. Geograp. Inf. Syst.* **1995**, *9* (5), 489–507.
(4) Pasher, S.; Zhou, X. Efficient Update and Retrieval of Objects in a Multi-resolution Geospatial Database. *Proceedings of SSDBM*, Cambridge: IEEE Computer Society, **2003**, pp 193–201.
(5) Cheng, C.; Niu, F.; Cai, J.; Zhu, Y. Extensions of GAP-tree and Its Implementation Based on a Non-topological Data Model. *Int. J. Geograp. Inf. Sci* **2008**, *22* (6), 657–673.
(6) Bertolotto, M.; Egenhofer, M.J. Progressive Transmission of Vector Map Data over the World Wide Web. *GeoInformatica* **2001**, *5* (4), 345–373.
(7) Cheng, C.; Lu, F.; Cai, J. A Quantitative Scale-setting Approach for Building Multi-scale Spatial Databases. *Comput. Geosci.* **2009**, *35* (11), 2004–2209.
(8) Aboulnaga, A.; Chaudhuri, S. Self-tuning Histograms: Building Histograms Without Looking at Data. *Proceedings of 1999 ACM SIGMOD*, New York: ACM Press, **1999**, pp 181–192.
(9) An, N.; Yang, Z.Y.; Sivasubramaniam, A. Selectivity Estimation for Spatial Joins. *Proceedings of the 17th International Conference on Data Engineering*, Heidelberg: IEEE Computer Society, **2001**, pp 368–375.
(10) Jin, J.; An, N.; Sivasubramaniam, A. Analyzing Range Queries on Spatial Data. *Proceedings of the 16th International Conference on Data Engineering*, San Diego, CA: IEEE Computer Society, **2000**, pp 525–534.
(11) Acharya, S.; Poosala, V.; Ramaswamy, S. Selectivity Estimation in Spatial Database. *Proceedings of 1999 ACM SIGMOD*, New York: ACM Press, **1999**, pp 13–24.
(12) Beigel, R.; Tanin, E. The Geometry of Browsing. *Proceedings of the Latin American Symposium on Theoretical Informatics*, Campinas: Springer, **1998**, pp 331–340.
(13) Sun, C.; Agrawal, D.; Abbadi, A. Selectivity for Spatial Joins with Geometric Selections. *Proceedings of EDBT*, Prague: Springer, **2002**, pp 609–626.
(14) Sun, C.; Agrawal, D.; Abbadi, A. Exploring Spatial Datasets with Histograms. *Distrib. Parallel Databases* **2006**, *20* (1), 57–88.
(15) Theodoridis, Y.; Papadias, D. Range Queries Involving Spatial Relations: A Performance Analysis. *Proceedings of COSIT'95*, Semmering: Springer, **1995**, pp 537–551.
(16) Theodoridis, Y.; Sellis, T.K. A Model for the Prediction of R-tree Performance. *Proceedings of the 15 ACM SIGACT_SIGMOD-SIGART Symposium on Principles of Database Systems*, Montreal: ACM Press, **1996**, pp 161–171.
(17) Proietti, G.; Faloutsos, C. I/O Complexity for Range Queries on Region Data Stored Using an R-tree. *Proceedings of the 15th International Conference on Data Engineering*, Sydney: IEEE Computer Society, **1999**, pp 628–635.
(18) Chi, J.H.; Kim, S.H.; Ryu, K.H. Selectivity Estimation Using the Generalized Cumulative Density Histogram. *Proceedings of ASGIS*, Chongqing, **2004**, pp 201–207.
(19) Chi, J.H.; Kim, S.H.; Ryu, K.H. Spatial Selectivity Estimation Using Compressed Histogram Information. *Lect. Notes Comput. Sci.* **2005**, *3399*, 489–494.