Multiscale Histograms: Summarizing Topological Relations in Large Spatial Datasets

Xuemin Lin, University of NSW, Sydney, Australia

Qing Liu, University of NSW, Sydney, Australia

Yidong Yuan, University of NSW, Sydney, Australia

Xiaofang Zhou, University of Queensland, Brisbane, Australia

Abstract

Summarizing topological relations is fundamental to many spatial applications including spatial query optimization. In this paper, we present several novel techniques to effectively construct cell density based spatial histograms for range (window) summarizations restricted to the four most important topological relations: contains, contained, overlap, and disjoint. We first present a novel framework to construct a multiscale histogram composed of multiple Euler histograms with the guarantee of the exact summarization results for aligned windows in constant time. Then we present an approximate algorithm, with the approximate ratio 19/12, to minimize the storage spaces of such multiscale Euler histograms, although the problem is generally NP-hard. To conform to a limited storage space where only k Euler histograms are allowed, an effective algorithm is presented to construct multiscale histograms to achieve high accuracy. Finally, we present a new approximate algorithm to query an Euler histogram that cannot guarantee the exact answers; it runs in constant time. Our extensive experiments against both synthetic and real world datasets demonstrated that the approximate multiscale histogram techniques may improve the accuracy of the existing techniques by several orders of magnitude while retaining the cost efficiency, and the exact multiscale histogram technique requires only a storage space linearly proportional to the number of cells for the real datasets.

1 Introduction

Research in spatial database systems has a great impact on the technical support to many applications, such as geographic information systems, digital libraries, robotics, image processing, CAD and VLSI. In the last 20 years, spatial indexing and searching have drawn a great deal of attention from research society, and a number of techniques have been developed [6]. With the availability [9, 20, 21] of a huge collection of on-line spatial data, there are strong demands for effective techniques to support efficient browsing of large datasets to summarise spatial characteristics. It becomes extremely important in large digital libraries/archives to support *interactive* queries by *query preview* [3, 9]. These applications require a system to provide a fast summary information to users for quickly identifying relevant data among enormous available data resources. Summarizing spatial datasets also plays an important role in spatial query processing optimization by providing selectivity estimation [1, 2, 14, 19].

A variety of techniques [7, 8] have been recently developed for effectively summarizing data in relational datasets. The most common techniques are *samples* [16], *histograms* [18, 7], *wavelets* [17], and *sketches* [7]. In contrast, techniques for summarizing topological relations against spatial datasets are relatively a little.

In this paper, we will investigate the problem of summarizing *rectangular* objects for range (window) queries. Several histogram based summarization techniques have recently been developed to provide selectivity estimation for spatial range queries. The existing techniques may be divided into two categories: 1) *data partition techniques*, and 2) *cell density*. The Min-skew algorithm in [2] and the SQ-histogram technique in [1] belong to the first category, and propose to group "similar" objects together according to some mathematic models to form a bucket for estimating the number of *disjoint* objects and the number of *non-disjoint* objects in window queries.

Techniques based on cell density [3, 14, 20] propose to divide the object space into a number of *disjoint* cells, and to record some kind of object density for each cell. To estimate the number of non-disjoint objects against a window, a cumulative density based approach (CD) was proposed in [14], while the *Euler* formula [11] has been effectively used in [3] in creating a cell density based histogram (called Euler histogram). Sun, Agrawal, and El Abbadi [20] substantially extended the Euler histogram techniques to summarizing the 4 most important topological relations: "contains", "contained", "overlap", and "disjoint" (to be defined in section 2) by dividing the non-disjoint relation into the three relations: contains, contained, and overlap.

It has been demonstrated in [14] that a cell density

Proceedings of the 29th VLDB Conference, Berlin, Germany, 2003

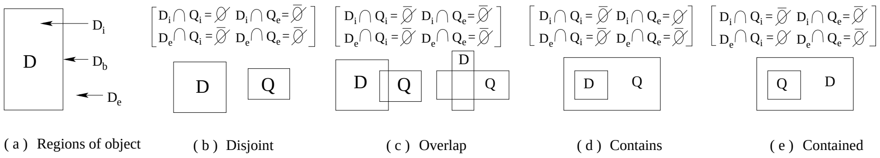


Figure 1: 4 Topological Relations between two Objects

based technique may perform better than data partition techniques. Further, the existing data partition techniques dealt with only the two topological relations: disjoint and non-disjoint.

In a cell density approach, summarizing rectangular objects against an arbitrary window is usually processed against the related *aligned windows* which consist of whole cells only. Therefore, the summarization accuracy over aligned windows is fundamental to the quality of a cell density based histogram. As with the conventional histogram techniques in relational datasets, a cell density based histogram is *practical* if its storage space is linearly proportional to the number of the disjoint cells to be used.

The histogram techniques in [3, 14, 20] are practical, and guarantee the exact solutions for estimating the number of disjoint and non-disjoint objects, respectively, against aligned windows. To the best of our knowledge, [20] is the only paper investigating the problem of summarizing the 4 topological relations above. The techniques in [20] are practical but rely on a few strong assumptions; the accuracy may greatly degrade if the spatial data do not conform to the assumptions. To resolve the deficiency, in this paper, we proposed a novel multiscale paradigm by effective utilization of the object scales (to be precisely defined in section 2). It consists of two steps: 1) group together the objects with "similar" scales, and 2) for each group of objects, create an Euler histogram. The main contributions of the paper may be summarized below.

1. We proposed a novel framework to construct a multiscale histogram that is composed of multiple Euler histograms. Such a multiscale histogram can generate exact summarization results for aligned windows in constant time restricted to the four topological relations.
2. We investigated the problem of storage space minimization in a multiscale histogram. Although the problem is generally NP-hard, our approximate algorithm, based on the *Duh-Furer semi-local* optimisation technique [4], can guarantee the approximate ratio 19/12.
3. To conform to a limited storage space where only k Euler histograms are allowed, we present an effective algorithm to construct multiscale histograms with high accuracy.
4. Finally, we present a new approximate algorithm to query an Euler histogram that cannot guarantee the exact answers; it runs in constant time.

We evaluate our new techniques by both synthetic and real world datasets. Our experiment results demonstrated that the approximate multiscale techniques may improve the accuracy of the existing techniques by several orders of magnitude while retaining the cost efficiency. The experiments also showed that the exact multiscale histogram technique requires only a storage space linearly proportional to the number of disjoint cells for the real world datasets, and thus is practical.

The rest of the paper is organized as follows. In section 2, we provide preliminaries and related works. In section 3, we present our first and second contributions of the paper - efficient histogram construction algorithms for generating exact summarization results against aligned windows while minimizing the storage space. Section 4 presents the third and fourth contributions of the paper. Section 5 presents the applications, possible generalizations, and the maintenance of our new histogram techniques. Section 6 presents the experiment results. This is followed by conclusion and remarks.

2 Preliminaries

In this section, we give a brief overview of Euler histograms [3], and the three algorithms (S-Euler, EulerApprox, and M-Euler) [20] to query an Euler histogram. These techniques are closely related to our work in this paper.

In this paper, we study only *axis-aligned* rectangular objects; this is because different types of objects can be represented by their minimum bounding rectangles (MBR) to approximate the spatial extents. A set S of objects, in this paper, always means a set of axis-aligned rectangles.

A binary topological relation between two objects, D and Q, is based upon the comparison [5] of D's *interior* D_i, *boundary* D_b, *exterior* D_e (see Figure 1(a)) with Q's interior, boundary, and exterior. It can be classified [5, 10, 20] into 8 high-resolution topological relations according to the 9-intersection model, and can be also classified into the 5 medium-resolution topological relations by omitting 3 less important relations. In this paper, we focus only on the 4 important medium-resolution topological relations (as depicted in Figures 1(b) - (e)) - disjoint (ds), overlap (ov), contains (cs), and contained (cd).

2.1 Euler Histograms

To build an Euler histogram H for a set S of objects, the axis-aligned MBR containing the whole S is first divided into $n_1 \times n_2$ disjoint cells, called a *grid* or a *resolution* of H and S respectively. For instance, Figure 2(a) illustrates 5×4 grid.

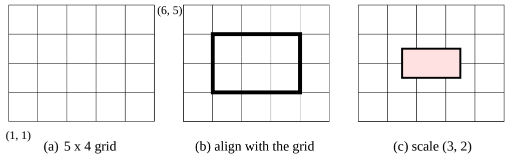


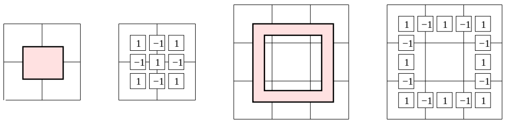
Figure 2: Grid, Query Window, and Object Scale

Note that in a $n_1 \times n_2$ grid, each node on the grid is called *grid point* and labelled by (i, j), lexicographically, with the integers i and j in the range of $1 \le i \le n_1 + 1$ and $1 \le j \le n_2 + 1$. There are n_1 cells spanned the grid horizontally and n_2 cells spanned the grid vertically; thus, n_1 is the *width* and n_2 is the *height* of the grid. The total number of cells, and internal nodes and internal edges is $(2n_1 - 1) \times (2n_2 - 1)$.

In an Euler histogram with a resolution $n_1 \times n_2$, $(2n_1 - 1) \times (2n_2 - 1)$ buckets are given where each cell, internal edge, and internal node are allocated a bucket to store an integer, respectively, such that [3, 20]:

- The integer, corresponding to a cell in the grid, is increased by 1 if an object intersects the cell.
- The integer, corresponding to a node (grid point) in the grid, is increased by 1 if an object contains the node.
- The integer, corresponding to an edge in the grid, is decreased by 1 if an object crosses the edge.

Figure 3(a) gives an example of an Euler histogram. Note that in an Euler histogram H, we do not deal with the information that a boundary of an object *aligns with* the grid of H (see Figure 2(b) for example); this is because we can always "shrink" the object a little bit to remove the situation.



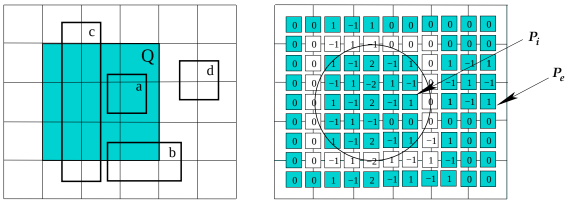
(a) Object without Hole (b) Object with Hole

Figure 3: Euler Histograms

Suppose that H is an Euler histogram for a set S of objects, and Q is an aligned query window.

- $|S|$ denotes the total number of objects in S.
- N_{ds} denotes the number of objects in S which disjoint with Q.
- N_{nds} denotes the number of objects which non-disjoint with Q.
- P_i denotes the summation of all the bucket values inside Q (excluding the boundary of Q, shown in Figure 4).

Assume that S has only one object without holes. The Euler formula [11] implies that the summation of values from the buckets non-disjointing with the object is 1 (see Figure 3(a) for example). Therefore, $N_{nds} = P_i$; this together with $N_{nds} + N_{ds} = |S|$ yield the exact solutions for N_{nds} and N_{ds}.



(a) (b)

Figure 4: Compute P_i and P_e

2.2 S-Euler Algorithm

In [20], the non-disjoint relation is decomposed into 3 relations: overlap, contains, and contained, as depicted in Figure 1 since *equal* relation does not occur in an Euler histogram. Sun, Agrawal, and El Abbadi then proposed to use the histogram information outside a query window Q as well to summarize these 3 new relations. They extended the original Euler formula for processing an object with a *hole* inside. It can be shown that if we sum up the values of the buckets that an object intersects, then the result will be 0 if the object has a hole inside (shown in Figure 3(b)). Here, a *hole* means that there is at least one cell on the grid which is not contained by the object but encompassed by the "outside" boundary of the object; see Figure 3(b) for example. Note that any object studied in this paper is assumed without hole; however by investigating holes we can effectively use the histogram information outside Q.

It has been shown that in an Euler histogram, the ov relation as illustrated in Figure 1(c) has to be separated into 1) *intersect* (the left figure in Figure 1(c)), and 2) *cross-over* (the right figure in Figure 1(c)), since they contribute differently to the outside of Q. Note that here, we abuse the original term "intersect" from [5, 20] for a term simplification; the "intersect" relation in [5, 20] corresponds to the "non-disjoint" relation in this paper. Although in this paper we aim to count the number of ov objects, we will have to first deal with the relations cross-over (cr) and intersect (it), and then add them together to obtain the number of ov objects.

- N_{cs} denotes the number of objects in S which Q contains;
- N_{it} denotes the number of objects in S which intersect Q.
- N_{cr} denotes the number of objects in S which cross over Q.
- N_{cd} denotes the number of objects in S by which Q is contained.
- P_e denotes the summation of all the bucket values outside Q (shown in Figure 4).

By a generalized Euler formula [20], we have

$$P_e = N_{it} + 2N_{cr} + N_{ds} \tag{1}$$

Note that here, we have to count a cr object twice in (1). Clearly, $N_{nds} = N_{it} + N_{cr} + N_{cd} + N_{cs}$; this together with the equation (1) and the equations in the last subsection lead to:

$$N_{ds} = |S| - P_i \tag{2}$$

$$N_{cr} = \frac{1}{2}(P_i + P_e - |S| - N_{it}) \tag{3}$$

$$N_{cs} + N_{cd} = \frac{1}{2}(P_i - P_e + |S| - N_{it}) \tag{4}$$

Clearly, N_{ds} can be computed exactly since P_i can be computed from the histogram and $|S|$ is known. In the equations (3) and (4), there are 4 variables to be fixed. In fact, it tends to be impossible to create more equations without introducing new variables. This is because that the information in one Euler histogram is not enough to determine the 3 relations, cs, cd, and ov. For instance, in Figure 5 the two different scenarios (Figure 5(a) and Figure 5(b)) lead to the same histogram (Figure 5(c)). If we use the shadow area as a query window, we have no idea about what the scenario should be.

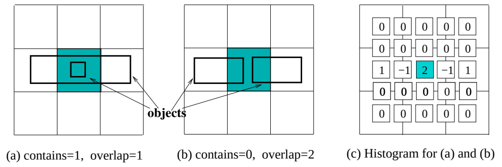


Figure 5: A Counter Example

Motivated by this example, in S-Euler N_{cr} and N_{cd} are both removed from the equations (3) and (4) for approximation. Therefore, the two equations are just enough for the remaining two variables.

2.3 EulerApprox Algorithm

In this algorithm, N_{cr} is still assigned to 0 while N_{cd}, N_{it}, and N_{cs} remain in the two equations (3) and (4). Therefore, one more equation is needed. This is done by adding the following equation.

$$N_{it} + N_{cd} + N_{ds} = N_{cs}(B) + P_e(A) \tag{5}$$

As depicted in Figure 6, the whole space is split into two parts along one edge of Q. Here, $N_{cs}(B)$ is the number of objects contained in the shadow area B, which can be calculated exactly by the algorithm S-Euler. $P_e(A)$ is the summation of all bucket values in the interior of the shadow area A. It has been shown that (5) holds if $N_{cr} = 0$ and the number of O_1 type objects equals the number of O_2 type objects.

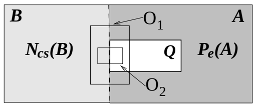


Figure 6: EulerApprox

2.4 M-Euler Algorithm

This algorithm attempts to reduce the deficiency caused by those strong assumptions in S-Euler and EulerApprox by adopting multiple Euler histograms. In M-Euler, the object areas are divided into k ranges and construct one Euler histogram for the objects in each range.

Querying an Euler histogram H against an aligned query window Q in M-Euler proceeds as follows. If the area of each object involved in H is smaller (greater) than the area of Q then S-Euler algorithm is used (in the later $N_{cs} = 0$ instead of $N_{cd} = 0$). Otherwise, EulerApprox is applied.

2.5 Costs of Euler Histograms

An Euler histogram H with a resolution $n_1 \times n_2$ has $(2n_1 - 1) \times (2n_2 - 1)$ buckets, and each bucket stores an integer. Therefore, the storage space required by H is $O(n_1 \times n_2)$.

Since $|S|$ is given, S-Euler and EulerApprox run in constant time by solving the linear equations if P_i and P_e are already obtained; consequently, M-Euler takes $O(k)$ time where k is the number of histograms. In fact, P_i and P_e can be computed in constant time as follows if the *prefix-sum* techniques in [12] is applied to representing Euler histograms.

In H, $H(x, y)$ represents the value in the bucket (x, y) where (x, y) is a representation of a grid point, an edge, or a cell. For a $n_1 \times n_2$ grid, the grid points are $\{(i, j) : 1 \le i \le n_1 + 1, 1 \le j \le n_2 + 1\}$, an edge from the grid point (a, b) to the grid point (a', b') is represented by $(\frac{a+a'}{2}, \frac{b+b'}{2})$, and a cell, with four grid points (a, b), $(a + 1, b)$, $(a, b + 1)$, $(a + 1, b + 1)$, is represented by $(a + 0.5, b + 0.5)$. Note that for presentation simplification, we use non-grid points to represent a cell and an edge.

In the prefix-sum techniques, we use a cumulative representation $H^c(x, y)$ for each (x, y), that is,

$$H^c(x, y) = \sum_{x' \le x, y' \le y} H(x', y').$$

Note that we assume $H(x, y)$ equals zero if there is no entry in the histogram for (x, y). For a query window Q with the bottom-left corner (x_1, y_1) and the upper-right corner (x_2, y_2), the corresponding P_i and P_e are:

$$\begin{aligned} P_i &= H^c(x_2 - \tfrac{1}{2}, y_2 - \tfrac{1}{2}) + H^c(x_1, y_1) - \qquad (6) \\ &\quad H^c(x_1, y_2 - \tfrac{1}{2}) - H^c(x_2 - \tfrac{1}{2}, y_1), \\ P_e &= |S| - H^c(x_2, y_2) + H^c(x_1 - \tfrac{1}{2}, y_2) + \qquad (7) \\ &\quad H^c(x_2, y_1 - \tfrac{1}{2}) - H^c(x_1 - \tfrac{1}{2}, y_1 - \tfrac{1}{2}). \end{aligned}$$

3 Multiscale Histograms

In this section, we will present a multiscale paradigm to construct Euler histograms which can guarantee the exact solutions for N_{cs}, N_{cd}, N_{cr}, N_{it}, and N_{ds} for aligned windows. We first identify our motivation.

3.1 Motivation

The strong assumptions that $N_{cr} = 0$ or/and $N_{cd} = 0$ in S-Euler and EulerApprox greatly downgrade the performance of the two algorithms if the underlying data do not follow the assumptions.

M-Euler aims to remove the disadvantages of S-Euler and EulerApprox by grouping objects together according to their areas. As rectangles with different shapes may have the same area, the disadvantages of S-Euler and EulerApprox cannot be removed effectively by M-Euler.

The full paper version of [20] proved that in the *worst case*, any cell density based histogram requires $\Omega(n_1^2 \times n_2^2)$ storage space to count N_{cs} exactly for aligned windows with respect to a $n_1 \times n_2$ resolution. Therefore, it is impossible for a practical (i.e., storage space linearly proportional to the number of cells) cell density based spatial histogram to provide exact solutions to N_{cs} since $n_1^2 \times n_2^2$ is quadratic with respect to $n_1 \times n_2$. Motivated by these, in this section we will present a multiscale Euler histogram technique with the guarantee of exact solutions to the aligned windows, which may be practical for many real applications though not always. In the next section, we will present another multiscale Euler histogram with high accuracy of approximation (though no guarantee of exact solutions), which is always practical. Our new techniques do not take the assumptions that $N_{cr} = 0$ or/and $N_{cd} = 0$.

3.2 Construction Techniques

Note that in the rest of paper, we will focus only on aligned query windows with respect to a given grid (resolution);

thus, the expression is abbreviated to "a query window" in the rest of the paper whenever no ambiguities.

The basic idea of our multiscale paradigm is to group the objects together according to their *scales*. An object (rectangle) has the *scale* (w, h) with respect to a grid (resolution) if its horizontal edge crosses w cells and its vertical edge crosses h cells (see Figure 2(c) for example). A query window has the *scale* (w, h) if its query rectangle has the scale (w, h).

The following theorem characterises a relationship between the scales and the topological relations for a given query window. The theorem can be immediately verified.

Theorem 1 *Suppose that Q is a query window with the scale (i, j), and D is an object with the scale (w, h) (both scales are referred to the same grid).*

- *If Q contains D, then $w \leq i$ and $h \leq j$.*
- *If D crosses over Q, then ($w \leq i$ and $h \geq j + 2$) or ($w \geq i + 2$ and $h \leq j$).*
- *If Q is contained by D, then $w \geq i + 2$, and $h \geq j + 2$.*

The theorem below is the key to the correctness of our algorithm. It states that a histogram built on the objects with 4 "adjacent" scales can guarantee the exact solutions.

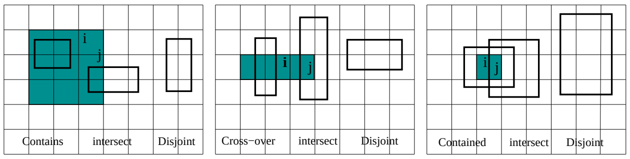
Theorem 2 *Suppose that H is an Euler histogram with a $n_1 \times n_2$ resolution, such that the objects involved in H have at most 4 scales, (w, h), $(w + 1, h)$, $(w, h + 1)$, and $(w + 1, h + 1)$. Then, H can provide the exact solutions to N_{ds}, N_{it}, N_{cs}, N_{cd} and N_{cr} for a query window Q.*

Proof: Suppose that the scale of Q is (i, j) ($1 \leq i \leq n_1$ and $1 \leq j \leq n_2$). Clearly, N_{ds} can be obtained exactly from equation (2). Below are the three cases by comparing (w, h) with (i, j) while querying H against Q:

Case 1: $w \leq i$ and $h \leq j$ (depicted in Figure 7(a)).

Case 2: ($w > i$ and $h \leq j$) or ($w \leq i$ and $h > j$) (depicted in Figure 7(b)).

Case 3: $w > i$ and $h > j$ (depicted in Figure 7(c)).



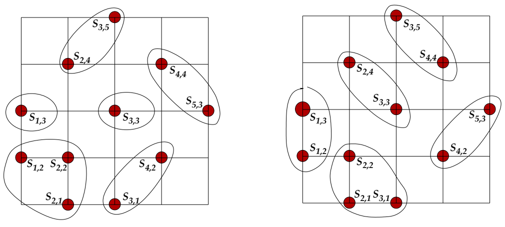
(a) Case 1 (b) Case 2 (c) Case 3

Figure 7: Three Cases by Comparing Q with (w, h)

According to Theorem 1, in case 1 no object in this histogram can cross over Q, nor Q is contained by an object. That is, $N_{cr} \equiv 0$ and $N_{cd} \equiv 0$. Clearly, the remaining two variables N_{cs} and N_{it} can be fixed from the equations (3) and (4).

In case 2, clearly there is no cd relation nor cs relation; that is, $N_{cd} \equiv 0$ and $N_{cs} \equiv 0$. Again, the two remaining variables can be fixed by the two equations.

In case 3, based on Theorem 1 there is no cr relation nor cs relation; that is, $N_{cr} \equiv 0$ and $N_{cs} \equiv 0$. Thus, the two remaining variables can also be fixed by the two equations. □



(a) (b)

Figure 8: Different Sets of Subsets

Let S denote a set of objects. In the rest of the paper, $Q_{i,j}$ denotes a query window with the scale (i, j) at a $n_1 \times n_2$ resolution; $S_{w,h}$ denotes the set of objects, in S, with the scale (w, h) with respect to the $n_1 \times n_2$ resolution. Let $\widehat{S} = \{S_{w,h} : 1 \leq w \leq n_1, 1 \leq h \leq n_2, |S_{w,h}| \neq 0\}$.

To describe our techniques, each dataset $S_{w,h} \in \widehat{S}$ is mapped into a grid point (w, h) on the $n_1 \times n_2$ grid, and $B_{\widehat{S}}$ denotes the set of subsets of $\widehat{S}$, such that each element (a subset of $\widehat{S}$) in $B_{\widehat{S}}$ consists of the datasets whose corresponding grid points belong to only one cell. For example, regarding Figure 8, $B_{\widehat{S}}$ is

$$\{\{S_{1,2}, S_{2,1}, S_{2,2}\}, \{S_{1,2}, S_{2,1}\}, \{S_{1,2}, S_{2,2}\}, \{S_{2,1}, S_{2,2}\}, \{S_{1,2}\}, \{S_{2,1}\}, \{S_{2,2}\}, \{S_{2,1}, S_{2,2}, S_{3,1}\}, ..., \}.$$

Clearly, $|B_{\widehat{S}}| = O(|\widehat{S}|)$. Note that $B_{\widehat{S}}$ is closed under intersection if we add $\emptyset$ to $B_{\widehat{S}}$. For a given S and a given resolution $n_1 \times n_2$ of S, $\widehat{S}$ is unique, and $B_{\widehat{S}}$ is also unique.

An object D is *involved* in an element ξ of $B_{\widehat{S}}$ if D is in a dataset in ξ; for instance, an object is involved in $\{S_{2,1}, S_{2,2}, S_{1,2}\}$ if the object is in one of $S_{2,1}$, $S_{2,2}$, and $S_{1,2}$. A subset of $B_{\widehat{S}}$ is *disjointed* if every pair of elements in the subset do not share a common dataset; for instance, $\{\{S_{1,2}, S_{2,2}\}, \{S_{2,1}, S_{3,1}\}\}$ is disjointed.

Theorem 2 states that a set of objects, which are involved in one element in $B_{\widehat{S}}$, can be represented by the Euler histogram to support the exact solutions. Our algorithm is to find a disjointed subset Λ of $B_{\widehat{S}}$ such that $\widehat{S}$ is covered by Λ; it consists of the following 3 steps.

Multi-Scale Exact Algorithm (MESA)

Step 1: Scan S to form $\widehat{S}$ and $B_{\widehat{S}}$ according to a given resolution (the $n_1 \times n_2$ grid).

Step 2: Find a disjointed subset Λ of $B_{\widehat{S}}$ such that $\widehat{S}$ is covered by Λ.

Step 3: For each element $\xi \in \Lambda$, construct the Euler histogram H_ξ, with the resolution, to represent the objects involved in ξ.

Figure 8(a) illustrates such a Λ with 6 elements, where each element in Λ is "circled". Querying the set of histograms constructed by MESA for a query window may be easily done by querying each histogram with respect to the three cases as described in the proof of Theorem 2, respectively; then adding up the values over all the histograms gives the global N_{cr}, N_{it}, N_{cs}, N_{cd}, and N_{ds}. According to Theorem 2, the algorithm MESA is correct; that is, the histograms constructed can provide the exact solutions to these five values for (aligned) window queries. Since querying each histogram takes constant time (see section 2), querying $|\Lambda|$ Euler histograms constructed by MESA takes $O(|\Lambda|)$ time.

Note that the Step 1 is immediate. An implementation of Step 3 has been briefly described in section 2; the details may be found in [3].

Suppose that Λ is chosen in the Step 2. The number ($|\Lambda|$) of histograms produced by MESA is called the *thickness* of the set of histograms. Clearly, there may be many disjointed subsets of $B_{\widehat{S}}$ to be chosen as an output of the Step 2. Figure 8 shows two different disjointed subsets of $B_{\widehat{S}}$; both cover $\widehat{S}$. One gives the thickness 6 and another gives the thickness 5. In this example, 5 is the minimum thickness.

Note that a multiscale histogram constructed by MESA requires $O(|\Lambda| n_1 \times n_2)$ space for the resolution $n_1 \times n_2$. The minimization of such a $|\Lambda|$ means the minimization of the histogram storage space and query processing costs.

3.3 Minimization of Thickness

According to the construction of $B_{\widehat{S}}$, it is immediate that for any such Λ produced by MESA, $\frac{k}{4} \leq |\Lambda| \leq k$ where $k = |\widehat{S}|$. The minimization problem is formally defined below.

<u>**Optimal Data Partitioning Problem**</u> (ODP)

Instance: Suppose that $\widehat{S}$ and $B_{\widehat{S}}$ are given as above.

Question: find a disjointed subset Λ of $B_{\widehat{S}}$, such that $\widehat{S}$ is covered by Λ and $|\Lambda|$ is minimized.

Recall that each element in $B_{\widehat{S}}$ corresponds to the grid points on one cell. ODP is a special case of the 4-set cover problem [4]; the 4-set cover problem is well-known NP-hard in general. Although a very special case of the 4-set cover problem, unfortunately ODP is still NP-hard.

Theorem 3 *ODP is NP-hard.*

Proof: The proof is quite lengthy and less relevant to the paper. The details are omitted here due to the space limitation; the interested readers may refer to a full version [15] of this paper for details.

The basic idea is to transfer the vertex cover problem for cubic planar graphs into a special case of ODP. To do this, we need to embed a cubic planar graph on a grid, and then construct a corresponding instance for ODP. The technique used is similar to that in [13]. □

Below we present an approximate algorithm to solve ODP.

<u>**Minimizing the Thickness**</u> (MT)

Step 1: Choose a disjointed subset of $B_{\widehat{S}}$ iteratively element by element such that the cardinality of each selected element is at least 3 and the maximum among the available elements.

Step 2: Remove from $B_{\widehat{S}}$ the elements intersecting an element chosen in Step 1. Run the graph maximum matching algorithm to obtain a subset of the remaining $B_{\widehat{S}}$.

Step 3: Output the union of the selections in Step 1 and Step 2 plus the uncovered singletons in $B_{\widehat{S}}$.

Note that in Step 2, after removing the elements from $B_{\widehat{S}}$ with an intersection to an element selected in Step 1, each remaining element of $B_{\widehat{S}}$ have the cardinalities at most 2. The remaining $B_{\widehat{S}}$ can be viewed as a graph G, where a vertex corresponds to a remaining singleton, and each edge corresponds to a remaining element with the cardinality 2. Thus, we can run the maximum matching algorithm to get a maximum matching. Each edge in the maximum matching corresponds to an element in the remaining $B_{\widehat{S}}$, which will be chosen in Step 2. It is immediate that Step 2 takes the dominant costs and runs in $O(|\widehat{S}|)$.

With respect to the example in Figure 8, Step 1 can select only one element. Suppose that we choose $\{S_{1,2}, S_{2,1}, S_{2,2}\}$ in Step 1, then after removing the relevant elements from $B_{\widehat{S}}$, we obtained a graph G with 8 vertices $\{S_{1,3}, S_{2,4}, S_{3,5}, S_{4,4}, S_{5,3}, S_{3,3}, S_{4,2}, S_{3,1}\}$, and 9 edges $\{(S_{1,3}, S_{2,4}), (S_{2,4}, S_{3,5}), (S_{2,4}, S_{3,3}), (S_{3,5}, S_{4,4}), (S_{4,4}, S_{3,3}), (S_{4,4}, S_{5,3}), (S_{5,3}, S_{4,2}), (S_{3,3}, S_{4,2}), (S_{4,2}, S_{3,1})\}$. For this graph, a maximum matching can have only 3 edges. Suppose that $\{(S_{2,4}, S_{3,5}), (S_{4,4}, S_{5,3}), (S_{4,2}, S_{3,1})\}$ is output as a maximum matching in Step2. Then, $S_{1,3}$ and $S_{3,3}$ are chosen in Step 3. In this case, Λ is what is depicted in Figure 8(a).

The *semi-local* optimization technique in [4] may be used to refine the result produced by the algorithm MT; this can guarantee [4] the approximation ration $\frac{19}{12}$. Below we briefly describe the semi-local optimization technique, the interested readers may refer to [4] for the details.

Semi-local Optimization: Iteratively do the following on Λ till there is no improvement.

> Choose an element ξ from Λ with cardinality 4 or 3. Remove, from Λ, ξ and the elements with cardinality less than 3. Remove, from $B_{\widehat{S}}$, the elements with an intersection to the remaining Λ. Run the algorithm MT on the remaining $B_{\widehat{S}}$. If the resultant new Λ' has either smaller number of thickness or the same number of thickness but with less number of singletons, then we continue the next iteration.

As depicted in Figure 8(a), suppose that the circled elements are the output of the algorithm MT. Running the semi-local optimization algorithm, we need to choose a replacement to $\{S_{1,2}, S_{2,1}, S_{2,2}\}$. In this example, either $\{S_{2,1}, S_{2,2}, S_{3,1}\}$ or $\{S_{1,3}, S_{1,2}, S_{2,2}\}$ is an option; both of them can guarantee the minimum thickness 5 by the semi-local optimization algorithm (see Figure 8(b) for example).

In the real datasets used in our experiments, we found that even the algorithm MT can generate the optimal thickness. For instance, for the Texas road segments data of US Census Tiger [22] with the 360×180 resolution, the minimum thickness 13 can be computed by our algorithm MT, while there are 35 different scales. The performance of MESA will be evaluated in section 6.

4 Multiscale Histograms with a Fixed Space

The exact algorithm proposed in the last section suits for datasets with small number ($|\widehat{S}|$) of scales for a given resolution. When $|\widehat{S}|$ is large or the storage space is limited, MESA is not always applicable. Further, we observed that in most real world datasets the majority of objects have similar scales at a reasonable resolution while the total number of outliers (objects) may be very small; thus, it is not economic to use more than one histogram to approximate a small set of objects. To resolve these, in this section we will present an effective algorithm to construct a set of histograms, such that the number of histograms to be used is $k+1$ for a fixed k.

The main idea of our algorithm is to construct k histograms which can provide the exact solutions for the objects involved, while the remaining objects are all put into the last histogram which cannot guarantee the exact solutions. Intuitively, less objects are involved in the last histogram, higher accuracy of approximation may be globally expected on average. Therefore, in our algorithm we aim to allocate the objects to the first k histograms as many as possible while retaining the property of providing exact solutions. Below is a description of our algorithm. For a set Λ of subsets of $\widehat{S}$, $||\Lambda||$ denotes the number of objects involved in Λ.

Multi-scale Approximate Algorithm (MAPA)

Step 1: Scan S to form $\widehat{S}$ and $B_{\widehat{S}}$ according to a given resolution (the $n_1 \times n_2$ grid).

Step 2: Find a disjointed subset Λ of $B_{\widehat{S}}$ such that $|\Lambda| = k$ and $||\Lambda||$ is maximized.

Step 3: For each element $\xi \in \Lambda$, construct the Euler histogram H_ξ, with the $n_1 \times n_2$ resolution, to represent the objects involved in ξ.

Step 4: Construct the Euler histogram H_{last} for the objects not involved in Λ.

In MAPA, the Steps 1, 3, 4 are the same as those in the algorithm MESA. In the subsection 4.1, we will present our results for Step 2. As with the algorithm MESA, the first k histograms generated by the algorithm MAPA can guarantee the exact solutions for N_{cr}, N_{it}, N_{cs}, N_{cd}, and N_{ds} restricted to the objects involved in Λ. In subsection 4.2, we will present a new algorithm to summarize the objects involved in the last histogram H_{last}.

4.1 Data Partition

The optimization problem in Step 2 may be formally described below.

Weighted k-Partitioning Problem (WkP)

Instance: Suppose that $\widehat{S}$ and $B_{\widehat{S}}$ are given as described in section 3, and an integer k is given.
Question: find a disjointed subset Λ of $B_{\widehat{S}}$ such that $|\Lambda| = k$ and $||\Lambda||$ is maximized.

Theorem 4 *WkP is NP-hard.*
Proof: A special case of WkP, where each dataset in $\widehat{S}$ has the same number of objects, is more general than the corresponding decision problem of ODP. □

Below we present a greedy heuristic to approach WkP.

GreedyWkP$(B_{\widehat{S}}, k, \Lambda)$
Sort the elements in $B_{\widehat{S}}$ decreasingly based on the number of objects involved in each element;
$\Lambda \leftarrow \emptyset$;
while $|\Lambda| \neq k$ and $|B_{\widehat{S}}| \neq 0$ **do**
{ get the 1st element ξ from $B_{\widehat{S}}$;
remove the element from $B_{\widehat{S}}$ intersecting ξ;
$\Lambda \leftarrow \Lambda \cup \{\xi\}$; }

According to the definition of $B_{\widehat{S}}$, there are a constant number of subsets (of $\widehat{S}$) in $B_{\widehat{S}}$ intersecting another subset in $B_{\widehat{S}}$. Thus, the dominant cost of GreedyWkP is in sorting $B_{\widehat{S}}$. Recalling that $|B_{\widehat{S}}| = O(|\widehat{S}|)$, GreedyWkP runs in time $O(n \log n)$ where $n = |\widehat{S}|$.

4.2 Summarizing the Last Histogram

In this subsection, we study the problem of summarizing the object set involved in the last histogram H_{last}. To achieve high accuracy, we propose to use scales information in combining with the Euler histogram.

Given an object scale (w, h) and a query window $Q_{i,j}$ with respect to the $n_1 \times n_2$ resolution, the 3 cases in Theorem 1 can be further divided into the following 5 cases:

Case 1. $w \leq i$ and $h \leq j$ - at most 3 relations: cs, it, and ds.
Case 2. $w = i+1$ or $h = j+1$ - at most 2 relations: it and ds.
Case 3a. $w \geq i+2$ and $h \leq j$ - at most 3 relations: cr, it, and ds.
Case 3b. $w \leq i$ and $h \geq j+2$ - at most 3 relations: cr, it, and ds.
Case 4. $w \geq i+2$ and $h \geq j+2$ - at most 3 relations: cd, it, and ds.

We should be able to estimate the occurring probabilities against the 5 relations (cr, it, cs, cd, and ds), respectively, in the object scale (w, h) with respect to $Q_{i,j}$. For a given case and a given topological relation, we will calculate the ratio of the number of possible grid points to be used as the *bottom-left* corner of an object with the scale (w, h) to form the relation over the number of possible grid points to be used as the bottom-left corner of an object with the scale (w, h).

As depicted by the rectangular areas in Figure 9, we use δ_{cr}, δ_{it}, δ_{cs}, δ_{cd}, and δ_{ds} to denote the number of grid points, possibly used as bottom-left object corners for the 5 relations, respectively. Note that in Figure 9, we illustrate only three cases: Case 1, Case 3a, and Case 4. Case 3b is similar to Case 3a; and Case 2 is similar to all these three cases but without the white area in the middle.

Below we present the detailed formulae to identify those rectangles in Figure 9 with respect to each case. The formulae may be immediately obtained by elementary geometry; thus, we omit the deduction details from this paper. Note that the size and position of each rectangle area in Figure 9 not only depend on the scales (w, h) and (i, j) but also depend on a position of $Q_{i,j}$.

Suppose that a query rectangle $Q_{i,j}$ with the bottom-left corner (q_x, q_y), and a rectangle is represented by $\{(x, y), (a, b)\}$ where (x, y) is the bottom-left corner and (a, b) represents (width, height).

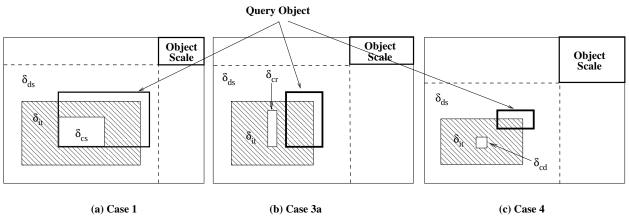


Figure 9: Different cases

Case 1:

In this case, $\delta_{cr} \equiv 0$ and $\delta_{cd} \equiv 0$, while the others can be calculated as follows.

δ_{cs} is the number of grid points in the rectangle $r_{cs} = \{(x_{cs}, y_{cs}), (a_{cs}, b_{cs})\}$ (the white area in Figure 9(a)). Here,

- $x_{cs} = q_x$ and $y_{cs} = q_y$;
- $a_{cs} = i - w$, and $b_{cs} = j - h$.

Note that "in the rectangle" also include the points on the edge; this is also applicable to the cases below. Note $a_{cs} = 0$ and $b_{cs} = 0$ imply there is only one point (x_{cs}, y_{cs}).

δ_{it} is the number of grid points in the rectangle (the one with slash lines in Figure 9(a)) $r_{it} = \{(x_{it}, y_{it}), (a_{it}, b_{it})\}$ but exclude the points in r_{cs}. Note that in this case, r_{it} is getting smaller when an edge of $Q_{i,j}$ approaches a boundary of the grid. Below is a precise formula.

- $x_{it} = 1+\max\{0, q_x - w\}$ and $y_{it} = 1+\max\{0, q_y - h\}$;
- $a_{it} = q_x + i - 1 - x_{it} - \max\{0, q_x + i + w - n_1 - 2\}$ and $b_{it} = q_y + j - 1 - y_{it} - \max\{0, q_y + j + h - n_2 - 2\}$.

Note that an object with the scale $(1, 1)$ never intersects $Q_{i,j}$; consequently $\delta_{it} \equiv 0$ in this case. This can be reflected by the formula. In fact we can verify that r_{it} is always the same as r_{cs} when $w = 1$ and $h = 1$; and thus δ_{it} is calculated as 0.

δ_{ds} is the number of grid points in the rectangle $r_{ds} = \{(x_{ds}, y_{ds}), (a_{ds}, b_{ds})\}$ (the one bounded by dashed line in Figure 9) except the grid points in r_{it}. Here,

- $x_{ds} = 1$ and $y_{ds} = 1$;
- $a_{ds} = n_1 - w$ and $b_{ds} = n_2 - h$.

Note that this rectangle is determined only by the object scale (w, h). In case 1, the probabilities for cs, it, and ds, respectively, are

$$\rho^1_{cs} = \frac{\delta_{cs}}{\delta_{cs} + \delta_{it} + \delta_{ds}},\ \rho^1_{it} = \frac{\delta_{it}}{\delta_{cs} + \delta_{it} + \delta_{ds}},$$
$$\rho^1_{ds} = \frac{\delta_{ds}}{\delta_{cs} + \delta_{it} + \delta_{ds}}.$$

It can be immediately verified that the rectangle r_{it} always includes the rectangle r_{cs}, and is always included by r_{ds}. Therefore, the computation of δ_{cs}, δ_{it}, and δ_{ds} is simple. We first calculate δ_{cs} from r_{cs}. We then calculate δ_{it} as the total points in r_{it} minus δ_{cs}, and calculate δ_{ds} as the total points in r_{ds} minus δ_{cs} and minus δ_{it}. Therefore, ρ^1_{cs}, ρ^1_{it}, and ρ^1_{ds} can be calculated in constant time.

Case 2:

In this case, $\delta_{cs} \equiv 0$, $\delta_{cr} \equiv 0$, and $\delta_{cd} \equiv 0$, while δ_{it} and δ_{ds} can be calculated by the above formula. By similar reasons, $\rho^2_{it} = \frac{\delta_{it}}{\delta_{it}+\delta_{ds}}$ and $\rho^2_{ds} = \frac{\delta_{ds}}{\delta_{it}+\delta_{ds}}$ can be computed in constant time. Note that we can more precisely handle this case by dividing it into two sub-cases: a) $a = i+1$ and $b \neq j+1$, and b) $b = j+1$. However, our experiments show that we gain very marginally by doing this. Therefore, we omit these from the paper.

Case 3a:

In this case, $\delta_{cs} \equiv 0$ and $\delta_{cd} \equiv 0$. We can calculate δ_{cr} as follows.

- $\delta_{cr} = 0$ if $q_x = 1$ or $q_x + i = n_1 + 1$, otherwise
- δ_{cr} is the number of grid points in the rectangle $r_{cr} = \{(x_{cr}, y_{cr}), (a_{cr}, b_{cr})\}$ (white rectangle in Figure 9(b)). Here,
  - $x_{cr} = 1 + \max\{0, q_x + i - w\}$ and $y_{cr} = q_y$;
  - $a_{cr} = q_x - 1 - x_{cr} - \max\{0, q_x + w - n_1 - 2\}$ and $b_{cr} = j - h$.

Note that the rectangles for δ_{it} and δ_{ds} can be calculated by the same formulae as those in Case 1. Again, it can be immediately verified that the rectangle r_{it} always includes r_{cr} and is included by r_{ds}. Similarly, $\rho^{3a}_{cr} = \frac{\delta_{cr}}{\delta_{cr}+\delta_{it}+\delta_{ds}}$, $\rho^{3a}_{it} = \frac{\delta_{it}}{\delta_{cr}+\delta_{it}+\delta_{ds}}$, and $\rho^{3a}_{ds} = \frac{\delta_{ds}}{\delta_{cr}+\delta_{it}+\delta_{ds}}$ can be computed in constant time, respectively.

Case 3b:

Every thing can be viewed as a reflectional image, by the diagonal of the query rectangle, of case 3a; and thus may be calculated in a similar way to those in Case 3a. Therefore, ρ^{3b}_{cr}, ρ^{3b}_{it}, and ρ^{3b}_{ds} can be computed in constant time.

Case 4:

In this case, $\delta_{cs} \equiv 0$ and $\delta_{cr} \equiv 0$. Clearly, $\delta_{cd} = 0$ if $q_x = 1$ or $q_y = 1$ or $q_x + i = n_1 + 1$ or $q_y + j = n_2 + 1$; otherwise the rectangle $r_{cd} = \{(x_{cd}, y_{cd}), (a_{cd}, b_{cd})\}$ for calculating δ_{cd} is as follows.

- $x_{cd} = 1+\max\{0, q_x + i - w\}$ and $y_{cd} = 1+\max\{0, q_y + j - h\}$;
- $a_{cd} = q_x - 1 - x_{cd} - \max\{0, q_x + w - n_1 - 2\}$ and $b_{cd} = q_y - 1 - y_{cd} - \max\{0, q_y + h - n_2 - 2\}$.

Note that the rectangles for δ_{it} and δ_{ds} can be calculated by the same formulae as those in Case 1. It can be immediately verified that the rectangle r_{it} includes r_{cd} but is included by r_{ds}. Similarly, ρ^3_{ds}, ρ^3_{it}, and ρ^3_{cd} may be computed in constant time.

Note that for a given query window $Q_{i,j}$ and a set of m objects with the scale (w, h), we can estimate N_{cr}, N_{it}, N_{cs}, N_{cd} purely by the above probabilities; that is, $N'_{cr} = \rho_{cr}m$, $N'_{it} = \rho_{it}m$, $N'_{cs} = \rho_{cs}m$, $N'_{cd} = \rho_{cd}m$, and $N'_{ds} = \rho_{ds}m$. Since we assume that $m_{w,h} = |S_{w,h}|$ is recorded for each $S_{w,h}$, we can immediately calculate the above estimation. Therefore, by summing up all the above estimates, respectively, we can get the global estimation of N_{cr}, N_{it}, N_{cs}, N_{cd} and N_{ds}.

The pure probability approach above has two limitations: 1) the running time is $O(k)$ (k is the number of object scales) which is not necessary a constant, 2) it does not make the use of the advantages of an Euler histogram.

Now we present the algorithm Prob with constant time, combining the probability approach above with the Euler histogram. The basic idea is to divide the objects involved in H_{last} into the possible 5 cases (groups) as stated above; then we use an average rectangle to approximately represent the objects in each case together with the number of objects. Thus, for each case we can use constant time to compute the conditional possibility. Below is the description of Prob.

Algorithm Prob (for a given Q)
$\alpha := 0$; $\beta := 0$; $\mu := 0$; $\gamma := 0$;
for each Case i ($i \in \{1, 2, 3a, 3b, 4\}$) **do**
{ calculate m_i; //the number of objects in this case.
calculate $\bar{w}_i$ and $\bar{h}_i$; // average width and height.
calculate ρ^i_{cr}, ρ^i_{it}, ρ^i_{cs}, and ρ^i_{cd} against $\bar{w}_i$ and $\bar{h}_i$;
$\alpha := \alpha + m_i\rho^i_{cr}$; $\beta := \beta + m_i\rho^i_{it}$;
$\mu := \mu + m_i\rho^i_{cs}$; $\gamma := \gamma + m_i\rho^i_{cd}$; }
if $\gamma + \mu = 0$ **then**
{ $N_{cs} := 0$; $N_{cd} := 0$;
calculate N_{cr} and N_{it} from (3) and (4); }
else
{ get N_{cr} and N_{it} from (3) by $N_{cr} : N_{it} = \alpha : \beta$;
get N_{cd} and N_{cs} from (4) by $N_{cs} : N_{cd} = \mu : \gamma$; }

Note that in Prob, if each m_i, $\bar{w}_i$ and $\bar{h}_i$ can be calculated in constant time then the whole algorithm will run in constant time. Below we show a pre-fix data structure to accommodate such a request.

Time Complexity of Prob

We apply the prefix-sum technique to representing $\{m_{a,b} : 1 \leq a \leq n_1, 1 \leq b \leq n_2\}$; for $1 \leq a \leq n_1$ and $1 \leq b \leq n_2$, $m'_{a,b} = \sum_{1 \leq w \leq a, 1 \leq h \leq b} m_{w,h}$.

Let $w_{a,b}$ and $h_{a,b}$ denote the total widths and total heights of the objects in the scales $[1, a] \times [1, b]$, respectively. Besides H_{last}, in algorithm Prob we also pre-store

$$\{(m'_{a,b}, w_{a,b}, h_{a,b}) : 1 \leq a \leq n_1, 1 \leq b \leq n_2\}.$$

By similar methods to those in section 2.5, the total width, total height, and total number can be computed, respectively, for each case in constant time; then we can calculate the average widths and heights accordingly. Consequently, the algorithm Prob runs in constant time.

Note that the algorithm Prob takes $(2n_1 - 1)(2n_2 - 1) + 3n_1n_2$ storage space which is about 75% more than the storage space ($(2n_1 - 1)(2n_2 - 1)$) for one Euler histogram.

Since querying every histogram runs in constant time, querying a set of histograms generated in MAPA runs in time $O(k)$ for a window query, where k is the number of histograms.

5 Maintenance, Generalization, and Applications

The histograms generated by MAPA may be maintained as follows for dataset updates. For an insertion, we need only to update the corresponding histogram for the relevant node and cell values, respectively, by increasing 1; and update the relevant edge values, respectively, by decreasing 1. For a deletion, the updates to the corresponding histogram are opposite to those for an insertion. Further, if an insertion or deletion is involved in the last histogram, we need also to update their corresponding statistic values accordingly. Note that as the histogram values and the statistic information are stored in a cumulative fashion, an update needs to be propagated in a cumulative fashion as well. Moreover, we can also keep a threshold for the number of changes to trigger MAPA to generate a new set of histograms.

Note that the techniques presented in this paper are also applicable to the case if the whole object space is unevenly divided into cells. In this case, the histogram construction time will be increased by a logarithmic factor due to binary search of the cells in a grid. The discussion and investigation of the best way to partition the object space is not within the scope of the paper.

It should be mentioned that the Euler histogram techniques are not only applicable to estimating spatial range query results but may also be immediately applicable to spatial digital libraries to support window browsers [3, 9, 20, 21]. Further, our results are also fundamental to the development of new selectivity estimation techniques in spatial joins with the join predicates, such as contains, intersection, cross-over, etc.

6 Performance Evaluation

In this section we evaluate the performance of our new techniques, MESA, MAPA, and Prob. As M-Euler in [20] is the only work dealing with the same problems studied in this paper, it will be used as a benchmark algorithm in the evaluation. Specifically, we evaluate the accuracy of the following techniques:

- M-Euler [20]: When only one histogram is used, it is the EulerApprox [20].
- MAPA-Prob: Prob is used to query the last histogram generated by MAPA. Note that when only one histogram is used, it is the Prob.
- MAPA-Simple: In case that a small number of objects left in the last histogram generated by MAPA, we throw away the remaining objects.

We also evaluated the query time in these techniques as well as the histogram construction costs; these will be done together with MESA. Note that we did not evaluate the performance of MESA against synthetic datasets used in the paper since they are designed to show the disadvantages of MESA - the necessity of developing MAPA; that is, there are a big number of object scales.

Our implementation has been carried out on PC Pentinum4 - 2.2GHz with 512M RAM.

Datasets and Resolutions

In our experiment, real-world and synthetic datasets are used. To do a fair comparison with M-Euler regarding accuracy, we adopt the 360×180 resolution to evaluate the accuracy of our algorithms, as this resolution was used in [20] to provide the experiment results. The 360×180 grid is a simulation of the earth resolution by the longitude and latitude. Below are the datasets used.

- **Ca_road** consists of the 2,851,627 California road segments obtained from the US Census TIGER [22] dataset. We normalized the dataset into the 360×180 grid.
- **Ca_Tx_road** consists of the 3,653,571 Texas road segments (**Tx_road**) and the 2,851,627 California road segments extracted from the US Census TIGER [22] dataset. We combine them together by normalizing both of them into the 360×180 grid. By combining the two real world datasets together, we hope that N_{cd} and N_{cr} may be reasonably significant.
- **SAME** is a synthetic dataset used in [20] such that each object has width 3.6 and height 1.8, while the distribution of the positions follow a Zipf fashion [23]. This dataset is believed a simulation of many real world datasets.

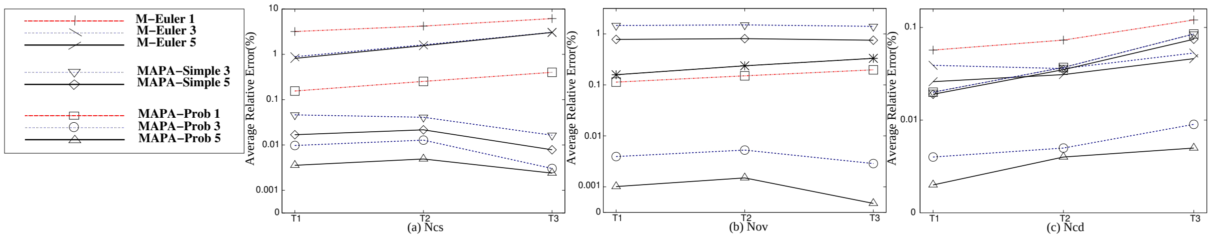


Figure 10: California Road Segments Data

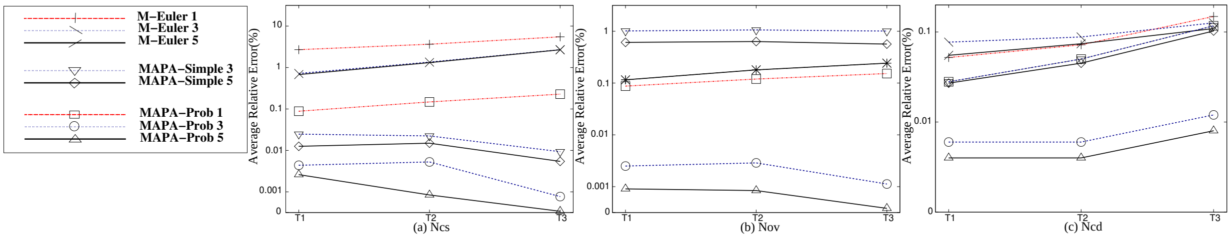


Figure 11: California & Texas Road Segments Data

- **Zipf1** is a synthetic dataset with one million square objects. The centers of the objects are uniformly distributed over the 360×180 grid, while the side lengths follow a Zipf distribution.
- **Zipf2** is to add $250,000$ objects with the scale $(1, 50)$ and $250,000$ objects with the scale $(30, 30)$ on the top of Zipf1 dataset. This dataset will produce large values of N_{cr} and N_{cd}. Though it is quite unusual in real world, it is expected to further confirm the advantages of our algorithms.

Query Sets

To evaluate thoroughly the performance of those algorithms, we select the query windows to accommodate various different user query patterns. We divide query windows into two classes, small and non-small. A query window in small class has a scale such that the width and height are both smaller than 5, while a query window in non-small class has either height between 6 and 20 or width between 6 to 20. We randomly generate 3 different sets of windows, T_1, T_2, and T_3, each of which has $100,000$ query windows.

In T_1, 20% of the $100,000$ query windows are in the small class. In T_2, 40% of the query windows are in the small class, while in T_3, 80% of the query windows are in the small class.

Error Metrics

For each $Q \in T_i$ $(1 \leq i \leq 3)$, we record the relative errors for N_{cs}, N_{cd} and N_{ov}, respectively, where $N_{ov} = N_{it} + N_{cr}$. Recall that the relations it and cr are subdivided from the relation ov, and we aim only to summarize the relation ov. The relative error is defined below.

$$\epsilon = \begin{cases} \frac{|e-e'|}{e} & \text{if } e \neq 0 \\ e' & \text{otherwise} \end{cases} \tag{8}$$

Here, e is the exact value and e' is an approximate value. The average relative error may be defined below.

$$\frac{\sum_{Q \in T_i} \epsilon_Q}{|T_i|} \tag{9}$$

Here, ϵ_Q is the relative error for a query window Q.

Effectiveness of MESA

In the dataset SAME used in [20], the objects have 4 different scales with respect to the 360×180 resolution: $(4, 2)$, $(4, 3)$, $(5, 2)$, and $(5, 3)$. While M-Euler cannot provide the exact solutions even with 4 histograms, MESA can always guarantee the exact solutions by only one Euler histogram. Further, in this application the querying time of MESA is about 4 times less than M-Euler when M-Euler uses 4 histograms. We will also present the experiment results regarding the histogram construction time and querying time later together with the other algorithms.

We examined the number of histograms produced by MESA, respectively, against the 3 resolutions 100×50, 180×90, and 360×180 for Ca_road and Tx_road. The numbers of histograms generated in Ca_road are 3, 8, and 17, respectively, while the breakdown numbers for Tx_road are 4, 9, and 13, respectively.

Note that the number of object scales is usually increased with the increment of resolution; this should be also applicable to the number of histograms generated by MESA. Further, it is quite unusual that a resolution higher than 360×180 cells will be applied to many spatial data processing applications (including selectivity estimation) for a dataset with less than 4 millions rectangles. Therefore, we can argue that in real applications, MESA for these two real datasets requires a storage space $O(l)$ where l is the number of cells. Thus, it is practical. This together with the dataset SAME (only one histogram is required) indicate that MESA may be practical for many real datasets.

Approximation Accuracy

We examine the approximation accuracy of 3 algorithms M-Euler, MAPA-Prob, and MAPA-Simple, against 3 different storage space requirements: 1 histogram, 3 histograms, and 5 histograms. In our experiments, we examine only the accuracies of N_{cs}, N_{cd}, and N_{ov} but N_{ds} is omitted; this is because these 3 algorithms are always able to produce the exact answers to N_{ds} (see equation

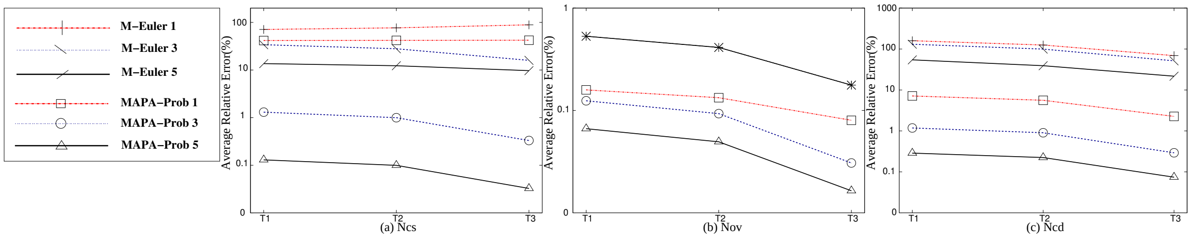


Figure 12: Synthetic Data - Zipf1

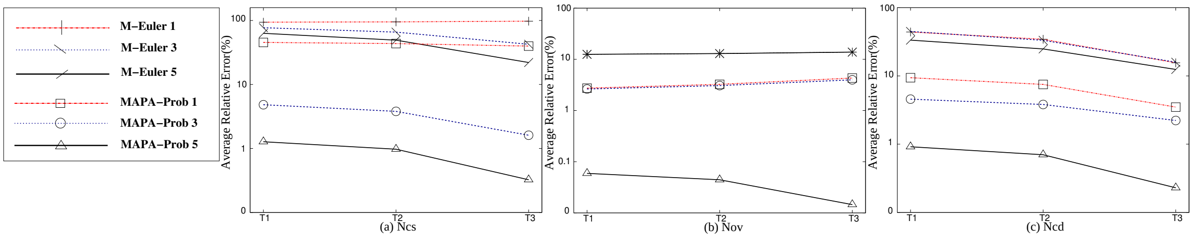


Figure 13: Synthetic Data - Zipf2

(2)). We recorded the average relative errors for a given storage space and a given query set for these 3 algorithms, respectively.

Our algorithm MAPA automatically generates a set of histograms but in M-Euler we need to intuitively specify the data partitioning manually to obtain a good set of histograms.

Figure 10 shows the experiment results against the real-world dataset Ca_road, where M-Euler 1, 3, 5 denote the experiment results of M-Euler while using 1, 3, 5 histograms, respectively. Similar notation is also applied to MAPA-Simple and MAPA-Prob. Note that we did not implement MAPA-Simple 1, as too many objects will be thrown away if doing so. The experiment results demonstrated that MAPA-Prob greatly improve the accuracy of M-Euler, while MAPA-Prob 5 may improve the accuracy of M-Euler by up to two orders of magnitude. It is interesting to note that MAPA-Simple performs clearly better than M-Euler only regarding N_{cs}.

The experiment results for dataset Ca_Tx_road continue the trends, as depicted in Figure 11.

Figure 12 presents the experiment results for the synthetic dataset Zipf1. For this dataset, we did not implement the algorithm MAPA-Simple as there are too many objects left in the last histogram. In our implementation of M-Euler, we use the data partitioning suggested in [20]. To generate 3 histograms, the first histogram contains the objects with the areas 1 to 8, the second histogram contains the objects from areas 9 to 99, and the third histogram contains the objects with the areas 100 and more. To generate 5 histograms, the first histogram contains the object with the areas from 1 to 8, the second with the areas from 9 to 24, the third with the areas from 25 to 99, the fourth with the areas from 100 to 224, and the fifth with the areas 225 and more. The experiment results follow similar trends to those in Ca_road. It is interesting to note that MAPA-Prob 1 already greatly out-performs M-Euler 5 with respect to N_{ov} and N_{cd} though the storage space required by MAPA-Prob 1 is about 4 times smaller than that in M-Euler 5.

Similar trends to those in Zipf1 continue in Zipf2, as depicted in Figure 13.

It is worth to note that the accuracy of estimating N_{ov} in M-Euler is always fixed regardless of the number of histograms to be used; this may be problematic as illustrated by Figure 13 (b). In summary, MAPA-Prob should be the best option among these 3 algorithms regarding the approximation accuracy.

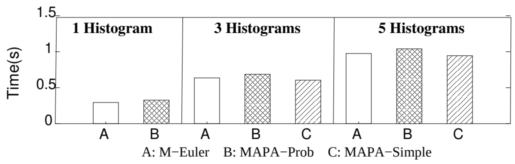


Figure 14: Query Time

Query Time

As analyzed earlier, the time for querying an Euler histogram in M-Euler, MAPA-Simple, and MAPA-Prob, is constant, respectively, which is irrelevant to the size of the Euler histogram and the underlying spatial data. Our experiment results (based on Ca_Tx_road) in Figure 14 confirmed this.

In fact, our implementation of these 3 algorithms against all the data demonstrated that the query time depends only on the number of histograms required and the types of the algorithm used. Note that the algorithm Prob is slightly slower than the algorithm EulerApprox, and the algorithm EulerApprox is slightly slower than the algorithm for solving the linear equations (2) - (4) directly when two variables are zero. These have been reflected in the experiment results.

We implemented MESA against Ca_Tx_road; it takes about 5 seconds for $100,000$ window queries; this is because there are 19 histograms involved.

Histogram Construction Time

We evaluated the running time to construct the histograms. In the datasets used in the experiments, Ca_Tx_road has the largest number of objects, about 6,500,000. The time costs for constructing the histograms in M-Euler, MAPA-Prob, and MAPA-Simple, respectively for 1 histogram, 3 histograms, and 5 histograms with the two resolutions, 360×180 and 180×90 are between 40 seconds and 41 seconds, respectively. In fact, the costs of these 3 algorithms, for constructing the histograms, are dominated by the costs of scanning the dataset; this is why those construction costs are similar.

We also recorded the histogram construction time in MESA against the data Ca_Tx_road. It takes about 49 seconds for the resolution 360×180, and about 42 seconds for the resolution 180×90. The construction time in MESA for 360×180 is significantly higher than those of M-Euler, MAPA-Prob, and MAPA-Simple due to the costs of a search for the right histogram for each object and the costs for computing the bucket values, as there are 19 histograms involved.

7 Conclusion and Remarks

In this paper, we investigated the problem of effectively summarizing the four topological relations against large spatial datasets by histograms. By effectively utilising the object scale information, we first present an efficient algorithm MESA to construct a small set of histograms, based on a multi-scale paradigm, to provide exact summarization results for aligned windows. To conform to a limited storage space, we also provide an effective algorithm MAPA to construct a fixed number of histograms with the aim to minimize the estimation errors. Finally, we presented a novel and effective approximate algorithm, Prob, to query one histogram; it runs in constant time. Our experiment results demonstrated that our techniques, developed in this paper, greatly improve the accuracy of the existing techniques while retaining the costs efficiency.

As a possible future study, we will investigate the problem of dividing object spaces effectively, and explore the other related research directions.

References

[1] Ashraf Aboulnaga and Jeffrey F. Naughton. Accurate estimation of the cost of spatial selections. In *ICDE'00*, pages 123–134, March 2000.

[2] Swarup Acharya, Viswanath Poosala, and Sridhar Ranmaswamy. Selectivity estimation in spatial databases. In *SIGMOD'99*, pages 13–24, June 1999.

[3] R. Beigel and Egemen Tanin. The geometry of browsing. In *Proceedings of the Latin Ameriacn symposium on Theoretical Informatics,1998,Brazil*, pages 331–340, 1998.

[4] R. Duh and M. Furer. Approximation of k-set cover by semi-local optimization. In *STOC'97*, pages 256–264. ACM, 1997.

[5] Max J. Egenhofer and John R. Herring. Categorizing binary topological relations between regions, lines, and points in geographic databases. In Max J. Egenhofer, David M. Mark, and John R. Herring, editors, *The 9-Intersection: Formalism and Its Use for Natural-Language Spatial Predicates. National Center for Geographic Information and Analysis, Report 94-1*, pages 13–17, 1994.

[6] Volker Gaede and Oliver Gunther. Multidimentional access methods. *Computing Surveys*, 30(2):170–231, 1998.

[7] M. Garofalakis, J. Gehrke, and R. Rastogi. Query and mining data streams: You only get one look. In *VLDB'02: Tutorial*, Aug 2002.

[8] A.C. Gilbert, Y. Kotidis, S. Muthukrishnan, and M.j. Strauss. How to summarize the universe: Dynamic maintenance of quantiles. In *VLDB'02*, pages 454–465, Aug 2002.

[9] S. Greene, E. Tanin, C. Plaisant, B. Shneiderman, L. Olsen, G. Major, and S. Johns. The end of zero queries: Query preview for nasa's global change master directory. *International Journal of Digital Libraries*, pages 70–90, 1999.

[10] M. Grigni, D. Papadias, and C. Papadimitriou. Topological inference. In *IJCAI 1995*, pages 901–907, 1995.

[11] F. Harary. *Graph Theory.* Addison-Wesley Publishing Company, 1969.

[12] Ching Tien Ho, Rakesh Agrawal, Nimrod Megiddo, and Ramakrishnan Srikant. Range queries in olap data cubes. In *SIGMOD'97*, pages 73–88, May 1997.

[13] A. Itai, C.H. Papadimitriou, and J.L. Szwarcfiter. Hamilton paths in grid graphs. *SIAM J. Comput.*, 11(4):676–686, Nov 1982.

[14] Ji Jin, Ning An, and Anand Sivasubramaniam. Analyzing range queries on spatial data. In *ICDE'00*, pages 525–534, March 2000.

[15] X. Lin, Q. Liu, Y. Yuan, and X. Zhou. Multiscale histograms: Summarizing topological relations in large spatial datasets. Manuscript, 2003.

[16] R. J. Lipton, J. F. Naughton, and D. A. Schneider. Practical selectivity estimation through adaptive sampling. In *SIGMOD'90*, pages 1–11, May 1990.

[17] Y. Matias, J. S. Vitter, and M. Wang. Wavelet-based histograms for selectivity estimation. In *SIGMOD 1998*, pages 93–102. ACM, 1998.

[18] Viswanath Poosala. *Histogram-based estimation techniques in databases.* PhD thesis, Univ. of Wisconsin-Madison, 1997.

[19] C. Sun, D. Agrawal, and A. El Abbadi. Selectivity estimation for spatial joins with geometric selection. In *Extending Database Technology*, pages 609–626, 2002.

[20] Chengyu Sun, Divyakant Agrawal, and Amr El Abbadi. Exploring spatial datasets with histograms. In *ICDE'02*, pages 93–102, Feb 2002.

[21] A.S. Szalay, P. Kunzst, A. Thakar, J. Gray, D. Slutz, and R.J. Brunner. Designing and mining multi-terabyte astronomy archieves: The sloan digital sky survey. In *SIGMOD 2000*, 2000.

[22] TIGER. Tiger/line files. Technical report, U.S.Census Bureau, Washington, DC, 2000.

[23] G. K. Zipf. *Human Behaviour and the Principles of Least Effort.* Addison-Wesley, Reading, MA, 1949.