Clustering Moving Objects for Spatio-temporal Selectivity Estimation

Qing Zhang **Xuemin Lin**

School of Computer Science and Engineering,
University of New South Wales,
Sydney, Australia 2052,
Email: {qzhang,lxue}@cse.unsw.edu.au

Abstract

Many spatio-temporal applications involve managing and querying moving objects. In such an environment, predictive spatio-temporal queries become an important query class to be processed to capture the nature of moving objects. In this paper, we investigated the problem of selectivity estimation for predictive spatio-temporal queries. We propose a novel histogram technique based on a clustering paradigm. To avoid expensive computation costs, we developed linear time heuristics to construct such a histogram. Our performance study indicated that the new techniques improve the accuracy of the existing techniques by one order of magnitude.
Keywords: Predicative Queries, Spatio-temporal Databases, Histograms.

1 Introduction

Modern database applications such as Global Positioning Systems, Environmental Information Systems and mobile computing deal with moving objects, which are managed by a *spatial-temporal database management system* (STDBM). Recently, research in STDBM has attracted a great deal of attention (Erwig, Gting, Schneider & Vazirgiannis 1999, Forlizzi, Gting, Nardelli & Schneider 2000, Pankaj K. Agarwal 2000, Kollios, Gunopulos & Tsotras 1999, Saltenis, Jensen, Leutenegger & Lopez 2000). In STDBM, there are two kinds of queries based on different temporal natures. One is to query historical behavior of moving objects, and the other is to predict the future behavior of moving objects. A typical example of the predictive window query is to retrieve the number of taxicabs that will be within 1 mile distance from downtown during the next couple of hours. In general, a *predicative spatio-temporal* window query (PST query) is to find the objects intersecting a query window during a time interval. An accurate selectivity estimation not only provides a good approximation to the corresponding query aggregation, but also is required by a query processing optimizer.

Very recently, histogram techniques (Acharya, Poosala & Ramaswamy 1999, Poosala 1997, Muralikrishna & DeWitt 1988, Ioannidis 1993, Ioannidis & Poosala 1999) have been applied to selectivity estimation for PST query. More specifically, the Minskew technique (Acharya et al. 1999) is adopted in the recent research work (Choi & Chung 2002, Hadjieleftheriou, Kollios & Tsotras 2003, Tao, Sun & Papadias 2003) to approach the problem of selectivity estimation for PST query. Choi and Chung (Choi & Chung 2002) presented the first work by using the *Minskew technique* (Acharya et al. 1999) to construct a spatio-temporal histogram. It mainly focused on managing one dimensional moving objects (points or line segments), and then it is extended to a multi-dimensional space by multiplying the estimation results in each dimension. Hadjieleftheriou et al. (Hadjieleftheriou et al. 2003) proposed new techniques of *dual velocity-intercept* space and *primal space-time space* to handle multi-dimensional moving points, where the Minskew technique is also applied. Tao et al. (Tao et al. 2003) presented the first comprehensive study of the PST query in Spatio-Temporal database. Again, the Minskew technique is used in constructing histograms for dealing with multi-dimensional moving "rectangles".

In this paper, we propose a novel histogram construction method that is completely different from the Minskew technique. This new technique is based on a clustering paradigm with the aim to remove some deficiencies of the Minskew technique, which may be aggregated by moving objects. To the best of our knowledge, this is the first paper investigating an application of clustering techniques to construct histograms. This is the first contribution of the paper. The second contribution of the paper is that we developed a linear time heuristic algorithm to construct histograms, and one part of our algorithm can be also used as a refinement to the Minskew technique. Finally, we discuss a histogram maintenance technique regarding updates. Our extensive experiments on both synthetic and real data sets demonstrated that the new clustering based histogram construction technique may improve the accuracy of the existing techniques by 1.5 order of magnitude.

The rest of the paper is organized as follows. Section 2 briefly introduces the related work and presents the motivation of the research. Section 3 presents our clustering based histogram techniques. Section 4 reports our experiment results. This is followed by conclusion.

2 Preliminary

In this section, we first present background knowledge and then briefly review related works. Finally, we present the motivation for this research work.

2.1 Background Knowledge

In spatio-temporal datasets, each data object r of data set D is considered to be a moving rectangle in a n-dimensional space. We can use the following vector $T_r(0)$ to represent an object r's position at current time (time zero) and its velocities.

$$\begin{aligned} T_r(0) = &\{r_{1-}(0), r_{1+}(0), vr_{1-}, vr_{1+}, ..., \\ & r_{n-}(0), r_{n+}(0), vr_{n-}, vr_{n+}\} \end{aligned}$$

where, $[r_{i-}(0), r_{i+}(0)]$ is the *extent* of r on the i-th dimension ($1 \leq i \leq n$); vr_{i-} (vr_{i+}) is the velocity of the lower (upper) boundary of r on the ith dimension ($1 \leq i \leq n$). For each dimension i ($1 \leq i \leq n$), we use

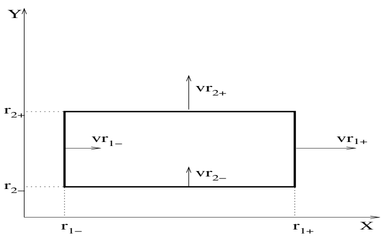


Figure 1: A Moving 2-D Rectangle

$[U_{i,min}, U_{i,max}]$ to denote an application domain (extent), and use $[V_{i,min}, V_{i,max}]$ to denote possible velocity values. Figure 1 illustrates an example in the 2-dimensional space. The extent of r on the i-th dimension at time t can be calculated by:

$$\begin{cases} r_{i-}(t) = r_{i-}(0) + vr_{i-} * t \\ r_{i+}(t) = r_{i+}(0) + vr_{i+} * t \end{cases}$$

The vector $T_r(0)$ can be updated to $T_r(t)$ at time t accordingly. PST query q can also be defined as a moving rectangle in the n-dimensional space, with the current space extents $[q_{i-}(0), q_{i+}(0)]$, velocities vq_{i-} and vq_{i+} $(1 \leq i \leq n)$, and the time interval $[q_{t-}, q_{t+}]$ $(0 \leq q_{t-} \leq q_{t+})$.

Spatio-temporal histogram is generally used to do selectivity estimation for the PST query. We usually construct a spatio-temporal histogram on a data set D through the following two phases.

Phase 1. Partition D into k disjointing parts, called buckets.

Phase 2. Each bucket records four kinds of information as follows:

1. MBB (minimum bounding box) of objects in the bucket is represented as extent on each dimension, i.e., $(b_{1-}, b_{1+}, ..., b_{n-}, b_{n+})$.
2. Minimum velocity vb_{i-} of lower boundaries and Maximum velocity vb_{i+} of upper boundaries among the objects in the bucket (MVBB). i.e., $(vb_{1-}, vb_{1+}, ..., vb_{n-}, vb_{n+})$.
3. Average length $\overline{b_i}$ of the extents, and average difference $\overline{vb_i}$ between the velocity at the lower boundary and the upper boundary per object on each dimension i.
4. The number of objects N_b in the bucket.

Figure 2 shows the information stored in a sample bucket b.

b_{1-}	b_{1+}	....	b_{n-}	b_{n+}	vb_{1-}	vb_{1+}	....	vb_{n-}	vb_{n+}	$\overline{b}_1$	....	$\overline{b}_n$	$\overline{vb}_1$	....	$\overline{vb}_n$	N_b

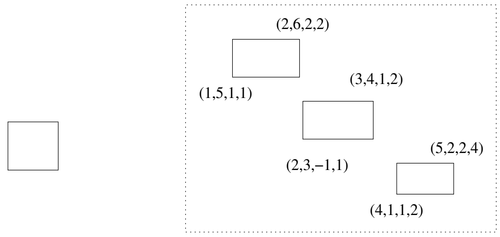
Figure 2: Information stored in a Bucket

For example, as depicted in Figure 3a, a bucket at time zero contains three 2-D rectangles. Numbers on the left-lower and right-higher corners of rectangle *r* represents $(r_{1-}, r_{2-}, vr_{1-}, vr_{2-})$ and $(r_{1+}, r_{2+}, vr_{1+}, vr_{2+})$ respectively. Figure 3b illustrates the contents stored in the bucket.

2.2 Related Work

There are three papers addressing selectivity estimation of the PST query. We will give a brief introduction on all of them.

The problem was first investigated in (Choi & Chung 2002). The main focus was on managing moving data



(a)

1	5	1	6	−1	2	1	4	1	1	1.33	1.33	3

(b)

Figure 3: Example of a Bucket

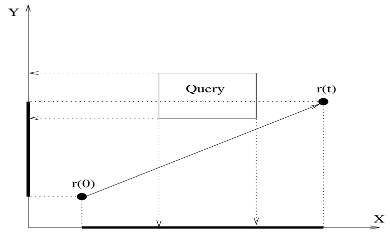


Figure 4: An Over Estimation on 2-D Space

points and static queries. Starting from a thorough discussion on estimating a 1-D line segment intersecting with a query window, (Choi & Chung 2002) presented formulae to calculate the approximate results of selectivity estimation under this 1-D environment. A spatio-temporal histogram is constructed by partitioning the spatial domain into several disjoint line segments as buckets through the Minskew technique. Then for each bucket, MBB, $MVBB$ and additional information are stored. To extend it into 2-D environment, the algorithm in (Choi & Chung 2002) first projected those data objects and queries into one dimension and multiplied the estimation results for another dimension. Two main drawbacks of this histogram technique are:

1. Frequent updates are needed in maintaining a good performance of this histogram
2. The method on estimating 2-D moving points sometimes overestimates the query results. Figure 4 shows that a data point r moving from $r(0)$ to $r(t)$ may intersect with the query window on X and Y dimensions during the time interval $[0, t]$, but it disjoints with the query.

In (Hadjieleftheriou et al. 2003), a different approach is proposed to handle the moving data points and the static queries. Those data points and queries are firstly transformed from the *primal space-time* space to the *dual velocity-intercept* space (Choi & Chung 2002, Khaled Elbassioni 2003, Jagadish 1990, Kollios et al. 1999). Then a histogram is constructed for selectivity estimation in the dual velocity-intercept space. Again Minskew technique was used in constructing the histogram. Figure 5 presents an example of this kind of transformation. Two moving points r_0, r_1 and a window query q are plotted both in the primal and the dual space. The gray trapezoid area in Figure 5b corresponds to the query window in Figure 5a. As depicted in 5, a point (r_0 or r_1) intersects with q if it is inside the gray trapezoid area. Note that the techniques in (Hadjieleftheriou et al. 2003), like that of (Choi & Chung 2002), are applicable to points only. Further, transforming a rectangular window into a trapezoid window may potentially increase the problem complexity.

(Tao et al. 2003) proposed another novel histogram technique, as well as a comprehensive query model to

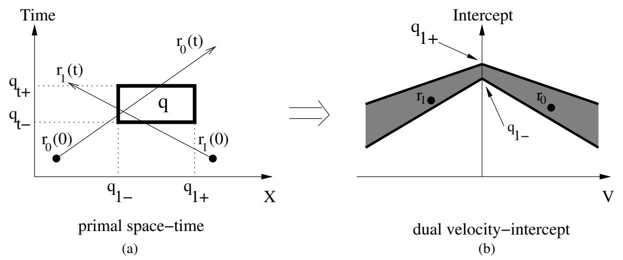
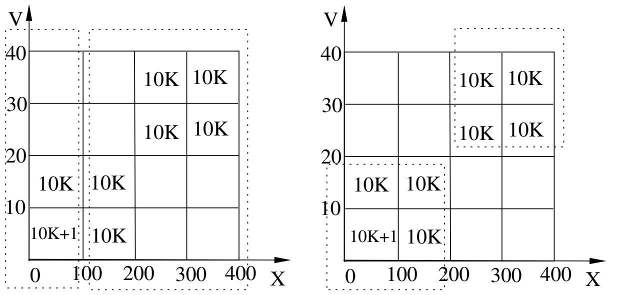


Figure 5: Primal Space *VS.* Dual Space



(a) Minskew Partition

(b) Better Partition

Figure 6: Two different partitions

give a general solution to the PST query. Using the Minskew technique, it enforces the uniformity regarding both spatial and velocity during the histogram constructing phase; it can also generally process boxes. These overcome the limitation of the other existing techniques. (Tao et al. 2003) also presented a query model for a general PST query. This query model will be used in our work and its details are introduced in the following subsection.

2.3 Selectivity Estimation using Histogram

Using histograms to do selectivity estimation in spatio-temporal databases is more complicated than that in relational databases. Under the query model proposed by (Tao et al. 2003), the approximate result of a PST aggregation query is the summation of the estimated number of objects in each bucket retrieved by the query. Note that in each bucket, data objects are assumed to follow the uniform distribution. Thus, for a bucket b and a query q, the estimation result, Est_b, is:

$$Est_b = N_b * \prod_{i=1}^{n}\left(\frac{1}{V_{i,max} - \overline{vb_i} - V_{i,min}}\right) * \int_{V_{1,min}}^{V_{1,max}-\overline{vb_1}} \cdots \int_{V_{n,min}}^{V_{n,max}-\overline{vb_n}} P(u_1, u_2, ..., u_n) du_1 du_2 ... du_n \quad (1)$$

Where $P(u_1, u_2, ..., u_n)$ is the probability that a data object *r*, whose velocity vr_{i-} on *i*-th dimension takes value u_i, will intersect with the query window during time interval $[q_{t-}, q_{t+}]$. The calculation of the probability $P(u_1, u_2, ..., u_n)$ follows a series of transformation proposed by (Tao et al. 2003). With the computed $P(u_1, u_2, ..., u_n)$, we can calculate Est_b in Equation 1 through numeric integration method (Press, Teukolsky, Vetterling & Flannery 2002). The final selectivity estimation result, over k buckets, is $\sum_{b=1}^{k} Est_b$.

2.4 Motivation

The accuracy of existing histogram techniques in spatial-temporal databases is mainly based on that of the Minskew technique (Acharya et al. 1999) where the *marginal skewness* is used to partition the original data space to avoid expensive computation costs. The application of marginal skewness, however, may limit the choices of space partition. Figure 6 shows such an example. Suppose we consider 1-D data points in a $4X4$ grid. The value in each grid represents the number of data points falling in that grid. If we want to build a histogram with two buckets on the original data space, Figure 6a gives a possible partition under Minskew technique. However, the obvious best partition is depicted in Figure 6b which can never be obtained by the Minskew technique. The *total skewness of buckets* (Acharya et al. 1999) in Figure 6b is much smaller than that in Figure 6a. Thus the partition in Figure 6b is expected to have a better performance in the selectivity estimation. Moreover, the right bucket of Figure 6a has objects whose velocities distributing in the whole velocity domain. This makes the bucket's skewness even worse in the future time. Motivated by the above observations, we propose a novel spatio-temporal histogram (CSTH) construction technique based on the clustering technique.

3 Clustering Moving Objects

The section is organized as follows. Section 3.1 discusses the construction of CSTH. Section 3.2 presents a linear time heuristic which aims to improve the performance of the histogram. Section 3.3 explains how to dynamically maintain CSTH.

3.1 CSTH Construction

The basic idea of constructing CSTH is to cluster data objects with similar properties (e.g. initial positions, velocities and sizes). Before presenting how to construct a CSTH with k buckets for a spatio-temporal dataset that contains n-dimensional moving objects, we firstly introduce the *distance* of moving objects. The distance of two objects represents the similarity between them. A small distance means they are quite similar to each other (i.e. they have similar in initial positions, velocities and sizes). As mentioned earlier, a $4n$-dimensional vector can be used to represent a n-dimensional moving object. The distances of those objects can then be defined as the distances of the corresponding vectors. Note that the Euclidean distance is not quite suitable in this application. Since in a spatio-temporal dataset, the object's initial position and velocity are both important to decide whether this object validates a PST query. We modify the Euclidean distance by doing a normalization on the spatial and velocity part respectively. This normalization ensures spatial variation and velocity variation equally important in measuring the similarity between two moving objects. The formal definition of our distance function is:

Let $s = \max_{1\le i\le n}(U_{i,max} - U_{i,min})$ and $t = \max_{1\le i\le n}(V_{i,max} - V_{i,min})$. $\delta(T_x, T_y)$ is the distance function of vector T_x and T_y. Note that in the rest of the paper, $T_r(0)$ is abbreviated to T_r for a data object r unless it creates ambiguity.

$$\delta(T_x, T_y) = \frac{\sqrt{\sum_{i=1}^{n}(x_{i-} - y_{i-})^2 + (x_{i+} - y_{i+})^2}}{s * \sqrt{n}} + \frac{\sqrt{\sum_{i=1}^{n}(vx_{i-} - vy_{i-})^2 + (vx_{i+} - vy_{i+})^2}}{t * \sqrt{n}} \quad (2)$$

It is easy to verify that $\delta(T_x, T_y)$ satisfies:

Reflexive: $\delta(T_x, T_x) = 0$

Symmetric: $\delta(T_x, T_y) = \delta(T_y, T_x) = 0$

Triangle Inequality: $\delta(T_x, T_y) + \delta(T_y, T_z) \geq \delta(T_x, T_z)$

Thus $\delta(T_x, T_y)$ is a metric and the $4n$-dimensional vector space is a metric space.

Based on the distance function $\delta(T_x, T_y)$, we define the sum of spatial-temporal deviation (*SSTD*) of a given bucket b containing N_b objects as:

$$SSTD_b = \sum_{r=1}^{N_b} \delta(T_r, T_b),$$

where T_b is the vector representing the centroid of bucket b:

$$T_b = \{\frac{\sum_{r=1}^{N_b} r_{1-}}{N_b}, ..., \frac{\sum_{r=1}^{N_b} r_{n+}}{N_b}, \frac{\sum_{r=1}^{N_b} vr_{1-}}{N_b}, ..., \frac{\sum_{r=1}^{N_b} vr_{n+}}{N_b}\}$$

The goal of constructing CSTH with k buckets is to find a partition such that the summation of each bucket's SSTD (HSSTD) is minimized. This is actually the well-known k-Median clustering problem in metric space, which is NP-hard. A simple linear time heuristic algorithm, k-Means, is always used in practice to give approximate answer to the k-Median problem. The deficiency of the k-Means algorithm is that it has no error guarantee. Approximate algorithms with error guarantee are proposed in (Lin & Vitter 1992, Charikar, Guha, Tardos & Shmoys 1999, Jain & Vazirani 1999, Arya, Garg, Khandekar, Meyerson, Munagala & Pandit 2001). However, those techniques are not practical for large datasets because of high time complexities.

To obtain a practical approximate technique with a precision guarantee, we can first modify our optimization goal from minimizing $HSSTD$ to minimizing "$MSTD$" (the maximum spatio-temporal deviation over all buckets). The formal definition of $MSTD$ for a histogram with k buckets is:

$$MSTD = \max_{1 \leq b \leq k} (\max_{1 \leq r \leq N_b} (\delta(T_r, T_b))$$

It can be immediately verified that $HSSTD \leq MSTD * \sum_{1 \leq b \leq k} N_b$. Therefore, $HSSTD$ will be reduced by minimizing $MSTD$. Note that minimizing $MSTD$ is another well-known clustering problem, k-Center problem (Gonzalez 1985). Unfortunately, it is also NP-hard. However, it has a good approximate algorithm which achieves an approximation ratio 2 with a linear time complexity. That algorithm was first proposed by Gonzalez at (Gonzalez 1985). It starts from a randomly picked data object as the first center, then the algorithm scans the whole data set to find a data which has the maximum distance to it's closest center. That data will be picked as the second center. This process is repeated until all k centers are found. Then the data space can be partitioned into k clusters according to those k centers. Figure 7 shows an example of using Gonzalez algorithm to partition the data set into four clusters. Based on this clustering algorithm, our new histogram is constructed by Algorithm 1. Experiments demonstrated that CSTH greatly reduces the average relative error compared to the spatio-temporal histogram technique in (Tao et al. 2003).

3.2 Refining Histograms

In this subsection we propose a refinement algorithm, which refines an existing histogram by reducing its HSSTD. This refinement can be applied to any already built histogram; thus it can be used as a post-processor. The refinement starts from a randomly picked bucket. For

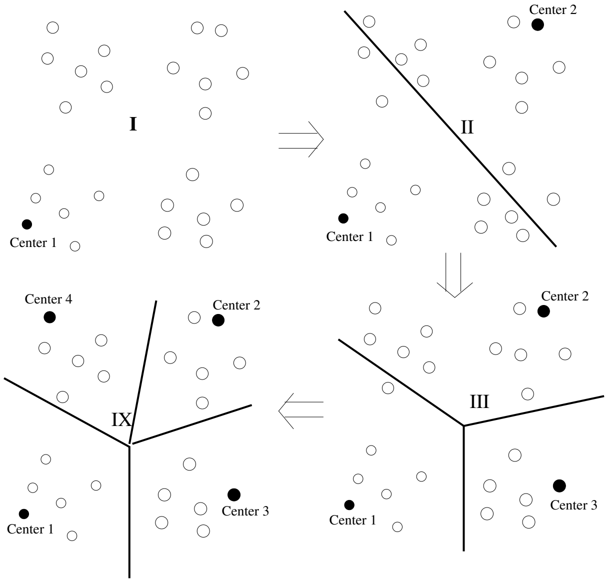
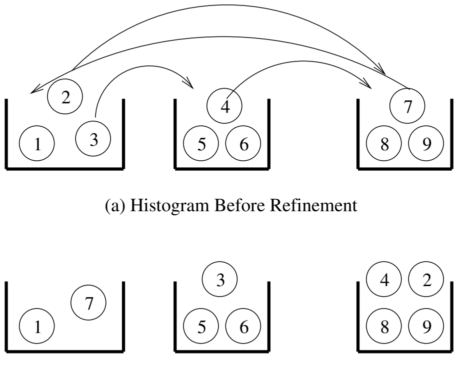


Figure 7: Gonzalez Clustering



(a) Histogram Before Refinement

(b) Histogram After Refinement

Figure 8: Example of Refining Histogram

each data object in this bucket, it calculates the changes of $HSSTD$ by iteratively re-assigning this object to the other $k - 1$ buckets one by one. The re-assignment with the biggest reduction of $HSSTD$ will be actually executed. If no reduction is possible for any of those reassignments, the algorithm moves to the next bucket. The algorithm terminates after it examines all the buckets. To avoid expensive computation cost, this refining procedure is forward only. That is, if a data object has been moved once, it will not be considered as a re-assignment candidate again. Figure 8 gives an example of a histogram before and after the refinement. Details of the algorithm are presented in Algorithm 2.

Note that the computation of the best re-assignment of an object among the k buckets can run in time $O(k)$ if each refinement can be done in a constant time. However, due to the complexity form of $HSSTD$ it is impossible to do this. Below we use an upper-bound of $HSSTD$ as the refinement goal function to make this happen. This makes our refining algorithm have a linear time complexity $O(nk)$ (n is the number of objects in the database).

Supporting a Linear Time Refinement

Let D be a n-dimensional spatial-temporal data set which contains N data objects. Let b represent a bucket of the histogram with k buckets ($1 \leq b \leq k$). Let r represent a data object in D. We use $r_i, r_{i+1}(1 \leq i \leq 2n)$ to represent $r_{i-}, r_{i+}(1 \leq i \leq n)$, and $vr_i, vr_{i+1}(1 \leq i \leq 2n)$ to represent $vr_{i-}, vr_{i+}(1 \leq i \leq n)$ in the vector T_r. Let

Algorithm 1 Constructing CSTH

Input:
Spatial-Temporal Data Set D, bucket number k
Output:
Histogram B with k buckets
Description:

```
The Set of Clustering Centers C := φ;
Randomly choose an object r_1 as the first center;
C := C ∪ {r_1}
k := k − 1;
while k > 0 do
    for each r in D do
        d_r := min_{i∈C} δ(T_r, T_i);
    end for
    r' := max_{r∈D} d_r;
    C := C ∪ {r'};
    k := k − 1;
end while
for each r in D do
    assign r to closest center in C to form k buckets;
end for
for each bucket do
    storing MBB, MVBB and other information of data
    objects in the bucket;
end for
output the histogram B with those k buckets;
```

s represent the maximum value of spatial domain, and t represent the maximum value of velocity domain.

$$\begin{cases} s = \max(U_{i,\max} - U_{i,\min}), 1 \le i \le 2n \\ t = \max(V_{i,\max} - V_{i,\min}), 1 \le i \le 2n \end{cases}$$

From the inequity

$$\left(\sum_{i=1}^{n} x_i\right)^2 \le n \sum_{i=1}^{n} x_i^2,$$

we get:

$$HSSTD = \sqrt{HSSTD^2} \le \sqrt{N * \sum_{b=1}^{k}\sum_{r=1}^{N_b} \delta^2(T_r, T_b)}$$
$$\le \sqrt{2N * \sum_{b=1}^{k}\sum_{r=1}^{N_b} \frac{\sum_{i=1}^{2n}(r_i - \frac{\sum_{r=1}^{N_b} r_i}{N_b})^2}{s^2 * n} + \frac{\sum_{i=1}^{2n}(vr_i - \frac{\sum_{r=1}^{N_b} vr_i}{N_b})^2}{t^2 * n}} \tag{3}$$

To simplify Equation 3, let $\overline{r_{ib}} = \frac{\sum_{r=1}^{N_b} r_i}{N_b}$, and $\overline{vr_{ib}} = \frac{\sum_{r=1}^{N_b} vr_i}{N_b}$. After elementary mathematical deduction, Equation 3 can be reduced to:

$$HSSTD \le \sqrt{\frac{2N}{n} * \sum_{b=1}^{k} \frac{\sum_{i=1}^{2n}(\sum_{r=1}^{N_b} r_i^2 - N_b \overline{r_{ib}}^2)}{s^2} + \frac{\sum_{i=1}^{2n}(\sum_{r=1}^{N_b} vr_i^2 - N_b \overline{vr_{ib}}^2)}{t^2}} \tag{4}$$

Since $HSSTD$ is tightly bounded in Equation 4, we will use the right-hand part (HVAR) of this equation as the goal function to choose the best re-assignment.

Lemma 1 *Suppose that we use the goal function HVAR in Algorithm 2. The computation of the best re-assignment, of an object, among k buckets (line 7) can run in time $O(k)$ by keeping the average value of r_{i-}, r_{i+}, vr_{i-} and vr_{i+} for each bucket on each dimension i.*

Algorithm 2 Forward-Only Refining

Input:
Histogram B_{in}
Output:
Histogram B_{out}
Description:

```
1:  k := number of buckets in B_in;
2:  while k > 0 do
3:      IMPROVABLE = false;
4:      choose next bucket b in B_in;
5:      for each data object r in b do
6:          if r is not moved from other buckets then
7:              pick the best reassigning plan of r;
8:              reassign r following the best plan;
9:              IMPROVABLE = true;
10:         end if
11:     end for
12:     if IMPROVABLE = FALSE then
13:         stop the algorithm
14:     end if
15:     k := k-1;
16: end while
```

Proof:
It can be immediately verified that if we keep $\overline{r_{ib}}$ and $\overline{vr_{ib}}$ of each dimension i in bucket b, we can compute the change of $HVAR$, regarding adding or removing one object, in constant time. Thus picking the best reassignment in algorithm 2 can be finished with $O(k)$ time complexity. ■

Our experiments demonstrated that the effectiveness of the Tuning algorithm. We will give detail discussions in the experiment part.

3.3 Update Maintenance

In the PST query scenario, an update includes three situations:

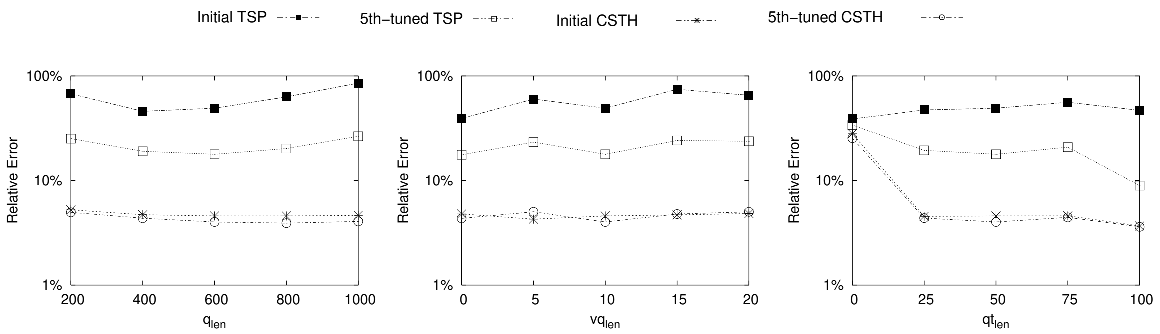
- inserting a new data object
- updating a data object's velocity
- removing a data object from the database

Since the second situation can be viewed as the combination of first and third situations. We will mainly discuss how to maintain $CSTH$ when some data object's velocity has been changed. Without lose of generality, we consider a 1-D object r, whose velocity changes from $vr(0)$ to $vr(t)$ at time t. r's locations at time zero and t are $r(0)$ and $r(t)$ respectively. It is obvious that we can not insert the updated speed $vr(t)$ and location $r(t)$ in our histogram since the histogram is constructed at time zero. However, using $r(t)$ and $vr(t)$, we can derive dummy data object r' at time zero, which will move to position $r(t)$ at time t with velocity $vr(t)$. Thus r' can be computed as:

$$\begin{cases} r'(0) = r(t) - t * vr(t) \\ vr'(0) = vr(t) \end{cases}$$

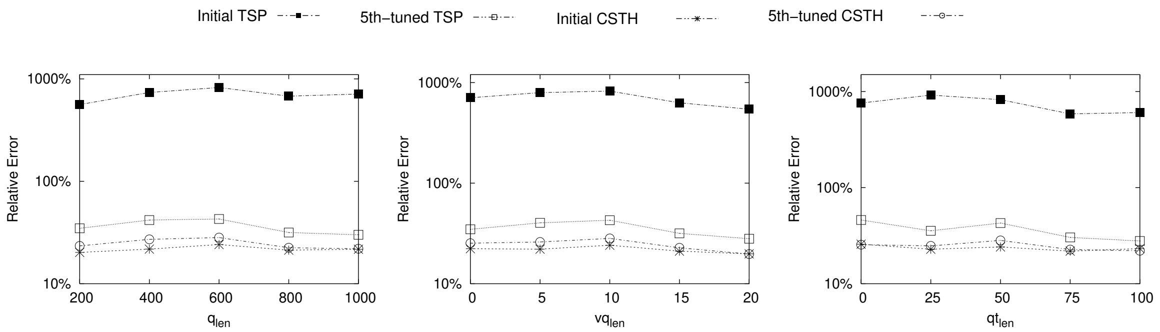
This r' will be inserted into the histogram. Since the buckets of $CSTH$ may overlap with each other, the scan of more than one bucket is needed for inserting the object r'. If r' is contained by several buckets, we choose the bucket whose centroid is closest to r' to increase the number of the objects by 1. The final step of update is to remove r from the original bucket. This can be done by decreasing the number of the objects in this original bucket by 1.

Sometimes if new added objects fall out of the bucket's range, the borders of corresponding bucket need to be recalculated. If too many updates happens, the histogram may need to be re-constructed. This operation can be triggered by setting up a threshold.



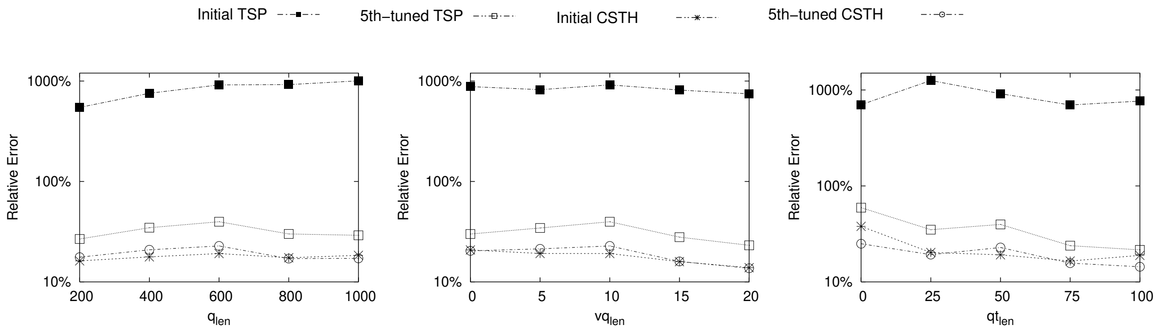
(a) $q_{len}(vq_{len} = 10, qt_{len} = 50)$ (b) $vq_{len}(q_{len} = 600, qt_{len} = 50)$ (c) $qt_{len}(q_{len} = 600, vq_{len} = 10)$

Figure 9: Accuracy Comparison for Uniform Data Set



(a) $q_{len}(vq_{len} = 10, qt_{len} = 50)$ (b) $vq_{len}(q_{len} = 600, qt_{len} = 50)$ (c) $qt_{len}(q_{len} = 600, vq_{len} = 10)$

Figure 10: Accuracy Comparison for Carec Data Set



(a) $q_{len}(vq_{len} = 10, qt_{len} = 50)$ (b) $vq_{len}(q_{len} = 600, qt_{len} = 50)$ (c) $qt_{len}(q_{len} = 600, vq_{len} = 10)$

Figure 11: Accuracy Comparison for Capoint Data Set

4 Experiments

In this section, we present extensive experiment results to evaluate our two new techniques. We will use the algorithm in (Tao et al. 2003) as a bench mark algorithm in our experiments since it is the best existing technique; we use their source code in the experiment. All experiments were performed on a Pentium III 700Mhz CPU, 256 MB memory computer with Linux 2.4.20.

Data Sets

Each spatio-temporal data set contains 2-D moving data points or rectangles. Three data sets are used in our experiments:

Uniform This data set contains one million synthetic data points whose initial positions follow a uniform distribution in the 2-D spatial space $[0, 10000]^2$. Every point's velocity also takes a uniform distribution from $[-50, 50]$ on each dimension.

CArec The spatial parts of this data set is taken from a real spatial data set CA [1], which contains 2.2 million rectangles to represent streets in California. We normalize the data set into a 2-D spatial space $[0, 10000]^2$. Due to lack of real moving objects, we need to generate velocities for those spatial data. For a fair comparison, we adopt the velocity generating method used in (Tao et al. 2003). A moving rectangle r's velocity on each dimension i follows:

- The absolute value of vr_{i-} follows a *Zipf* Distribution (Zipf 1949) from $[0, 50 - (vr_{i+} - vr_{i-})]$, with Zipf coefficient 0.8.
- vr_{i-} has the same probability to take a positive or a negative value.
- The value of $vr_{i+} - vr_{i-}$ follows a uniform distribution from $[0, 5]$.

CApoint This data set contains 2.2 million moving data points generated from the CArec data set.

- The point's initial position is the centroid of the moving rectangles of CArec data set.
- On each dimension, the point's velocity is the average of the corresponding lower and upper boundary velocities of the moving rectangles in CArec data set.

Query Workloads

A predicative spatio-temporal query q is also a moving rectangle. It is a square in the 2D space with side length q_{len}, where $q_{len} = q_{i+}(0) - q_{i-}(0)$. It's velocity extent (i.e., the difference of the velocities of the lower and upper boundaries) on each dimension equals a constant vq_{len}, where $vq_{len} = vq_{i+} - vq_{i-}$. The query interval time of q is qt_{len}, where $qt_{len} = q_{t+} - q_{t-}$. Queries are divided into several groups. Each group contains 200 randomly generated queries with the same value q_{len}, vq_{len} and qt_{len}. The left boundary of the query q's spatial extent, the velocity extent and the time interval follow the uniform distribution respectively. That is, q_{i-} is taken uniformly from $[0, 10000 - q_{len}]$, vq_{i-} from $[-50, 50 - qv_{len}]$ and q_{t-} from $[0, 100 - qt_{len}]$.

Error Metrics

Let A_i denotes the actual result of a query q_i and A'_i denotes the approximate result. The relative error is define as:

$$e_i^{rel} = \begin{cases} \frac{|A_i - A'_i|}{A_i} & if A_i \neq 0 \\ A'_i & otherwise \end{cases}$$

The average relative error of N queries is defined as:

$$\overline{e_N^{rel}} = \frac{\sum_{i=1}^{N} e_i^{rel}}{N}$$

Note that here we use different error metrics to that used in (Tao et al. 2003) since we believe it is more sensitive. Our error metrics is also commonly used in evaluating selectivity estimation of range queries and spatio-temporal queries; for example, (Poosala, Haas, Ioannidis & Shekita 1996, Matias, Vitter & Wang 1998, Choi & Chung 2002, Hadjieleftheriou et al. 2003).

Accuracy Comparison

Firstly, we construct CSTH (Initial CSTH) and the histogram proposed by (Tao et al. 2003) (Initial TSP). Then we apply the refining algorithm (Algorithm 2) on those two histograms respectively. Each histogram has been refined 5 times (i.e., Algorithm 2 is consecutively executed 5 times). We recorded error rates of the initial histograms and their fifth refined histograms. Different experiment results for the three mentioned data sets are shown in Figure 9, 10 and 11. In the experiments of each data set, queries are divided into three groups, a, b and c. Group a (Figure 9a, 10a and 11a) plots the average error rates on various q_{len}, with fixed $vq_{len} = 10$ and $qt_{len} = 50$. Group b (Figure 9b, 10b and 11b) shows the error rates on different vq_{len} with fixed $q_{len} = 600$ and $qt_{len} = 50$. Group c (Figure 9c, 10c and 11c) evaluates the error rates on diverse qt_{len} with fixed $q_{len} = 600$ and $vq_{len} = 10$.

From the results of our experiments, we can see that the Initial CSTH outperforms Initial TSP. Especially in the real data set where objects has skew velocity distribution (Figure 10 and 11), Initial TSP has an average error rate around 1000%, while Initial CSTH has an error rate below 20%. Our refining algorithm also has the expected refinement affects when applied to Initial TSP. It can improve the accuracy by one magnitude of order (Figure 10 and 11). However, when the refining algorithm is applied to the Initial CSTH, the benefits of the refinement is not quite obvious. Sometimes, refined histogram even performs worse than the original one (Figure 10, 11a and 11b). This shows that CSTH is already a good approach to do selective estimation on PST queries.

5 Conclusion and Future work

In this paper, we presented a new histogram construction method to approach the selectivity estimation for predictive spatio-temporal queries. It is based on a clustering paradigm. To avoid expensive computation costs, we developed a linear time heuristic to construct CSTH - a new clustering based spatio-temporal histogram. Our experiments demonstrated that this new histogram technique significantly outperforms existing histogram techniques, especially for skew real data. We also presented a linear time refining algorithm to refine already built histograms as a post-processor. Experiment results demonstrated the effectiveness of this refinement. In fact, our new techniques may improve the accuracy of the existing histogram techniques by one order of magnitude.

Although CSTH can be constructed in a linear time and greatly improve the accuracy of the existing techniques, it is not as efficient as the existing techniques (Tao et al. 2003) in our experiments. Speed up the computation is a possible future work. We will also investigate other clustering models to approach various estimation problems in spatio-temporal datasets.

[1] http://dke.cti.gr/People/ytheod/research/ datasets/spatial.html

Acknowledgements

We would like to thank Dr. Yufei Tao from City University of Hong Kong for providing us the source codes that were used in (Tao et al. 2003).

References

Acharya, S., Poosala, V. & Ramaswamy, S. (1999), Selectivity estimation in spatial databases, *in* 'ACM SIGMOD'99'.

Arya, V., Garg, N., Khandekar, R., Meyerson, A., Munagala, K. & Pandit, V. (2001), Local search heuristics for k-median and facility location problems, *in* 'ACM STOC'01, ACM Symposium on Theory of Computing'.

Charikar, M., Guha, S., Tardos, E. & Shmoys, D. B. (1999), A constant-factor approximation algorithm for the k-median problem, *in* 'ACM STOC'99, Symposium on Theory of Computing'.

Choi, Y.-J. & Chung, C.-W. (2002), Selectivity estimation for spatio-temporal queries to moving objects, *in* 'ACM SIGMOD'02, Special Interest Group on Management of Data'.

Erwig, M., Gting, R. H., Schneider, M. & Vazirgiannis, M. (1999), 'Spatio-temporal data types: An approach to modeling and querying moving objects in databases', *GeoInformatica* **3**(3), 265–291.

Forlizzi, L., Gting, R. H., Nardelli, E. & Schneider, M. (2000), A data model and data structures for moving objects databases, *in* 'ACM SIGMOD'00'.

Gonzalez, T. F. (1985), 'Clustering to minimize the maximum intercluster distance', *Theoretical Computer Science* **38**, 293–306.

Hadjieleftheriou, M., Kollios, G. & Tsotras, V. J. (2003), Performance evaluation of spatio-temporal selectivity estimation techniques, *in* 'SSDBM'03, 15th International Conference on Scientific and Statistical Database Management.'.

Ioannidis, Y. (1993), Universality of serial histograms, *in* 'Proceedings of the 19th Conference on Very Large Databases, Morgan Kaufman pubs. (Los Altos CA), Dublin'.

Ioannidis, Y. E. & Poosala, V. (1999), Histogram-based approximation of set-valued query-answers, *in* 'The VLDB Journal', pp. 174–185.

Jagadish, H. V. (1990), On indexing line segments, *in* 'VLDB'90'.

Jain, K. & Vazirani, V. V. (1999), Primal-dual approximation algorithms for metric facility location and k-median problems, *in* 'FOCS'99, IEEE Symposium on Foundations of Computer Science'.

Khaled Elbassioni, Amr Elmasry, I. K. (2003), An efficient indexing scheme for multi-dimensional moving objects, *in* 'ICDT'03, The 9th International Conference on Database Theory'.

Kollios, G., Gunopulos, D. & Tsotras, V. J. (1999), On indexing mobile objects, *in* 'ACM, PODS'99'.

Lin, J.-H. & Vitter, J. S. (1992), Approximation algorithms for geometric median problems, *in* 'Inform. Proc. Lett.', 44, pp. 245–249.

Matias, Y., Vitter, J. S. & Wang, M. (1998), Wavelet-based histograms for selectivity estimation, pp. 448–459. *citeseer.nj.nec.com/matias98waveletbased.html

Muralikrishna, M. & DeWitt, D. J. (1988), Equi-depth histograms for estimating selectivity factors for multi-dimensional queries, *in* 'Proceedings of the 1988 ACM SIGMOD International Conference on Management of Data, Chicago, Illinois, June 1-3, 1988', ACM Press.

Pankaj K. Agarwal, Lars Arge, J. E. (2000), Indexing moving points, *in* 'ACM PODS'00, Symposium on Principles of Database Systems'.

Poosala, V. (1997), Histogram-Based Estimation Techniques in Database Systems, PhD thesis, University of Wisconsin-Madison.

Poosala, V., Haas, P. J., Ioannidis, Y. E. & Shekita, E. J. (1996), Improved histograms for selectivity estimation of range predicates, pp. 294–305. *citeseer.nj.nec.com/poosala96improved.html

Press, W. H., Teukolsky, S. A., Vetterling, W. T. & Flannery, B. P. (2002), *Numerical recipes in C*, Cambridge University Press.

Saltenis, S., Jensen, C. S., Leutenegger, S. T. & Lopez, M. A. (2000), Indexing the positions of continuously moving objects, *in* 'ACM SIGMOD'00'.

Tao, Y., Sun, J. & Papadias, D. (2003), Selectivity estimation for predictive spatio-temporal queries, *in* 'ICDE'03, Proceedings of the 19th International Conference on Data Engineering.'.

Zipf, G. K. (1949), *Human behaviour and the principle of least effort*, Addison-Wesley, Reading.