Approximation Techniques for Spatial Data [*]

Abhinandan Das
Cornell University
asdas@cs.cornell.edu

Johannes Gehrke
Cornell University
johannes@cs.cornell.edu

Mirek Riedewald
Cornell University
mirek@cs.cornell.edu

ABSTRACT

Spatial Database Management Systems (SDBMS), e.g., Geographical Information Systems, that manage spatial objects such as points, lines, and hyper-rectangles, often have very high query processing costs. Accurate selectivity estimation during query optimization therefore is crucially important for finding good query plans, especially when spatial joins are involved. Selectivity estimation has been studied for relational database systems, but to date has only received little attention in SDBMS. In this paper, we introduce novel methods that permit high-quality selectivity estimation for spatial joins and range queries. Our techniques can be constructed in a single scan over the input, handle inserts and deletes to the database incrementally, and hence they can also be used for processing of streaming spatial data. In contrast to previous approaches, our techniques return approximate results that come with provable probabilistic quality guarantees. We present a detailed analysis and experimentally demonstrate the efficacy of the proposed techniques.

1. INTRODUCTION

In recent years, data management for spatial applications such as Geographic Information Systems, the earth sciences, and environmental monitoring has gained significant importance. In spatial data management, records in the database have a spatial extent, and users can pose expressive queries such as a spatial join between two relations (join all objects that overlap or are within certain distance of each other) or a range query (report all objects in a selected range, or return an aggregate over the selected objects). Due to the spatial extent of the objects, executing spatial queries can be very expensive.

When selecting different query plans for a spatial query, we need accurate estimates of the cost of different execution strategies. Thus as in relational systems, we need accurate selectivity (cardinality) estimates. Note that these estimates can also be used for fast approximation of an aggregate query, e.g., a classical example is the approximate range aggregate; another interesting example is to use approximate join cardinality for correlation analysis between data sets.

The current state of the art in estimating the sizes of spatial joins uses either sampling or histograms, where recent work has shown that histograms are superior for a wide range of possible query classes [5, 26]. However, existing histogram-based techniques still have significant drawbacks. First, they either give no or only very conservative worst-case error guarantees that are usually overly pessimistic in practice. Due to the lack of error guarantees, it is not possible to quantify the tradeoff between storage and result accuracy (except by performing extensive experimental evaluation with a priori known data distributions). Second, existing histogram techniques are designed for static datasets, and require usually several passes over the data during construction. The only exception are histograms that use a fixed partitioning of the space (e.g., equi-width). These can be constructed in a single pass and can be maintained incrementally, but they cannot adapt to skewed or changing data distributions.

Our Contributions. In this paper, we introduce a novel framework for spatial data that is based on randomized projections. Our framework can handle a large set of challenging spatial operators, and it can handle objects with spatial extent. Our approach is grounded on randomizing techniques for computing small, pseudo-random *sketches* of the spatial dataset. The basic sketching technique was originally introduced for on-line self-join size estimation by Alon, Matias, and Szegedy in their seminal paper [4] and, as we demonstrate in our work, can be generalized to provide approximate answers to spatial join queries with explicit and tunable performance guarantees on the approximation error. The main challenge in using sketches for spatial data is to design selectivity estimation algorithms which are based on the *right way of counting* events like points being contained in intervals. The algorithm should not only determine the correct estimate *on expectation*, but its results should also have low variance.

More concretely, the key contributions of our work can be summarized as follows:

- We introduce the first summary data structure for spatial data that allows estimation of the selectivity of

[*] The authors are supported by NSF grants IIS-0330201, CCF-0205452, and IIS-0133481, and by a gift from Microsoft. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the sponsors.

SIGMOD 2004 June 13-18, 2004, Paris, France.
Copyright 2004 ACM 1-58113-859-8/04/06 . . . $5.00.

spatial queries with provable performance guarantees. It permits a graceful tradeoff between space consumption and the quality of the resulting estimation. We apply our spatial sketch technique to spatial joins of intervals and rectangles (Section 4). All of our techniques are accompanied by a thorough analytical evaluation in which we show probabilistic bounds on the quality of the resulting summary data structures.

- We show the generality of our technique by discussing several extensions: Joins of hyper-rectangles in a d-dimensional space where $d > 2$, spatial joins with other join predicates like "contained", ϵ-joins, and range queries (Section 6).
- In a thorough experimental evaluation, we compare our techniques with the best previously known estimation techniques for spatial joins, the Geometric and the Euler histograms (Section 7).

Note that, even though we develop our sketching algorithms in the classic database context of stored relations, our techniques are more generally applicable to scenarios where only a single pass over the data is possible — from streams of spatial data to huge Terabyte databases where performing multiple passes over the data for the exact computation of query results may be prohibitively expensive.

The rest of the paper is organized as follows. Section 2 defines spatial queries and discusses the sketching approach of [4]. In Section 3 we introduce the basic atomic sketches which will be used to construct selectivity estimators. Selectivity estimation for joins of interval sets and rectangle sets are discussed in Section 4. Sections 5 and 6 show how to generalize the technique to virtually any practical setting and how to extend our techniques to the d-dimensional case, other join conditions, and other spatial operators. Experimental results are discussed in Section 7, and Section 8 reviews related work. Section 9 concludes this article.

2. BACKGROUND

2.1 Queries

We focus mostly on selectivity estimation for spatial joins, more precisely spatial joins of sets of hyper-rectangles and ε-joins of point sets. These are the most common types of joins in spatial DBMS. In the following, let $\mathcal{N}$ be a (discrete) metric space and let $r = r(1) \times r(2) \times \cdots \times r(d)$ be a hyper-rectangle which is defined by range $r(i)$ in dimension i, $1 \leq i \leq d$. A range $r(i)$ is defined by its lower and upper endpoint $l(r(i))$ and $u(r(i))$ such that $r(i) = \{x \in \mathcal{N} | l(r(i)) \leq x \leq u(r(i))\}$. Hence $r = \{x = (x_1, x_2, \ldots, x_d) \in \mathcal{N}^d | \forall 1 \leq i \leq d : l(r(i)) \leq x_i \leq u(r(i))\}$. Note that our techniques easily generalize to multidimensional data spaces with different domains for the dimensions. In Section 5.1 we discuss how to handle real-valued domains.

Definition 1. Let R and S be two sets of hyper-rectangles in d-dimensional space $\mathcal{N}^d$. Function `overlap` for two hyper-rectangles $r(1) \times r(2) \times \cdots \times r(d) \in R$ and $s(1) \times s(2) \times \cdots \times s(d) \in S$ returns `true` if $\forall 1 \leq i \leq d : l(r(i)) < l(s(i)) < u(r(i)) \vee l(r(i)) < u(s(i)) < u(r(i)) \vee l(s(i)) < l(r(i)) < u(s(i)) \vee l(s(i)) < u(r(i)) < u(s(i))$, and `false` otherwise. The *spatial join* of R and S then is defined as

$$R \bowtie_o S = \{(r, s) | r \in R \wedge s \in S \wedge \texttt{overlap}(r, s)\} ,$$

and hence its selectivity is $\frac{|R \bowtie_o S|}{|R| \cdot |S|}$.

According to this definition objects that only "touch at their boundaries" are not part of the join result. We will examine possible generalizations in Section 6.

Definition 2. Let A and B be two sets of points in d-dimensional space $\mathcal{N}^d$ and `dist` be a function that returns the distance between a point in A and a point in B. The *ε-join* of A and B then is defined as

$$A \bowtie_\varepsilon B = \{(a, b) | a \in A \wedge b \in B \wedge \texttt{dist}(a, b) \leq \varepsilon\} .$$

Its selectivity is the cardinality of the result divided by the total number of point-pairs, i.e., $\frac{|A \bowtie_\varepsilon B|}{|A| \cdot |B|}$.

Typically the distance $\texttt{dist}(a, b)$ of two points $a = (a_1, \ldots, a_d)$ and $b = (b_1, \ldots, b_d)$ is computed by using an L_i-distance $\texttt{dist}_{L_i}$, or $\texttt{dist}_i$ for short, where

$$\texttt{dist}_i(a, b) = (\sum_{i=1}^{d} |a_i - b_i|^i)^{\frac{1}{i}} ,$$

including the L_∞ distance

$$\texttt{dist}_\infty(a, b) = \max\{|a_1 - b_1|, \ldots, |a_d - b_d|\} .$$

We also examine selectivity estimation for range queries.

Definition 3. Let R be a set of d-dimensional hyper-rectangles and $q = (q(1) \times q(2) \times \cdots \times q(d))$ be the query hyper-rectangle such that range $q(i)$ is selected in dimension i. This *range query* selects the set $Q(q, R) = \{r \in R | \texttt{overlap}(r, q)\}$. The selectivity is defined as $\frac{|Q(q,R)|}{|R|}$

Notice that for all introduced query types the challenge is to compute the result cardinality. Knowing the cardinality, we can compute the selectivity by simply keeping track of the input cardinalities. Hence in the following we are concerned with estimating the *cardinality* of the results of spatial queries.

2.2 AMS Sketches

Sketches were proposed for processing data streams. Consider a simple scenario where the goal is to estimate the size of the self-join $\mathrm{SJ}(A)$ of relation R over one of its attributes $R.A$ as the tuples of R are streaming in; that is, we seek to approximate the result of query $Q = \texttt{COUNT}(R \bowtie_A R)$. W.l.o.g. let the domain of join attribute A be $\texttt{dom}(A) = \{0, 1, \cdots, |\texttt{dom}(A)| - 1\}$, where $|\texttt{dom}(A)|$ denotes the size of the domain. Using $f(i)$ to denote the frequency of attribute value i in $R.A$, we can rewrite query Q as $Q = \mathrm{SJ}(A) = \sum_{i \in \texttt{dom}(A)} f(i)^2$ (i.e., the *second moment* of A). In their seminal paper, Alon, Matias, and Szegedy [4] prove that *any deterministic algorithm* that produces a tight approximation to $\mathrm{SJ}(A)$ requires at least $\Omega(|\texttt{dom}(A)|)$ bits of storage, rendering such solutions impractical for a data-stream setting. Instead, they propose a *randomized technique* that offers strong probabilistic guarantees on the quality of the resulting $\mathrm{SJ}(A)$ approximation while using only logarithmic space in $|\texttt{dom}(A)|$.

The basic idea of their scheme is to define a random variable Z that can be easily computed over the streaming values of $R.A$, such that (1) Z is an *unbiased* estimator for $\mathrm{SJ}(A)$, i.e., $E[Z] = \mathrm{SJ}(A)$; and (2) Z has sufficiently small variance $\mathrm{Var}(Z)$ to provide strong probabilistic guarantees for the quality of the estimate. This random variable Z is constructed on-line from the streaming values of $R.A$ as follows:

- Select a family of *four-wise independent binary random variables* $\{\xi_i : i = 1, \ldots, |\mathtt{dom}(A)|\}$, where each $\xi_i \in \{-1, +1\}$ and $\Pr[\xi_i = +1] = \Pr[\xi_i = -1] = 1/2$ (i.e., $E[\xi_i] = 0$). Informally, the four-wise independence condition means that for any 4-tuple of ξ_i variables and for any 4-tuple of $\{-1, +1\}$ values, the probability that the values of the variables coincide with those in the $\{-1, +1\}$ 4-tuple is exactly $1/16$ (the product of the equality probabilities for each individual ξ_i).
- Define $Z = X^2$, where $X = \sum_{i \in \mathtt{dom}(A)} f(i)\xi_i$. Note that X is simply a randomized linear projection (inner product) of the frequency vector of $R_i.A$ with the vector of ξ_i's that can be efficiently generated from the streaming values of A as follows: Start with $X = 0$ and simply add ξ_i to X whenever the i^{th} value of A is observed in the stream.

The crucial point here is that, by employing known tools (e.g., orthogonal arrays) for the explicit construction of small sample spaces supporting four-wise independent random variables, such families can be efficiently constructed on-line using only $O(\log|\mathtt{dom}(A)|)$ space [4]. More precisely, we do not explicitly store the ξ_i. Instead we store a single *seed* (for ξ_i with i of length k bits, the seed has length $2k+1$ bits) for the whole ξ-family. Whenever a ξ_i is needed for the computation, its value is generated on-the-fly from the seed in time linear in the seed size.

2.3 Boosting Accuracy

To improve the quality of the estimation guarantees one can use a standard *boosting technique* (see [4]) that maintains several independent identically-distributed (i.i.d.) instantiations of a random variable and uses averaging and median-selection operators to boost accuracy and probabilistic confidence. The i.i.d. instances can be constructed by simply selecting independent random seeds for generating the families of four-wise independent ξ_i's for each instance.

Let $\{Z_{i,j}\}$, $i \in \{1, 2, \ldots, k_1\}$ and $j \in \{1, 2, \ldots, k_2\}$, be a set of $k_1 \cdot k_2$ of such i.i.d. instances of an estimator Z for a quantity $\mathcal{Q}$, such that $E[Z] = \mathcal{Q}$. Each $Z_{i,j}$ is a randomized linear projection of the data stream as discussed in Section 2.2. We use the term *atomic sketch* to describe such a single projection, and the term *sketch* for the overall synopsis consisting of $k_1 \cdot k_2$ i.i.d. instances of these random projections.

Figure 1 shows a sketch and illustrates how the boosting works. For each row j we compute the average $\bar{Z}_j = \sum_{i=1}^{k_1} Z_{i,j}$. Then we output the median of these averages $\{\bar{Z}_1, \bar{Z}_2, \ldots, \bar{Z}_{k_2}\}$. The following lemma establishes the quality guarantees.

Lemma 1. *[4] Using* $16\frac{\mathrm{Var}[Z]}{\varepsilon^2 E[Z]^2} \lg\frac{1}{\phi}$ *independent copies of* Z *we can guarantee that the computed estimate* $\bar{Z}$ *of* $E[Z]$ *satisfies* $\Pr[|\bar{Z} - E[Z]| > \varepsilon E[Z]] \leq \phi$.

Proof. By setting $k_1 = \frac{8}{\varepsilon^2}\frac{\mathrm{Var}[Z]}{E[Z]^2}$ and $k_2 = 2\lg(1/\phi)$ the lemma follows from Chebychev's inequality and Chernoff bounds. □

Note that determining the number of instances of Z actually requires knowledge of $E[Z]$, the very value we would like to estimate! This is a common problem shared by previous sketching techniques (see Section 8) and Online Aggregation [19, 18], in fact also by any experiment in the natural sciences where we want to estimate the error in measuring an unknown quantity. We can generally use simple approximation techniques that provide a lower bound on $E[Z]$ ("sanity bounds" as discussed in [3]), or use historic data, e.g., previously computed exact answers, to predict future values of $E[Z]$. The tradeoff here is that the tighter these bounds are, the stronger the quality guarantees returned by the technique.

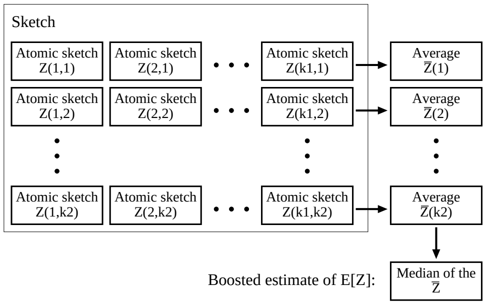

Figure 1: Boosting the accuracy

3. SKETCHES FOR SPATIAL DATA

Techniques designed for relational DBMS often do not easily extend to spatial data. This is mainly caused by the fact that SDBMS manage data with extent. A typical operation during the processing of a spatial query is to check if two objects overlap, or if they are within a certain distance ε of each other. Our techniques address this problem for spatial queries over collections of hyper-rectangles (including points, lines, and rectangles as special cases). Most SDBMS manage hyper-rectangles like geographical maps, environmental measurements for points on the earth's surface, and so on. (Hyper-)rectangles are also used as bounding boxes of more complex objects (e.g., polygons, polylines) since it is often more efficient to first compute a super-set of the final result based on these bounding boxes, and then to filter out the false positives in a final filtering step.

For processing hyper-rectangles, we identified that all of the targeted spatial queries rely on the same basic operation: *determine if a point lies within an interval.* Hence by constructing a corresponding sketch, we have the basic ingredient for selectivity estimation for spatial queries. The actual challenge then is to determine how to use these sketches to obtain a good query result cardinality estimate with high probability, using only small space.

3.1 Basic Sketches for One Dimension

For simplicity assume our input data set R is a one-dimensional set of intervals. We use $[a, b]$ to denote an interval with lower endpoint a and upper endpoint b. To construct our sketch we use a family of four-wise independent random variables $\{\xi_i\}$ (as introduced in Section 2.2), such that variable ξ_i, $i \in \{0, \ldots, n-1\} = \mathcal{N}$, corresponds to coordinate $i \in \mathcal{N}$. An interval $[a, b] \in R$ intuitively is represented by the coordinates of the points it contains, i.e., we use $\xi_a + \xi_{a+1} + \cdots + \xi_b$ to sketch it. Now assume we want to test if a point $c \in \mathcal{N}$ lies within the interval. Observe that since the ξ-variables are four-wise (and hence also pairwise) independent we have

$$\begin{aligned} a \le c \le b &\Leftrightarrow E[(\xi_a + \xi_{a+1} + \cdots + \xi_b)\xi_c] = 1 \\ c < a \vee c > b &\Leftrightarrow E[(\xi_a + \xi_{a+1} + \cdots + \xi_b)\xi_c] = 0 \end{aligned}$$

This forms the basis of our spatial sketches. For data set R define the *standard* atomic spatial sketches

$$\begin{aligned} V_{\mathrm{I}} &= \sum_{[a,b]\in R} (\xi_a + \xi_{a+1} + \cdots + \xi_b) \text{ and} \\ V_{\mathrm{E}} &= \sum_{[a,b]\in R} (\xi_a + \xi_b) \end{aligned} \tag{1}$$

Intuitively V_{I} keeps track of the complete intervals, while V_{E} summarizes only their endpoints.

Notice that the update cost of V_{I} depends linearly on the interval length since we have to add the corresponding ξ_i for each point $i \in \mathcal{N}$ that is contained in the interval. For large coordinate domains, which are not uncommon in spatial applications, this quickly becomes costly and also results in high variance of the estimates. Hence we introduce *dyadic* spatial sketches. Similar to [11, 17], we partition the domain $\mathcal{N}$ into intervals of size 2^i. For simplicity let $n = |\mathcal{N}|$ be a power of 2, say 2^h for some positive integer h.[1] For each *level* $0 \le i \le h$ we partition $\mathcal{N}$ into 2^{h-i} intervals of size 2^i each. Hence for level $i = 0$ we have point "intervals", each corresponding to a single domain value, while for level $i = h$ we have a single interval covering the whole domain. Let D be the set of all dyadic intervals of all levels over $\mathcal{N}$. Our technique is based on the following lemmata.

LEMMA 2. *Let $[a,b]$ be an interval. Its dyadic cover, $D([a,b])$, is defined to be the smallest set of dyadic intervals $\{\delta_1, \delta_2, \ldots, \delta_m\}$ such that $\delta_i = [d_{i-1}, d_i]$, $1 \le i \le m$, and $d_0 = a$ and $d_m = b$. Then $m \le 2\log_2 n$.*

LEMMA 3. *For each point $a \in \mathcal{N}$, let $D([a])$ denote its dyadic point cover, which is the set of all dyadic intervals in D containing a. There are exactly $\log_2 n + 1$ dyadic intervals in D that contain this point. Each of these dyadic intervals is at a different level.*

LEMMA 4. *A point $c \in \mathcal{N}$ is contained in an interval $[a,b]$ iff there is exactly one dyadic interval $\delta \in D$ such that $\delta \in D([a,b])$ and $\delta \in D([c])$.*

PROOF. See [13]. □

Instead of having a ξ-variable for each coordinate in $\mathcal{N}$ we will use a ξ-variable for *each dyadic interval* over $\mathcal{N}$. For data set R, the corresponding atomic sketches for intervals and endpoints then are defined as:

$$\begin{aligned} X_{\mathrm{I}} &= \sum_{[a,b]\in R} \sum_{\delta\in D([a,b])} \xi_\delta \text{ and} \\ X_{\mathrm{E}} &= \sum_{[a,b]\in R} \sum_{\delta\in D([a])\cup D([b])} \xi_\delta \end{aligned} \tag{2}$$

Figure 2 shows an example for intervals and their dyadic covers. To simplify notation we will henceforth use

$$\bar{\xi}_{[a,b]} = \sum_{\delta\in D([a,b])} \xi_\delta \text{ and } \bar{\xi}_{[a]} = \sum_{\delta\in D([a])} \xi_\delta \tag{3}$$

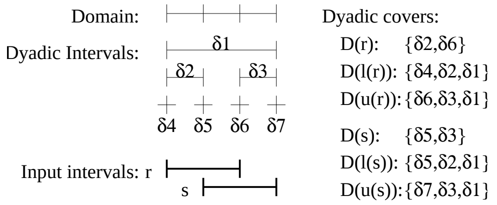

Figure 2: Intervals and their dyadic covers

With this notation we can rewrite the dyadic atomic sketches as

$$X_{\mathrm{I}} = \sum_{[a,b]\in R} \bar{\xi}_{[a,b]}; \qquad X_{\mathrm{E}} = \sum_{[a,b]\in R} (\bar{\xi}_{[a]} + \bar{\xi}_{[b]}) \tag{4}$$

Overall, using dyadic sketches the update cost is reduced to $\mathrm{O}(\log_2 n)$ per atomic sketch. Since the atomic sketches are defined over the set of dyadic intervals, which has cardinality $2n - 1$, we need $\mathrm{O}(\log_2(2n-1)) = \mathrm{O}(\log_2 n)$ space for generating each family of the ξ-variables.

For variance analysis it will often be more convenient to rewrite the dyadic sketch definition into a different format. Note that X_{I} is computed by adding one or more ξ_{δ_i} for each interval $[a,b] \in R$, where $\delta_i \in D$ is a dyadic interval over $\mathcal{N}$. For each dyadic interval $\delta \in D$ let $f_I(\delta) = |\{[a,b] \in R | \delta \in D([a,b])\}|$ be the number of intervals in R whose cover contains δ. Similarly, let $f_E(\delta) = |\{a | a \in \delta \wedge (\exists b \in \mathcal{N} : [a,b] \in R \vee [b,a] \in R)\}|$ be the number of interval endpoints in R whose cover contains dyadic interval δ. Then we can write

$$X_{\mathrm{I}} = \sum_{\delta\in D} f_I(\delta)\xi_\delta; \qquad X_{\mathrm{E}} = \sum_{\delta\in D} f_E(\delta)\xi_\delta \tag{5}$$

For instance for interval r in Figure 2 we have $f_I(\delta_2) = 1$, $f_I(\delta_6) = 1$, and $f_I(\delta_i) = 0$ for $i \in \{1,3,4,5,7\}$.

For an atomic sketch X we define its *self-join size* $\mathrm{SJ}(X)$ as $E[X^2]$. For example, for sketch X_{I} we have

$$\mathrm{SJ}(X_{\mathrm{I}}) = E[X_{\mathrm{I}}^2] = E[(\sum_{\delta\in D} f_I(\delta)\xi_\delta)^2] = \sum_{\delta\in D} f_I^2(\delta) \ .$$

The last step follows from linearity of expectation and the four-wise (and hence two-wise) independence of the ξ-variables.

3.2 Sketches for Multidimensional Data

We discuss sketches for $d = 2$, i.e., sets of rectangles. The generalization to higher dimensionality then is straightforward. As introduced in Section 2.1, we write a rectangle as the cross-product of its ranges in each dimension, e.g., $r = [a,b] \times [c,d]$.

To keep track of two-dimensional objects, we need an independent $\xi(i)$ family of four-wise independent random variables (see Section 2.2) for each dimension $i \in \{1,2\}$. More precisely, any $\xi(1)_j$ is independent of any $\xi(2)_k$ for any j, k. The main observation for sketching a rectangle $[a,b] \times [c,d]$ is that we (1) have to make sure that the sketch "remembers" that $[a,b]$ and $[c,d]$ "belong together", and that (2) we have to keep track not only of rectangles and points, but also of horizontal and vertical lines, as will become clear below. The latter is important for correct join size estimation.

The dyadic atomic sketches for a two-dimensional data set

[1] Otherwise we just pad the domain with additional values.

R are defined as follows:

$$
\begin{aligned}
X_{\mathrm{II}} &= \sum_{[a,b]\times[c,d]\in R} \bar{\xi}(1)_{[a,b]}\bar{\xi}(2)_{[c,d]} \\
X_{\mathrm{IE}} &= \sum_{[a,b]\times[c,d]\in R} \bar{\xi}(1)_{[a,b]}(\bar{\xi}(2)_{[c]}+\bar{\xi}(2)_{[d]}) \\
X_{\mathrm{EI}} &= \sum_{[a,b]\times[c,d]\in R} (\bar{\xi}(1)_{[a]}+\bar{\xi}(1)_{[b]})\bar{\xi}(2)_{[c,d]} \\
X_{\mathrm{EE}} &= \sum_{[a,b]\times[c,d]\in R} (\bar{\xi}(1)_{[a]}+\bar{\xi}(1)_{[b]})(\bar{\xi}(2)_{[c]}+\bar{\xi}(2)_{[d]})
\end{aligned}
$$

The $\bar{\xi}$ are sums over the ξ-variables for dyadic intervals that cover the corresponding interval or endpoint as defined in the previous section. Intuitively, X_{II} keeps track of the whole rectangles, X_{IE} and X_{EI} of their horizontal and vertical edges, respectively, and X_{EE} of the rectangles' corner points. The standard atomic sketches V_{II}, V_{IE}, V_{EI}, and V_{EE} are defined similarly.

Similar to Equation 5, we can rewrite X_{II} as

$$X_{\mathrm{II}} = \sum_{\delta(1)\times\delta(2)\in D^2} f_{\mathrm{II}}(\delta(1),\delta(2))\xi(1)_{\delta(1)}\xi(2)_{\delta(2)} \qquad (6)$$

Here variable $f_{\mathrm{II}}(\delta(1),\delta(2))$ denotes the number of rectangles in R whose dyadic cover contains rectangle $\delta(1)\times\delta(2)$, defined by dyadic intervals $\delta(1)$ and $\delta(2)$ in dimension 1 and 2, respectively. We can similarly rewrite the other atomic sketches. As in Section 3.1 we define $\mathrm{SJ}(X) = E[X^2]$, hence we obtain in a similar manner

$$\mathrm{SJ}(X_{\mathrm{II}}) = \sum_{\delta(1)\times\delta(2)\in D^2} f_{\mathrm{II}}^2(\delta(1),\delta(2))$$

The self-join size for the other atomic sketches is similar.

The generalization to higher dimensionality follows the same pattern. Let $\mathrm{IE}^d = \{\mathrm{I},\mathrm{E}\}^d$ be the set of all strings of length d over letters I and E. For $w \in \mathrm{IE}^d$ the term $w[i]$ refers to the i-th letter in w. For a hyper-rectangle r let $r(i)$ be its range in dimension i. With $\mathcal{N}^d$ we denote the data space. The atomic sketches for R are then defined for all $w \in \mathrm{IE}^d$ as:

$$X_w = \sum_{r(1)\times r(2)\times\cdots\times r(d)\in R} \xi(1)_{r(1)}\xi(2)_{r(2)}\cdots\xi(d)_{r(d)}$$

such that for each dimension i, $\xi(i)_{r(i)} = \bar{\xi}(i)_{[l(r(i)),u(r(i))]}$ if $w[i] = \mathrm{I}$, and $\xi(i)_{r(i)} = \bar{\xi}(i)_{[l(r(i))]} + \bar{\xi}(i)_{[u(r(i))]}$ otherwise (i.e., if $w[i] = \mathrm{E}$). As before, each family $\xi(i)$ for dimension i is a family of four-wise independent $\{-1,1\}$ random variables. For any $i \neq j$ the variables in $\xi(i)$ are independent from the variables in $\xi(j)$.

4. SPATIAL JOINS

4.1 Spatial Join of Intervals

We describe a counting procedure that will be used to compute the cardinality of the join of two sets of intervals. Since intervals only overlap if their intersection is a non-empty one-dimensional object (see Definition 1), we can safely assume that the data sets do not contain any degenerate objects, in this case point objects, since these would not contribute to the join result anyway. We also assume that the data domain is finite. We will discuss in Section 5.1 how to handle real valued coordinates.

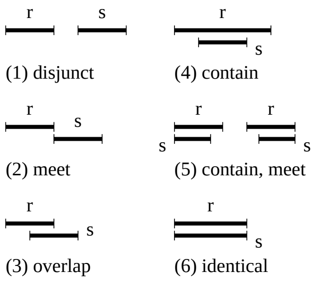

Figure 3: Spatial relationships between intervals

4.1.1 Spatial Relationships

Figure 3 shows all cases of spatial relationships between an interval in R and a potential join partner in S (cases which can be obtained by swapping r and s are omitted for simplicity). According to our definition, in cases (1) and (2) the intervals do not overlap, while intervals in cases (3)-(6) are overlapping. If two intervals $r \in R$ and $s \in S$ fall into case (i), we say that they have the *spatial relationship* (i).

4.1.2 Counting Intersections

Finding an appropriate sketch that correctly counts all overlaps, but does not count cases (1) and (2) is a non-trivial problem. None of the previously proposed approaches (cf. Section 8) can be directly applied to compute a function like "add 1 for each pair of intervals (r,s) for which one endpoint of r is between the two endpoints of s". In order to use sketches, we need a different way of counting interval intersections. As before, let $l(r)$ and $u(r)$ be the lower and upper endpoints of interval r, respectively (similar for s).

Intuitively, for each pair of intervals $r \in R$ and $s \in S$ we want to add 0 to the join result size if they have spatial relationship (1) or (2), and 1 otherwise. We obtain a first approximation by counting for each pair (r,s) how many of their endpoints are contained in the other interval. For instance, in case (3) $r.u$ is covered by s, and $s.l$ is covered by r, hence the count is 2. Overall we obtain counts 0, 2, 2, 2, 3, and 4 for cases (1) to (6), respectively. If we divide this result by 2, we obtain counts 0, 1, 1, 1, 1.5, 2. However, for correct join selectivity estimation the counts should be 0, 0, 1, 1, 1, 1 (add 1 only for cases (3) to (6)).

Notice that the "problem cases" with incorrect counts are only those cases where r and s have endpoints in common. Hence for the remainder of this section we will assume the following.

ASSUMPTION 1. *None of the intervals in R has an endpoint in common with any of the intervals in S.*

With the assumption it is easy to see that cases (2), (5), and (6) are eliminated, and since our technique counts correctly for cases (1), (3), and (4), we have a simple method for calculating the join size. We will show in Section 5.2 how to generalize our algorithm if the assumption does not hold.

4.1.3 An Atomic Sketch for Intervals

The simple counting procedure can be directly implemented with interval and endpoint sketches as introduced in Section 3.1. Here we use the dyadic sketches, i.e., we construct atomic sketches X_{I} and X_{E} for R, and the corresponding sketches Y_{I} and Y_{E} for S.

We want to count how many times an endpoint of an interval in S is contained in an interval in R, and vice versa. To do this we just multiply the corresponding sketches, i.e., we define a random variable $Z = (X_{\mathrm{I}}Y_{\mathrm{E}} + X_{\mathrm{E}}Y_{\mathrm{I}})/2$. This random variable is an unbiased estimator for the join cardinality.

LEMMA 5. *Random variable Z has the expected value $E[Z] = |R \bowtie_o S|$.*

PROOF. See [13]. □

For our example in Figure 2 we have $X_{\mathrm{I}} = \xi_2 + \xi_6$, $X_{\mathrm{E}} = 2\xi_1 + \xi_2 + \xi_3 + \xi_4 + \xi_6$, $Y_{\mathrm{I}} = \xi_3 + \xi_5$, and $Y_{\mathrm{E}} = 2\xi_1 + \xi_2 + \xi_3 + \xi_5 + \xi_7$. For Z we therefore obtain

$$\begin{aligned} Z &= ((\xi_2 + \xi_6)(2\xi_1 + \xi_2 + \xi_3 + \xi_5 + \xi_7) \\ &\quad +(2\xi_1 + \xi_2 + \xi_3 + \xi_4 + \xi_6)(\xi_3 + \xi_5))/2 \ . \end{aligned}$$

Multiplying these terms results in a formula that is the summation of terms of the form $\xi_i \cdot \xi_j$. Recall that the ξ-variables are four-wise (and hence also pairwise) independent, therefore it holds that $E[\xi_i\xi_j] = E[\xi_i]E[\xi_j] = 0$ if $i \neq j$. Furthermore, because each ξ is either 1 or -1, it holds that $E[\xi_i^2] = 1$. Thus from the above formula we obtain

$$E[Z] = (E[\xi_2^2] + E[\xi_3^2])/2 = 1 \ ,$$

which, as expected, is the correct value for the number of intersecting intervals.

4.1.4 Variance Analysis

Using standard transformations we obtain:

$$\begin{aligned} \mathrm{Var}[Z] &= \mathrm{Var}[(X_{\mathrm{I}}Y_{\mathrm{E}} + X_{\mathrm{E}}Y_{\mathrm{I}})/2] \\ &= 1/4(\mathrm{Var}[X_{\mathrm{I}}Y_{\mathrm{E}}] + \mathrm{Var}[X_{\mathrm{E}}Y_{\mathrm{I}}] \\ &\quad +2\mathrm{Cov}[X_{\mathrm{I}}Y_{\mathrm{E}}, X_{\mathrm{E}}Y_{\mathrm{I}}]) \end{aligned}$$

Since in general, for any random variables X and Y, $|\mathrm{Cov}[X,Y]| \leq \sqrt{\mathrm{Var}[X]}\sqrt{\mathrm{Var}[Y]}$:

$$\begin{aligned} \mathrm{Var}[Z] &\leq 1/4(\mathrm{Var}[X_{\mathrm{I}}Y_{\mathrm{E}}] + \mathrm{Var}[X_{\mathrm{E}}Y_{\mathrm{I}}] \\ &\quad +2\sqrt{\mathrm{Var}[X_{\mathrm{I}}Y_{\mathrm{E}}]}\sqrt{\mathrm{Var}[X_{\mathrm{E}}Y_{\mathrm{I}}]}) \\ &= 1/4(\sqrt{\mathrm{Var}[X_{\mathrm{I}}Y_{\mathrm{E}}]} + \sqrt{\mathrm{Var}[X_{\mathrm{E}}Y_{\mathrm{I}}]})^2 \qquad (7) \end{aligned}$$

To analyze the terms of Equation 7 we use the definition of the atomic sketches in the form of Equation 5. For this type of sketches (tug-of-war sketch) it can be shown that their variance is bounded by twice the product of their self-join size [3]:

$$\begin{aligned} \mathrm{Var}[X_{\mathrm{I}}Y_{\mathrm{E}}] &\leq 2\mathrm{SJ}(X_{\mathrm{I}})\mathrm{SJ}(Y_{\mathrm{E}}) \text{ and} \\ \mathrm{Var}[X_{\mathrm{E}}Y_{\mathrm{I}}] &\leq 2\mathrm{SJ}(X_{\mathrm{E}})\mathrm{SJ}(Y_{\mathrm{I}}) \ . \end{aligned}$$

Together with Equation 7 we have

$$\mathrm{Var}[Z] \leq 1/2(\sqrt{\mathrm{SJ}(X_{\mathrm{I}})\mathrm{SJ}(Y_{\mathrm{E}})} + \sqrt{\mathrm{SJ}(X_{\mathrm{E}})\mathrm{SJ}(Y_{\mathrm{I}})})^2 \ .$$

Since X_{I} and X_{E} together account for all dyadic intervals that cover the intervals of R and their endpoints (similarly the Y sketches for S), we define

$$\mathrm{SJ}(R) = \mathrm{SJ}(X_{\mathrm{I}}) + \mathrm{SJ}(X_{\mathrm{E}}) \text{ and } \mathrm{SJ}(S) = \mathrm{SJ}(Y_{\mathrm{I}}) + \mathrm{SJ}(Y_{\mathrm{E}}) \,.$$

Using the Cauchy-Schwarz inequality it holds that $(\sqrt{\mathrm{SJ}(X_{\mathrm{I}})}\sqrt{\mathrm{SJ}(Y_{\mathrm{E}})} + \sqrt{\mathrm{SJ}(X_{\mathrm{E}})}\sqrt{\mathrm{SJ}(Y_{\mathrm{I}})})^2 \leq (\mathrm{SJ}(X_{\mathrm{I}}) + \mathrm{SJ}(X_{\mathrm{E}}))(\mathrm{SJ}(Y_{\mathrm{E}}) + \mathrm{SJ}(Y_{\mathrm{I}}))$, hence:

$$\mathrm{Var}[Z] \leq 1/2\,\mathrm{SJ}(R)\mathrm{SJ}(S) \ . \qquad (8)$$

4.1.5 The Overall Technique

Having constructed the appropriate random variable Z and computed an upper bound on its variance, we can boost its accuracy as described in Section 2.3. The following theorem summarizes the result.

THEOREM 1. *Let $R = \{r_1, r_2, \ldots, r_{|R|}\}$ and $S = \{s_1, s_2, \ldots, s_{|S|}\}$ be two sets of intervals that satisfy Assumption 1. Furthermore let $X_{\mathrm{I}}^{(i)} = \sum_{[a,b]\in R} \bar{\xi}_{[a,b]}^{(i)}$, $X_{\mathrm{E}}^{(i)} = \sum_{[a,b]\in R}(\bar{\xi}_{[a]}^{(i)} + \bar{\xi}_{[b]}^{(i)})$, $Y_{\mathrm{I}}^{(i)} = \sum_{[c,d]\in S} \bar{\xi}_{[c,d]}^{(i)}$, and $Y_{\mathrm{E}}^{(i)} = \sum_{[c,d]\in S}(\bar{\xi}_{[c]}^{(i)} + \bar{\xi}_{[d]}^{(i)})$, $1 \leq i \leq 8\frac{\mathrm{SJ}(R)\mathrm{SJ}(S)}{\varepsilon^2 E[Z]^2} \lg\frac{1}{\phi}$, be atomic sketches such that $Z_i = (X_{\mathrm{I}}^{(i)}Y_{\mathrm{E}}^{(i)} + X_{\mathrm{E}}^{(i)}Y_{\mathrm{I}}^{(i)})/2$. The $\bar{\xi}$-variables are defined as in Equation 3 over $\xi^{(i)}$-families of four-wise independent $\{-1, 1\}$ random variables, which are generated from seeds $s^{(i)}$ of size $\mathrm{O}(\log_2 n)$ bits and where each $s^{(i)}$ is independently chosen. By computing the median of $2\lg(1/\phi)$ averages over groups of $4\frac{\mathrm{SJ}(R)\mathrm{SJ}(S)}{\varepsilon^2 E[Z]^2}$ estimators Z_i we obtain an estimate $\bar{Z}$ that with probability $1-\phi$ is within ε relative error of the true expected value $E[Z] = |R \bowtie_o S|$.*

Notice that the atomic sketch Z for estimating the join size only stores five values: a seed of size $\mathrm{O}(\log n)$ bits for generating the ξ-variables on-the-fly (during updates), and four counters for keeping track of the current value of X_{I}, X_{E}, Y_{I}, and Y_{E}. As mentioned before, when an interval is inserted into R (deleted from R), we simply generate the corresponding ξ-variables for its interval and endpoint cover from the seed and add (subtract) them from X_{I} and X_{E}, respectively (similarly for S).

The cost of generating a ξ-variable from the seed is linear in the seed size, and each interval or endpoint cover consists of $\mathrm{O}(\log n)$ dyadic intervals. Hence the total update cost for a single instance of Z is $\mathrm{O}(\log^2 n)$. To compute Z's estimate we simply combine the counters, resulting in a constant overhead. Since we use $8\frac{\mathrm{SJ}(R)\mathrm{SJ}(S)}{\varepsilon^2 E[Z]^2} \lg\frac{1}{\phi}$ independent instances of Z, the total query, update, and storage costs of our interval join sketch are $\mathrm{O}(\frac{\mathrm{SJ}(R)\mathrm{SJ}(S)}{\varepsilon^2 E[Z]^2} \log\frac{1}{\phi})$, $\mathrm{O}(\frac{\mathrm{SJ}(R)\mathrm{SJ}(S)}{\varepsilon^2 E[Z]^2} \log\frac{1}{\phi} \log^2 n)$, and $\mathrm{O}(\frac{\mathrm{SJ}(R)\mathrm{SJ}(S)}{\varepsilon^2 E[Z]^2} \log\frac{1}{\phi} \log n)$, respectively.

4.2 Spatial Join of Rectangles

The spatial relationships between intervals (cf. Section 4.1.1) generalize naturally to higher dimensionality by examining the one-dimensional axis-parallel projections of the hyper-rectangles (these projections are intervals). Figure 4 shows selected examples.[2] The spatial relationship of two hyper-rectangles r and s is a d-tuple $(i_1, \ldots, i_d)$ where i_j is the spatial relationship of the projection in dimension j, as per Figure 3. Hence r and s overlap iff $i_j \in \{3, 4, 5, 6\}$ for *all* dimensions j.

4.2.1 Rectangle Sketches

The simple counting procedure for interval overlaps generalizes to *two-dimensional* rectangles as follows. Let r and s be rectangles from R and S, respectively. The counting procedure adds the following values: number of corners of

[2] The spatial relationships are slightly different from the ones used in [24, 25] since we selected them to correspond to the sketches we use.

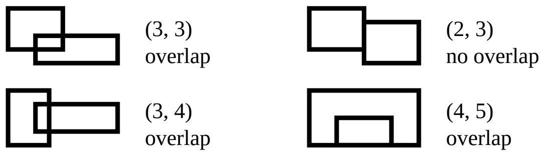

Figure 4: Spatial relationships between rectangles

r which are covered by s, number of horizontal edges of r which intersect vertical edges of s, number of vertical edges of r which intersect horizontal edges of s, and number of corners of s which are covered by r. The reader might easily convince herself that under the assumption that r and s have no common corner coordinates in any dimension, we will always obtain a count of 4 for intersecting rectangles (and 0 otherwise).

As it turns out, we can combine the two-dimensional sketches (see Section 3.2) to obtain the correct count; using atomic sketches X_{II}, X_{IE}, X_{EI}, and X_{EE} for R, and the analogously defined Y_{II}, Y_{IE}, Y_{EI}, and Y_{EE} for S:

LEMMA 6. *Let Z be a random variable such that $Z = (X_{\mathrm{II}}Y_{\mathrm{EE}} + X_{\mathrm{IE}}Y_{\mathrm{EI}} + X_{\mathrm{EI}}Y_{\mathrm{IE}} + X_{\mathrm{EE}}Y_{\mathrm{II}})/4$; and R and S be two sets of rectangles that satisfy Assumption 1 in each dimension. Then $E[Z] = |R \bowtie_o S|$ and $\mathrm{Var}[Z] \leq 1/16(8\mathrm{SJ}(R)\mathrm{SJ}(S)) = 1/2\,\mathrm{SJ}(R)\mathrm{SJ}(S)$.*

PROOF. It is easy to show that $E[Z] = |R \bowtie_o S|$, with the analysis being similar to the 1-dimensional case. The variance analysis, on the other hand, differs from the 1-dimensional case mainly because the set of random variables $\{\xi(1)_{\delta(1)}\xi(2)_{\delta(2)} : \delta(1) \times \delta(2) \in D^2\}$ does *not* have the 4-wise independence property. This might come as a surprise since the set of $\xi(1)$ and the set of $\xi(2)$-variables are both 4-wise independent, and the variables in one set are independent of those in the other. However, notice that for instance for the 4-tuple $\{x_1 = \xi(1)_{\delta(1)_1}\xi(2)_{\delta(2)_1}, x_2 = \xi(1)_{\delta(1)_1}\xi(2)_{\delta(2)_2}, x_3 = \xi(1)_{\delta(1)_2}\xi(2)_{\delta(2)_1}, x_4 = \xi(1)_{\delta(1)_2}\xi(2)_{\delta(2)_2}\}$ it is easy to show that $E[x_1x_2x_3x_4] = 1 \neq 0 = E[x_1]E[x_2]E[x_3]E[x_4]$. Hence we cannot directly use variance bounds from earlier results.

We can obtain a bound on the variance of Z as follows. In the following, for $w \in \mathrm{IE}^2$ let $\bar{w}$ be the string obtained from w by replacing I with E and vice versa. Then

$$\begin{aligned}\mathrm{Var}[Z] &= \frac{1}{16}\Big(\sum_{w\in \mathrm{IE}^2} \mathrm{Var}[X_w Y_{\bar{w}}] \\ &\qquad + \sum_{w_1 \neq w_2} \mathrm{Cov}[X_{w_1}Y_{\bar{w_1}}, X_{w_2}Y_{\bar{w_2}}]\Big) \\ &\leq \frac{1}{16}\Big(\sum_{w\in \mathrm{IE}^2} \sqrt{\mathrm{Var}[X_w Y_{\bar{w}}]}\Big)^2\end{aligned}$$

The inequality follows since, for any random variables X and Y, $|\mathrm{Cov}[X, Y]| \leq \sqrt{\mathrm{Var}[X]}\sqrt{\mathrm{Var}[Y]}$.

With some involved analysis we can show that for any $w \in \mathrm{IE}^2$, $\mathrm{Var}[X_w Y_{\bar{w}}] \leq 8\mathrm{SJ}(X_w)\mathrm{SJ}(Y_{\bar{w}})$. With the above inequality we have:

$$\mathrm{Var}[Z] \leq 1/16 \cdot 8 \cdot \Big(\sum_{w\in IE^2} \sqrt{\mathrm{SJ}(X_w)\mathrm{SJ}(Y_{\bar{w}})}\Big)^2$$

Using Cauchy-Schwarz and the fact that $\mathrm{SJ}(R) = \sum_{w\in \mathrm{IE}^2} \mathrm{SJ}(X_w)$ and $\mathrm{SJ}(S) = \sum_{w\in \mathrm{IE}^2} \mathrm{SJ}(Y_w)$ yields:

$$\mathrm{Var}[Z] \leq 1/16\,(8\,\mathrm{SJ}(R)\mathrm{SJ}(S))\ .$$

For details see [13]. □

Here $\mathrm{SJ}(R) = \mathrm{SJ}(X_{\mathrm{II}}) + \mathrm{SJ}(X_{\mathrm{IE}}) + \mathrm{SJ}(X_{\mathrm{EI}}) + \mathrm{SJ}(X_{\mathrm{EE}})$, similarly for $\mathrm{SJ}(S)$.

4.2.2 The Overall Technique

As for interval sketches, we boost the accuracy of the atomic rectangle sketch as described in Section 2.3 to obtain accurate estimates of $|R \bowtie_o S|$ with quality guarantees.

THEOREM 2. *Let $R = \{r_1, r_2, \ldots, r_{|R|}\}$ and $S = \{s_1, s_2, \ldots, s_{|S|}\}$ be two sets of rectangles whose endpoints have no coordinates in common in any dimension. Furthermore let $X_{\mathrm{II}} = \sum_{[a,b]\times[c,d]\in R} \bar{\xi}(1)_{[a,b]}\bar{\xi}(2)_{[c,d]}$, $X_{\mathrm{IE}} = \sum_{[a,b]\times[c,d]\in R} \bar{\xi}(1)_{[a,b]}(\bar{\xi}(2)_{[c]} + \bar{\xi}(2)_{[d]})$, $X_{\mathrm{EI}} = \sum_{[a,b]\times[c,d]\in R}(\bar{\xi}(1)_{[a]} + \bar{\xi}(1)_{[b]})\bar{\xi}(2)_{[c,d]}$, $X_{\mathrm{EE}} = \sum_{[a,b]\times[c,d]\in R}(\bar{\xi}(1)_{[a]} + \bar{\xi}(1)_{[b]})(\bar{\xi}(2)_{[c]} + \bar{\xi}(2)_{[d]})$, $Y_{\mathrm{II}} = \sum_{[a,b]\times[c,d]\in S} \bar{\xi}(1)_{[a,b]}\bar{\xi}(2)_{[c,d]}$, $Y_{\mathrm{IE}} = \sum_{[a,b]\times[c,d]\in S} \bar{\xi}(1)_{[a,b]}(\bar{\xi}(2)_{[c]} + \bar{\xi}(2)_{[d]})$, $Y_{\mathrm{EI}} = \sum_{[a,b]\times[c,d]\in S}(\bar{\xi}(1)_{[a]} + \bar{\xi}(1)_{[b]})\bar{\xi}(2)_{[c,d]}$, and $Y_{\mathrm{EE}} = \sum_{[a,b]\times[c,d]\in S}(\bar{\xi}(1)_{[a]} + \bar{\xi}(1)_{[b]})(\bar{\xi}(2)_{[c]} + \bar{\xi}(2)_{[d]})$, be atomic sketches such that $Z = (X_{\mathrm{II}}Y_{\mathrm{EE}} + X_{\mathrm{IE}}Y_{\mathrm{EI}} + X_{\mathrm{EI}}Y_{\mathrm{IE}} + X_{\mathrm{EE}}Y_{\mathrm{II}})/4$. Using $8\frac{\mathrm{SJ}(R)\mathrm{SJ}(S)}{\varepsilon^2 E[Z]^2} \lg \frac{1}{\phi}$ independent instances of Z we obtain an estimate $\bar{Z}$ that with probability $1 - \phi$ is within ε relative error of the true expected value $E[Z] = |R \bowtie_o S|$.*

Note that the number of instances of Z for the one- and two-dimensional case incidentally is the same (cf. Theorem 1). However, the actual storage cost can be quite different. First, the number of atomic sketches per instance of Z has doubled for $d = 2$. Second, the self-join size for R and S will be larger than for the one-dimensional case. Since update and query cost depend approximately linearly on the storage, those costs will be larger as well. In Section 6.1 we will show that our technique suffers from the "curse of dimensionality", like any other estimation or indexing technique.

5. GENERALIZATION

In this section we discuss how to remove the restricting assumptions we made for our technique.

5.1 Real-Valued Data Domains

Note that for our techniques to work, domain $\mathcal{N}$ should be finite. This is essential because a seed of size $2k+1$ limits us to a domain of size 2^k. Theoretically this prevents us from using our technique for real-valued domains.

However, in practice our technique will work just fine. There is no spatial application we know of that uses coordinates of unbounded precision. Typically real-valued coordinates are stored as 32 or 64 bit size floating point numbers—clearly a finite domain.

Our sketch-based estimation technique can handle such domains very well. Recall that the storage requirement of an atomic sketch is logarithmic in the domain size. Hence our technique scales very well with increasing domain size. In contrast, histograms traditionally suffer from the problem that large data domains result in large buckets and hence poor approximation. Histogram-based techniques therefore typically quantize the data space and make assumptions about the distribution within a bucket in order to obtain good estimates [23, 26].

5.2 Spatial Join with Common Endpoints

The counting algorithms used so far relied on Assumption 1 for correctness. We can make this assumption hold for any data set as follows. Recall that the interval endpoints are drawn from domain $\mathcal{N} = \{0, 1, \dots, n-1\}$. First, we create a new domain $\mathcal{M}$ which contains all values from $\mathcal{N}$ and in addition for each pair of consecutive values $i, i+1 \in \mathcal{N}$ two values i_+ and $(i+1)_-$ with $i < i_+ < (i+1)_- < (i+1)$. Then we replace each interval $s \in S$ by a new interval s' such that for $l(s) = i$, we set $l(s') = i_+$ and for $u(s) = j$ we set $u(s') = j_-$, i.e., we shrink the interval "a little". For instance if all interval endpoints are integer coordinates, we could augment the domain by $i - 0.1$ and $i + 0.1$ for each integer in the domain and thus shrink the S-intervals by 0.1 at each end. The spatial join size is not affected by this transformation: For each case we can easily verify $\texttt{overlap}(r, s) \Leftrightarrow \texttt{overlap}(r, s')$. All this transformation does is increase the dimension's domain size by at most a factor of 3. The asymptotic cost of our technique is not affected by this transformation (cf. Section 4.1.5 where n now becomes $3n$). We can apply the same transformation to each dimension of the data space.

Instead of using the endpoint transformation, we can alternatively adapt our original simple counting procedure to explicitly keep track of common endpoints [13].

6. EXTENSIONS

6.1 Join of Hyper-Rectangles

The one-and two-dimensional spatial join estimators generalize naturally to higher dimensionality d as the following theorem shows.

THEOREM 3. *For d-dimensional data sets R and S let* $\{X_w\}_{w \in \mathrm{IE}^d}$ *and* $\{Y_w\}_{w \in \mathrm{IE}^d}$ *be two families of atomic sketches over the same* $\xi(i)$ *families of four-wise independent random variables, such that families* $\xi(i)$ *and* $\xi(j)$, $1 \leq i, j \leq d$, *for* $i \neq j$ *are independent of each other. Set* $Z = 2^{-d} \sum_{w \in \mathrm{IE}^d} X_w Y_{\bar{w}}$. *Then it holds that* $E[Z] = |R \bowtie_o S|$ *and* $\mathrm{Var}[Z] \leq \frac{3^d - 1}{4^d} \mathrm{SJ}(R)\mathrm{SJ}(S)$.

PROOF. The proof is fairly involved and can be found in [13]. □

The atomic sketches are as defined in Section 3.2. As before, for $w \in \mathrm{IE}^d$ we define $\bar{w}$ as the string that is obtained from w by replacing I with E and vice versa (the "complement" of w).

At a first glance the variance might appear to be decreasing with increasing dimensionality d due to the $(3/4)^d$ factor. Unfortunately this is not the case. The self-join terms for R and S each have 2^d contributing sums. Also note that to compute Z we have to maintain 2^d atomic sketches for each stream.

6.2 Other Join Conditions

We can extend our algorithm to estimate the cardinality for a *slightly different notion of overlap*, where case (2) (cf. Figure 3) is also counted as overlap. We can also support other join predicates like *containment joins* by maintaining the appropriate spatial sketches. Due to space constraints we defer the details to [13].

6.3 ε-Joins

For simplicity we will first discuss ε-joins for two-dimensional data. Let A and B be two-dimensional sets of data points over space $\mathcal{N}^2$. In the following we assume that the L_∞ distance is used. We can estimate the ε-join cardinality based on the following observation. Let B' be a data set obtained from B by replacing each point $b \in B$ with the spatial object b' which is defined as $b' = \{x | x \in \mathcal{N}^2 \wedge \texttt{dist}_\infty(x, b) \leq \varepsilon\}$. Notice that for the L_∞ distance this object b' is a square of sidelength 2ε with b as its center.

According to definition, it holds that $\texttt{dist}_\infty(a, b) \leq \varepsilon \Leftrightarrow$ a is contained in b'. Hence we can compute the cardinality of the ε-join by counting how many points of A are contained in the squares of B'. This turns out to be a special case of the rectangle join where endpoints might have equal coordinates. Since the objects in A are points, we do not need most of the two-dimensional atomic sketches, except for:

$$\begin{aligned} X_{\mathrm{EE}} &= \sum_{(a_1, a_2) \in A} \bar{\xi}(1)_{[a_1]} \bar{\xi}(2)_{[a_2]} \\ Y_{\mathrm{II}} &= \sum_{[e,f] \times [g,h] \in B'} \bar{\xi}(1)_{[e,f]} \bar{\xi}(2)_{[g,h]} \ , \end{aligned}$$

where the $\bar{\xi}$-variables are as defined in Equation 3.

LEMMA 7. *For random variable* $Z = X_{\mathrm{EE}} Y_{\mathrm{II}}$ *it holds that* $E[Z] = |A \bowtie_\varepsilon B|$ *and* $\mathrm{Var}[Z] \leq 8\mathrm{SJ}(X_{\mathrm{EE}})\mathrm{SJ}(Y_{\mathrm{II}})$.

The generalization to any dimensionality follows the same pattern as for the spatial join. The following lemma summarizes the result:

LEMMA 8. *For d-dimensional point sets A and B let* $B' = \{b' | b' = \{x | x \in \mathcal{N}^d \wedge \texttt{dist}_\infty(x, b) \leq \varepsilon\}\}$ *be the set of hyper-cubes of sidelength* 2ε *around the points of B. For atomic sketches* $X_{\mathrm{E}} = \sum_{(a_1, a_2, \dots, a_d) \in A} \bar{\xi}(1)_{[a_1]} \bar{\xi}(2)_{[a_2]} \cdots \bar{\xi}(d)_{[a_d]}$ *and* $Y_{\mathrm{I}} = \sum_{l_1 \times l_2 \times \cdots \times l_d \in B'} \bar{\xi}(1)_{[l(l_1), u(l_1)]} \bar{\xi}(2)_{[l(l_2), u(l_2)]} \cdots \bar{\xi}(d)_{[l(l_d), u(l_d)]}$ *let* $Z = X_{\mathrm{E}} Y_{\mathrm{I}}$. *Then* $E[Z] = |A \bowtie_\varepsilon B|$ *and* $\mathrm{Var}[Z] \leq (3^d - 1)\mathrm{SJ}(X_{\mathrm{E}})\mathrm{SJ}(Y_{\mathrm{I}})$.

Here $\mathrm{SJ}(X_{\mathrm{E}}) = \sum_{\delta(1) \times \delta(2) \times \cdots \times \delta(d) \in D^d} f^2(\delta(1), \delta(2), \dots, \delta(d))$ and $\mathrm{SJ}(Y_{\mathrm{I}}) = \sum_{\delta(1) \times \delta(2) \times \cdots \times \delta(d) \in D^d} g^2(\delta(1), \delta(2), \dots, \delta(d))$ where $f(\delta(1), \delta(2), \dots, \delta(d))$ is the number of points in A which are covered by the hyper-rectangle spanned by the dyadic intervals $\delta(i)$. Similarly, $g(\delta(1), \delta(2), \dots, \delta(d))$ is the number of hyper-cubes in B' whose dyadic cover contains $\delta(1) \times \delta(2) \times \cdots \times \delta(d)$.

Notice that the constant factor of the variance bound seems surprisingly high, compared to the bound for the more general spatial join (see Theorem 3). However, here the self-join sizes will be much lower (1 term versus 2^d terms added for the spatial join), and the number of atomic sketches is much lower (2 per instance of Z versus $2 \cdot 2^d$ for the spatial join).

6.4 Range Queries

The range query is a special case of the spatial join, where the second "data set" consists only of a single hyper-rectangle—the query (see Definition 3). Hence we could readily apply the spatial join estimation technique. The following optimization can further improve performance. We illustrate it for the case of R being a one-dimensional set of intervals.

Assume we want to estimate how many intervals are selected by range query $q = [u, v]$. The reader might convince herself that an interval $[a, b] \in R$ is selected iff either its upper endpoint lies in $[u, v]$ or if v lies in $[a, b]$. Notice that both conditions are mutually exclusive and exhaust all possible cases of intervals being selected. To implement this counting procedure we only need two atomic sketches—one that keeps track of the intervals as a whole (X_{I}), and one for their *upper endpoints* (X_{U}):

$$X_{\mathrm{I}} = \sum_{[a,b]\in R} \bar{\xi}_{[a,b]}; \qquad X_{\mathrm{U}} = \sum_{[a,b]\in R} \bar{\xi}_{[b]} \ .$$

LEMMA 9. *For interval set R let X_{I} and X_{U} be atomic sketches as defined above. For range query $q = [u, v]$ let $Z = \bar{\xi}_{[u,v]} X_{\mathrm{U}} + \bar{\xi}_{[v]} X_{\mathrm{I}}$. Then $E[Z] = |Q([u, v], R)|$ and $\mathrm{Var}[Z] \leq 2 \cdot (3 \log_2 n + 1) \cdot \mathrm{SJ}(R)$ where n is the size of the domain $\mathcal{N}$.*

The logarithmic factor is caused by the fact that the dyadic cover of $[u, v]$ can consist of up to $2 \log_2 n$ intervals and the dyadic point cover of v contains $\log_2 n + 1$ intervals. Compared to the result for the spatial join (see Section 4.1.4) the self-join size here does not depend on the lower interval endpoints and hence will be smaller. The generalization to d dimensions follows the same pattern as for the spatial join (just replace X_{E} with X_{U}).

6.5 Taking Data Properties into Account

In Section 3.1 we presented two different atomic sketch approaches for summarizing intervals: standard sketches and dyadic sketches. For each interval $[a, b] \in R$, the standard sketch adds the ξ-variables for all points in $[a, b]$ to the interval sketch V_{I}, a cost of $\mathrm{O}(n)$. If the domain is large and the data set contains many long intervals, then the dyadic atomic sketches are clearly preferable since they guarantee $\mathrm{O}(\log n)$ update cost. However, if most intervals are very short, then the standard sketch might deliver better estimates for the same storage cost. Recall that the dyadic endpoint sketch keeps track of all dyadic intervals covering an endpoint, hence it will add the ξ-variable for the dyadic interval that covers the whole domain on each interval insertion. For sets with mostly short intervals this might lead to a higher self-join size than if standard atomic sketches were used.

We therefore propose an adaptive approach. Based on statistics about the interval length distribution, the algorithm determines the *maximum level*, maxLevel. When computing the cover of an interval or endpoint it only uses dyadic intervals from levels up to maxLevel. Note that the input stream can have intervals of length greater than $2^{maxLevel}$. The lower maxLevel, the lower the self-join size for X_{E}. On the other hand, long intervals will require more dyadic intervals than before (since the longer dyadic intervals on the upper levels above maxLevel can not be used any more). Notice that the more small intervals the data set contains, the lower maxLevel, in the extreme the dyadic sketch turns into the standard sketch for maxLevel = 0.

7. EXPERIMENTS

We evaluate the performance of our spatial join size estimation techniques (henceforth referred to as *SKETCH*) on both synthetic and real life 1D and 2D spatial datasets. We compare our algorithms with recently proposed techniques based on generalized Euler Histograms [26] (henceforth referred to as *EH*) and with the Geometric Histograms technique [5] (henceforth referred to as *GH*). These are currently the best known techniques for spatial join selectivity estimation.[3]

The *EH* approach partitions the data space using a grid of a given level L. A grid of level L partitions each dimension into 2^L equi-width cells. An Euler histogram allocates buckets not only for grid cells, but also for grid edges and grid vertices. Besides storing object counts in a cell, the generalized Euler histogram [26] also stores information such as the average height, width and area of the intersection regions between the objects and the cell. In terms of storage space, a generalized Euler histogram of level L uses $9 \cdot 2^{2L} - 6 \cdot 2^L + 1$ units (words) of memory.

Geometric Histograms also partition the data space using a grid of given level L, similar to the generalized Euler Histograms. The information stored in each cell is the total number of corner points, the sum of the areas of the objects, the sum of the lengths of the vertical edges and the sum of the lengths of the horizontal edges of objects intersecting the cell. Thus a Geometric Histogram of level L uses 4^{L+1} units of memory.

In the following, all references to memory allocation refer to the memory allocated *per dataset* to the *SKETCH*, *GH* or *EH* techniques.

7.1 Effect of Input Size and Skew

None of the existing techniques provides guaranteed error bounds (probabilistic or otherwise) for spatial join size estimation. Hence, for comparison purposes, we provide *SKETCH*, *GH* and *EH* with the same space and compare the actual relative errors on their respective estimates as the size of the spatial data set increases, for datasets with different degrees of skew.

We use synthetic two-dimensional datasets, with intervals along each dimension i generated independently according to a Zipfian distribution with Zipf parameter z_i. The average length of an object along a dimension is $O(\sqrt{d_i})$ where d_i is the size of the domain along the i^{th} dimension. We used generalized Euler histograms with grid level $L = 6$, which corresponds to about $36K$ units of memory. The same space was allocated to *SKETCH* and *GH*, and the actual relative errors are shown in Figures 5 and 6. Since our sketch techniques are based on randomization, the relative errors reported are averages over multiple independent runs.

Figure 5 plots the relative error as the size of the dataset increases from $30K$ to 0.5 million. Both the joining datasets had the same size, as specified on the x-axis. The datasets used here have rectangles generated on the two streams with Zipf parameter $z = 0$ (uniform). As can be seen from the figure, the error for *SKETCH* (and *GH*) remains fairly stable as the input size increases, if the distribution of the datasets remains the same. For uniform data ($z = 0$), the *SKETCH* and *GH* techniques perform similarly, with an average rel-

[3] The authors would like to thank Chengyu Sun for providing the original *EH* code and the data sets used in [26].

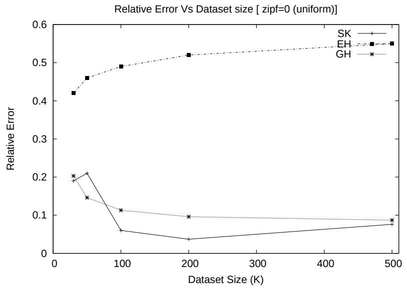

Figure 5: Zipf = 0

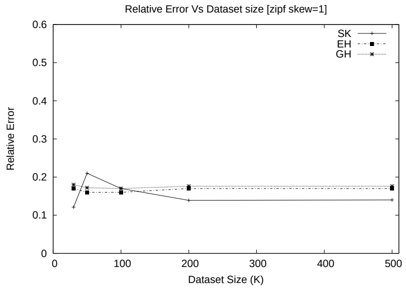

Figure 6: Zipf = 1

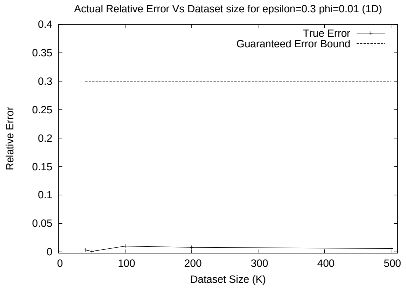

Figure 7: Relative error guarantee

ative error which is much lower than the error of the *EH* technique. Similar results were observed on other datasets with low skew in the distribution of the spatial objects over the data space. Note that *EH* is more general than *GH*, and hence its performance is more sensitive to data distribution and grid level, which explains the poor performance for the given data set.

We would like to point out that for a given grid level, the performance of both *EH* and *GH* depends heavily on the domain size, since that determines the granularity of the grid. Thus, for example, if the domain size is doubled along each dimension, the relative errors for both *EH* and *GH* increase considerably, even if the input is not changed! For the *SKETCH* technique, on the other hand, if the maximum dyadic level is not changed (see Section 6.5), the relative error remains the same for the same input, even if the underlying domain is 'conceptually' (i.e. without changing the input datasets) doubled.

Figure 6 shows the relative error for datasets with a fairly high degree of skew in the distributions of the spatial objects. For these datasets, the projection of the objects along each dimension is distributed with Zipf parameter $z = 1$. Here the performance of *EH* is comparable to *GH* and *SKETCH*, with *SKETCH* faring marginally better than the other two techniques. For other data sets we observed the general trend that *SKETCH* does comparatively better than the grid-based histogram techniques for skewed input.

7.2 Error Guarantees and Space Requirements

Our next set of experiments shows how the *actual* relative error and space requirement for our technique varies for a given guaranteed relative error bound for spatial datasets of different sizes. In Figures 8 and 7 we consider interval joins of two datasets with intervals uniformly distributed over domains of sizes ranging from 16384 to 65536. Figure 8 shows the space requirement and Figure 7 shows the actual relative error for a guaranteed relative error bound of 0.3 at the 99% confidence level. As can be seen from the figure, the space required remains almost constant at around 63K as the size of the dataset increases. The reason is that the distribution of the objects does not change significantly. Note that for a d-dimensional dataset of size N, the total space required to store the dataset completely is $2d \cdot N$. Thus the size of the *SKETCH* summary structure as a fraction of dataset size varies from around 60% for smaller datasets, to about 6% for larger datasets. As can be seen from Figure 7, the actual relative error is much smaller than the guaranteed error bound.

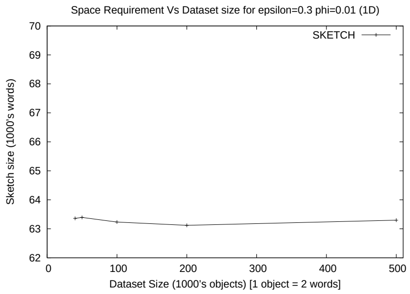

Figure 8: Space requirement

7.3 Real Life Datasets

In this section, we compare the performance of *EH*, *GH* and *SKETCH* on the three real life 2-dimensional spatial datasets used in [26]. The descriptions of the datasets are as follows:

- **LANDO**: This dataset contains land cover ownership and management information for the state of Wyoming at a $1 : 10^6$ scale. Number of objects = 33860.
- **LANDC**: This dataset contains land cover information such as vegetation types for the state of Wyoming at a $1 : 10^6$ scale. Number of objects = 14731.
- **SOIL**: This dataset represents soils of Wyoming at a $1 : 10^6$ scale. Number of objects = 29662.

We consider all the 3 combinations of joins on these spatial datasets. Figures 9, 10 and 11 show the relative errors obtained on **LANDO** $\bowtie_o$ **LANDC**, **LANDC** $\bowtie_o$ **SOIL** and **LANDO** $\bowtie_o$ **SOIL** respectively, as the allocated space is varied.

The three graphs are similar in nature. The most surprising result is the *unpredictability* of the *EH* estimates: Whereas the estimates provided by *SKETCH* improve as the space allocated to it is increased, histogram-based methods (with no quality guarantees) such as *EH* might sometimes produce worse estimates. Similar to the results in [26], *EH* provides good estimates with small memory allocated to it, but the relative error increases rapidly with finer grid partitioning. This behavior is related to the probabilistic model used for estimation by *EH* — it might produce small per-bucket errors which add up if the actual data distribution differs from the model assumption. The *GH* technique is more robust than *EH* in this regard since it uses a simpler model. However, it fares well only for higher amounts

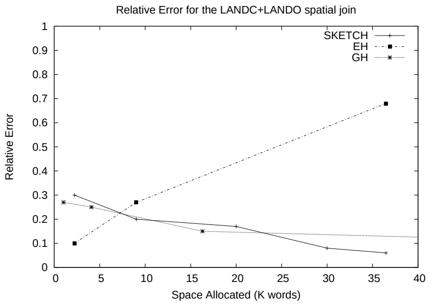

Figure 9: LANDC $\bowtie_o$ LANDO

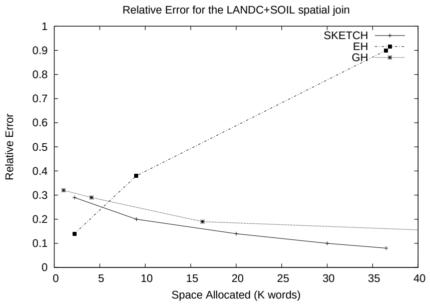

Figure 10: LANDC $\bowtie_o$ SOIL

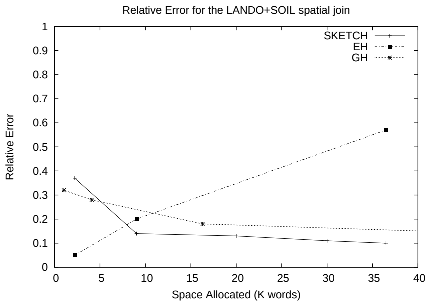

Figure 11: LANDO $\bowtie_o$ SOIL

of memory, and is mostly outperformed by *SKETCH* (by a small margin). The relative error of *SKETCH* shows a steady decline with increasing space in all the three graphs. This is expected, since *SKETCH* is an unbiased estimator and therefore the more independent instances of this estimator are used, the better the expected results.

7.4 Discussion of Experimental Results

We presented a comparison of *SKETCH*, *GH* and *EH* on synthetic and real life datasets. Overall, the performance of *GH* and *SKETCH* is comparable, with *SKETCH* performing slightly better. In general the experiments highlight the practicality of our randomized estimation technique. It not only provides quality guarantees, but also has predictable behavior: The more space we allocate, the better the estimate. The histogram techniques might sometimes produce better estimates, sometimes they do much worse. Without knowing the data distribution in advance, their behavior and best storage allocation cannot be determined. For instance, the *EH* technique does very well for small space, but produces high errors when the number of buckets is increased. Since in practice *robustness* of an estimation technique is an important factor, we believe that *SKETCH* in general is preferable over techniques whose behavior cannot be predicted a priori.

There are essentially only two parameters that effect *SKETCH*'s performance: the self-join sizes of the data sets, and the result size. As long as the self-join sizes are not too large compared to the result size, *SKETCH* provides good estimates and its performance is independent of the actual object distribution (skew) in the joining spatial datasets. This dependence on result size is a characteristic of all probabilistic estimation techniques with provable error bounds.

8. RELATED WORK

The best known general techniques for selectivity estimation for joins involving spatial objects are based on sampling or on histograms. An et al. [5] examine different sampling techniques and propose new histogram-based approaches. Their results indicate that for achieving a comparable accuracy, histograms will require less storage and estimation time than sampling. Mamoulis and Papadias [23] take a more general approach of analytically estimating the selectivity of complex spatial queries, i.e., queries that combine selection and join operators. Their formulas are based on uniformity assumptions. A skewed data set is partitioned into a regular grid of cells, and the formulas are applied for each cell. Sun et al. [26] improve on these results with a framework based on Euler histograms [25]. They also use a regular grid partitioning of the space, but the Euler histograms together with adaptively selected per-cell estimation techniques provide more accurate estimates for spatial joins with geometric selections.

Faloutsos et al. [15] and Belussi and Faloutsos [8] propose parametric methods for estimating the selectivity of ε-(self-) joins of point-sets. For self-similar data sets they achieve good approximations using power laws and fractal dimensionality. The techniques of Acharya et al. [2] estimate the selectivity for point and range queries over two-dimensional rectangular data. The basic idea is to partition the data space into buckets and to use models based on uniformity assumptions per bucket. Closely related to selectivity estimation are approaches for estimating the cost (in terms of CPU or I/O) of specific operator implementations. Examples are cost models for range queries and index-supported joins for spatial data [9, 20, 21, 22, 29, 30].

None of the previous approaches provides any result quality guarantees. Samples are difficult to maintain in the presence of updates, especially deletes which could remove objects from the sample. Similarly, there is no efficient algorithm for maintaining fractal dimensionality and power law parameters in the presence of updates. The histogram techniques, due to the regular grid partitioning, are easy to maintain dynamically, however, their utility is limited when the data is skewed. Other histograms that are more appropriate for skewed data cannot be maintained efficiently and introduce an additional error when the buckets of the joined data sets do not align.

Our techniques are based on AMS sketches [4, 3] which are dynamically maintainable and provide approximation quality guarantees. Different types of sketches so far have been used for efficiently maintaining aggregates over data streams [6]. Example applications are complex aggregates [14], quantiles [17], frequent items [10, 11], multidimensional histograms [28], and graph statistics [7]. An upcoming paper by Tao et al. proposes to use sketches to improve the result quality for aggregate queries over spatio-temporal point sets [27].

9. CONCLUSION AND FUTURE WORK

In this paper, we proposed a new framework for using spatial sketch techniques for approximately answering spatial queries. To the best of our knowledge, our technique is the first method for spatial data that gives probabilistic quality guarantees. It permits incremental construction under insertion and deletion of records while at the same time being effective for skewed distributions and applications where deletions of objects are frequent. Our experimental evaluation shows that our techniques are competitive compared to the best previously known methods, and we believe that their additional features of quality guarantees, incremental maintenance, and predictable behavior make them a viable choice in practice.

Note that in principle any sketching technique that can estimate join sizes with guaranteed error bounds (not only AMS sketches) could be incorporated into our framework. Thus for example one could potentially use some of the sketching techniques proposed in upcoming publications [12, 16] with their corresponding error guarantees suitably adapted to our spatial framework.

In future work, we plan to extend our techniques to other spatial queries, including non-rectangular selections [1]; and we would like to incorporate quality boosting techniques such as sketch partitioning into our framework. One especially promising candidate for future work are complex spatial queries that contain multiple operators, e.g., joins with selections. Selectivity estimation for spatial joins with selections can be done using our sketch-based framework. However a straightforward application (use counting sketches like for spatial joins, but multiply each inserted interval and endpoint contribution with a factor that encodes the range-query sketch for independent ξ-families), would lead to dramatically increased variance and hence space requirement, compared to spatial join without selection.

10. REFERENCES

[1] A. Aboulnaga and J. F. Naughton. Accurate estimation of the cost of spatial selections. In *Proc. ICDE*, pages 123–134, 2000.

[2] S. Acharya, V. Poosala, and S. Ramaswamy. Selectivity estimation in spatial databases. In *Proc. SIGMOD*, pages 13–24, 1999.

[3] N. Alon, P. B. Gibbons, Y. Matias, and M. Szegedy. Tracking join and self-join sizes in limited storage. In *Proc. PODS*, pages 10–20, 1999.

[4] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. In *Proc. STOC*, pages 20–29, 1996.

[5] N. An, Z.-Y. Yang, and A. Sivasubramaniam. Selectivity estimation for spatial joins. In *Proc. ICDE*, pages 368–375, 2001.

[6] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In *Proc. PODS*, pages 1–16, 2002.

[7] Z. Bar-Yossef, R. Kumar, and D. Sivakumar. Reductions in streaming algorithms, with an application to counting triangles in graphs. In *Proc. SODA*, pages 623–632, 2002.

[8] A. Belussi and C. Faloutsos. Self-spacial join selectivity estimation using fractal concepts. *Proc. ACM TOIS*, 16(2):161–201, 1998.

[9] T. Brinkhoff, H.-P. Kriegel, and B. Seeger. Parallel processing of spatial joins using R-trees. In *Proc. ICDE*, pages 258–265, 1996.

[10] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. In *Proc. ICALP*, pages 693–703, 2002.

[11] G. Cormode and S. Muthukrishnan. What's hot and what's not: Tracking most frequent items dynamically. In *Proc. PODS*, pages 296–306, 2003.

[12] G. Cormode and S. Muthukrishnan. An improved data stream summary: The count-min sketch and its applications. In *LATIN*, pages 29–38, 2004.

[13] A. Das, J. Gehrke, and M. Riedewald. Approximation techniques for spatial data. Technical report, Cornell University, 2004.
http://techreports.library.cornell.edu.

[14] A. Dobra, M. N. Garofalakis, J. Gehrke, and R. Rastogi. Processing complex aggregate queries over data streams. In *Proc. SIGMOD*, pages 61–72, 2002.

[15] C. Faloutsos, B. Seeger, A. J. M. Traina, and C. Traina. Spatial join selectivity using power laws. In *Proc. SIGMOD*, pages 177–188, 2000.

[16] S. Ganguly, M. N. Garofalakis, and R. Rastogi. Processing data-stream join aggregates using skimmed sketches. In *Proc. EDBT*, pages 569–586, 2004.

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

[18] P. J. Haas and J. M. Hellerstein. Online query processing. In *Proc. SIGMOD*, page 623, 2001.

[19] J. M. Hellerstein, P. J. Haas, and H. J. Wang. Online aggregation. In *Proc. SIGMOD*, pages 171–182, 1997.

[20] Y.-W. Huang, N. Jing, and E. A. Rundensteiner. A cost model for estimating the performance of spatial joins using R-trees. In *Proc. SSDBM*, pages 30–38, 1997.

[21] J. Jin, N. An, and A. Sivasubramaniam. Analyzing range queries on spatial data. In *Proc. ICDE*, pages 525–534, 2000.

[22] H.-P. Kriegel, M. Pfeifle, M. Pötke, and T. Seidl. A cost model for interval intersection queries on RI-trees. In *Proc. SSDBM*, pages 131–141, 2002.

[23] N. Mamoulis and D. Papadias. Selectivity estimation of complex spatial queries. In *Proc. SSTD*, pages 155–174, 2001.

[24] D. Papadias, Y. Theodoridis, T. K. Sellis, and M. J. Egenhofer. Topological relations in the world of minimum bounding rectangles: A study with R-trees. In *Proc. SIGMOD*, pages 92–103, 1995.

[25] C. Sun, D. Agrawal, and A. El Abbadi. Exploring spatial datasets with histograms. In *Proc. ICDE*, pages 93–102, 2002.

[26] C. Sun, D. Agrawal, and A. El Abbadi. Selectivity estimation for spatial joins with geometric selections. In *Proc. EDBT*, pages 609–626, 2002.

[27] Y. Tao, G. Kollios, J. Considine, F. Li, and D. Papadias. Spatio-temporal aggregation using sketches. In *Proc. ICDE*, 2004. to appear.

[28] N. Thaper, S. Guha, P. Indyk, and N. Koudas. Dynamic multidimensional histograms. In *Proc. SIGMOD*, 2002.

[29] Y. Theodoridis and T. K. Sellis. A model for the prediction of R-tree performance. In *Proc. PODS*, pages 161–171, 1996.

[30] Y. Theodoridis, E. Stefanakis, and T. K. Sellis. Efficient cost models for spatial queries using R-trees. *IEEE TKDE*, 12(1), 2000.