Containment Join Size Estimation: Models and Methods

Wei Wang Haifeng Jiang Hongjun Lu
Department of Computer Science
The Hong Kong University of
Science and Technology
Hong Kong, China
{fervvac, jianghf, luhj}@cs.ust.hk

Jeffrey Xu Yu
Department of Systems Engineering and
Engineering Management
The Chinese University of Hong Kong
Hong Kong, China
yu@se.cuhk.edu.hk

ABSTRACT

Recent years witnessed an increasing interest in researches in XML, partly due to the fact that XML has now become the *de facto* standard for data interchange over the internet. A large amount of work has been reported on XML storage models and query processing techniques. However, few works have addressed issues of XML query optimization. In this paper, we report our study on one of the challenges in XML query optimization: containment join size estimation. Containment join is well accepted as an important operation in XML query processing. Estimating the size of its results is no doubt essential to generate efficient XML query processing plans. We propose two models, the interval model and the position model, and a set of estimation methods based on these two models. Comprehensive performance studies were conducted. The results not only demonstrate the advantages of our new algorithms over existing algorithms, but also provide valuable insights into the tradeoff among various parameters.

1. INTRODUCTION

XML, the Extensible Markup Language, has now become the *de facto* standard for data interchange over the internet. Unlike relational data, a unique feature for data encoded in XML format is that they are virtually semi-structured data. The structure of XML data is usually defined by XML schema (or previously DTD). Several data models have been proposed for XML data, e.g., Document Object Model (DOM), and they all assume a tree based model to represent XML data. Consequently, many query languages proposed are designed to allow queries that have both content and structural constraints. For example, the XPath query `//paper[appendix/table]` selects those papers that have a table in their appendix sections.

Containment join, proposed in [29], has been recognized as the core part for XML query processing. A containment join between a set of ancestor nodes (denoted as A) and a set of descendant nodes (denoted as D) is to find all pairs of (a, d), $a \in A$ and $d \in D$, such that a is an ancestor of d. Region coding scheme, i.e., each element is encoded with its (physical or logical) *start* and *end* positions, is a powerful technique widely used to efficiently evaluate containment join. When the region coding scheme is used, a containment join between A and D can be evaluated as the following complex *relational* θ-join: $A \bowtie_\theta D$, where the θ is $A.start < D.start \wedge D.end < A.end$. [20, 29, 2] have shown that containment joins not only can be used to answer arbitrary XPath queries, but often outperform other alternative methods as well.

While there are many studies on efficient containment join algorithms, there has not been much research into the size estimation problem for containment join. An accurate estimate of the result size of a containment join is a prerequisite for cost-based XML query optimizers. For example, consider the same XPath query: `//paper[appendix/table]`. Given that we have some mechanism to retrieve `//paper`, `//appendix` and `//table` efficiently, a query optimizer has to choose the containment join order. One plan is to containment join `//paper` and `//appendix` and then join the *intermediate result* with `//table` by another containment join. Another plan is to containment join `//appendix` and `//table` first and then join the *intermediate result* with `//paper`. If the sizes of two intermediate results differ greatly, then the total costs of the two plans might differ greatly as well. Accurate estimation of the intermediate result sizes, in this example, can help to choose a better query execution plan. Size estimation can also be useful in internet or interactive applications, where the estimated result sizes can be presented to the user in order to decide whether it is necessary to really launch the query, as it might require substantial system resources.

Containment join on data coded in region codes is essentially a complex relational θ-join. In addition, XML data has unique features that make any estimation method nontrivial. Although we are aware of existing works in the following areas, none of them is directly applicable to our containment join size estimation problem.

- *Relational techniques.* Estimation of join sizes has been widely studied in the context of relational equijoins. However, the proposed methods were either designed to optimize for the worst case [18] or hard to be generalized to handle complex inequality join conditions [21, 13, 3].
- *Estimation methods for XML data.* To the best of our

SIGMOD 2003, June 9-12, 2003, San Diego, CA.

knowledge, [28] is the only work that addressed the same estimation problem for XML data. In particular, PH and coverage histograms were proposed for join size estimation. However, those methods rely on a few *strong* assumptions and their performance may degrade greatly if the data do not conform to the assumptions.

- *Spatial techniques.* Most existing spatial join size estimation methods [5, 23, 27] rely heavily on multidimensional uniform distribution assumption and cannot take into consideration the constraints of XML data distribution.

In this paper, we propose two models, the interval model and the position model, and a set of estimation methods based on these two models. A unique feature of our models is that they can capture alternative information for an element set according to its role (ancestor or descendant set) in a containment join. The interval model treats every element in a node set as an interval, when the node set acts as the ancestor set in the join or a point, when the node set acts as the descendent set. In the position model, conceptually, two auxiliary tables are constructed for each element set. The former table records the coverage information when the element set acts as the ancestor set, while the latter captures the *start* position information of each element when the element set acts as the descendant set. These two models enable us to transform the original challenging estimation problem to other estimation problems which are tractable. Based on the two models, two classes of estimation algorithms are proposed: histogram based and sampling based algorithms. We show that our histogram based method uses weaker and more reasonable assumptions than those used by PH histogram and Coverage histogram [28]. Our sampling based methods leverage adaptive sampling techniques and have provable accuracy of the estimated results. Extensive experiments were conducted. The results not only demonstrate the advantages of our new algorithms over existing ones, but also provide valuable insights into the tradeoff among various parameters.

The major contributions of the paper can be summarized as follows.

1. We propose two models for the containment join size estimation problem. The challenge of the original problem is to estimate the result size for a complex, inequality join, which renders existing models and hence methods inapplicable. We show that, in our newly proposed models, the original problem can be converted to other estimation problems that are tractable.

2. We propose and study two classes of estimation methods based on the new models. For each method, we exhibit our efforts to exploit the unique features of XML data and containment joins. We also give theoretical bounds on the quality of estimated results (whenever applicable) as well as detailed discussions.

3. We report results of experiments performed for various datasets under different settings. Our new algorithms are shown to outperform previous ones up to an order of magnitude and are more robust. The experiment results also provide valuable insights into the tradeoff between various parameters.

The rest of the paper is organized as follows. Section 2 presents related work. Section 3 introduces and discusses two proposed models. Histogram based approaches are discussed in Section 4 while Section 5 discusses approaches based on sampling. Section 6 presents our detailed experiment results and analyses. Section 7 concludes the paper.

2. RELATED WORK

We will give a brief review of existing works on statistical estimation. Then we focus on PH histogram [28] and bifocal sampling [13], which are closely related to the work reported in this paper.

Although histograms are by far the most popular statistical data structure for estimating query sizes, there is little work on estimating join sizes using histograms. [18] studied the kind of histogram based method that is optimal in the worst case and pointed out that all histograms are equally good on average with respect to the metric defined.

There is not much work on estimating join sizes in the spatial database context either. [5] studied several histogram schemes for spatial join estimation. They proposed the parametric histogram (PH) based on the previous work [6]. Another histogram, Geometric Histogram (GH), was also proposed, based on the key observation that the intersection of two MBRs results in four intersecting points. GH was demonstrated to outperform PH and other methods without using histograms. The GH method, however, requires a fine grid over the workspace and may result in inaccurate estimation when this condition fails. [23] proposed formulae for selectivity estimation of pairwise joins restricted by two selections. Their focus is on adjusting the workspace given two original workspaces of the joining datasets. Recently, a framework for spatial join size estimation was proposed in [27], based on the notion of Euler Diagram. Different probabilistic data distribution models can be plugged into the framework which generates different formulae to estimate the join results of corresponding grids from the datasets. Specifically, they integrated the formulae in [23, 5] and showed the improved accuracy over previous approaches.

There have been many studies of (adaptive) sampling based estimation algorithms since Hou, Özsoyoglu and Taneja presented their initial work in this area [16, 17]. [14] introduced a partial ordering that compares the variability of the estimators of different classes, such as *t_index*, *t_cross*, etc. [13] proposed a new *bifocal* sampling technique that overcomes some well-known problems in previous schemes. We will discuss it in more details later. Recently, [22] reexamined Lipton and Naughton's adaptive sampling algorithm [21] and Haas and Swami's adaptive algorithm mainly in terms of efficiency. It also studied the Monte-Carlo algorithm due to E. Cohen [9]. [15] proposed the *systematic sampling* technique, which was demonstrated to outperform the *t_cross* sampling method with the same amount of sampling. The idea is to sample the first $k = \lceil \frac{N}{n} \rceil$ tuples of R and every k-th tuple thereafter in the sorted order of the join attribute.

More recently, [4, 3, 10] proposed to use *sketch* based methods to estimate self-join and join sizes in the stream data processing setting. The basic idea of those approaches is much similar to that of sampling approaches: in order to estimate a function of $F(x)$ over a dataset, we instead compute an unbiased estimator $\hat{F}(x)$ of $F(x)$ such that the

variance of $\hat{F}(x)$ is small and bounded. The mathematical tool used is *four-wise independent binary random variables*, which can be constructed from the *orthogonal array* of strength 4. Although sketch based methods have been successfully applied in many problems, it is not obvious to be applicable to the containment join size estimation problem.

A different but related problem is estimation of the selectivity of path expressions for XML data [8, 1, 12, 24]. Most existing approaches tried to solve the problem by capturing the structure of the XML data tree or graph. [1] proposed two techniques, namely path trees and Markov tables, to summarize the structures of XML data. [12] employed XML Schema types and histograms as statistical summaries. [24] exploited localized graph stability in a graph-synopsis model to approximate path and branching distribution in an XML data graph. Our problem differs from theirs in that we do not have any restriction on the two datasets involved in the join nor do we need to capture the global statistics for the whole XML document.

2.1 PH Histogram and Coverage Histogram

[28] is the only previous work that addressed the *same* containment join size estimation problem. Their approach is to mapping XML data into 2D space and maintaining certain statistics for data fallen in each pre-defined grid over the workspace. XML specific characteristics, such as (a) strictly nested property and (b) nesting within each dataset, are taken into consideration, which results in the PH-Histogram and Coverage Histogram. The underlying assumption is two-dimensional uniform distribution of both datasets. That is, as we will show later, a strong and *static* assumption. For example, the estimated result for two corresponding, off-diagonal grids (buckets) will always be $\frac{1}{4} \cdot n_a \cdot n_d$. In addition, their PH histogram based estimation methods will give a highly erroneous estimation when no two nodes in the ancestor set can be of ancestor-descendant relationship, i.e., when the ancestor set has *no-overlap* property. Coverage histogram and new estimation algorithms were hence proposed as a remedy. However, their new method still suffers a lot from the additional assumption that the global coverage statistics are the same as the local coverage statistics.

2.2 Bifocal Sampling

Bifocal sampling was introduced as a new adaptive sampling technique for estimating equijoin sizes in the relational context [13]. It classifies tuples in each relation into two groups, sparse and dense, based on the number of tuples with the same join value. Distinct estimation procedures were employed for various combination of joining tuples, namely, the DenseDense subjoins and SparseAny subjoins. Bifocal sampling was shown to be able to overcome some well-known problems in previous adaptive sampling schemes. However, it focuses on relational equijoin only and it is not obvious as to extend the method to the complex containment join here.

3. ESTIMATION MODELS FOR XML DATA

In this section, we first give the formal definition of the problem, followed by discussions of the unique features of XML data and containment joins. They render existing estimation methods in other areas not directly applicable. Two novel models that address above-mentioned challenges are proposed. They underpin the three classes of estimation algorithms to be discussed in the later sections.

3.1 Problem Definition and Challenges

XML data are commonly modelled as trees, termed as data trees. Given an XML data tree $\mathcal{T}$, we can get a set of element nodes by certain predicate based on values and structures. For example, the predicate could be an XPath query like `//appendix`. A set of element nodes is defined when the above predicate is evaluated against a given XML data tree.

A containment join between two element sets A and D returns all pairs of (a, d), where $a \in A$ and $d \in D$, such that a is an ancestor of d. [29] proposed to use region codes, i.e., $(start, end)$, for each element and the containment join between A and D can be evaluated as a complex θ-join: $A \bowtie_\theta D$, where $\theta = A.start < D.start \wedge D.end < A.end$. Note that due to the strictly nested property of XML data, the above join condition can be simplified to θ' as $A.start < D.start < A.end$. In addition, without loss of generality, we assume the region codes are *distinct*, that is, no two elements in $\mathcal{T}$ have the same *start* or *end* value.

Given an XML data tree $\mathcal{T}$ labelled with region codes, we denote c_{min} and c_{max} as the minimum of the *start* codes of all elements in the tree and the maximum of the *end* codes of all elements in the tree respectively. Formally, $c_{min} = \min_{e \in \mathcal{T}}\{e.start\}$ and $c_{max} = \max_{e \in \mathcal{T}}\{e.end\}$. The region $[c_{min}, c_{max}]$ is also termed *workspace*.

We formally define our containment join size estimation problem as follows.

Definition 1. Given two element sets A and D where every element in both sets is encoded in region code, i.e., $(start, end)$, correctly estimate the cardinality of the containment join result between A and D.

There are several unique features of our estimation problem that impose serious challenges.

1. *Complex θ-join conditions.* The join condition consists of two correlated inequality conditions on multiple attributes.

2. *Data distribution.* There are two constraints that confine the distribution of the elements.

   (a) *end* value should always be greater than *start*.

   (b) Due to the strictly nested nature, the regions of any two elements cannot partially overlap. In the two-dimensional description, this constraint results in the *forbidden region* observed in [28].

3. *Correlation pattern.* Correlation is recognized as the major difficulty in estimating the result size of equijoins or spatial joins. There are two observations about the correlation between the ancestor and descendant sets.

   (a) The correlation is determined by the structures of the data tree $\mathcal{T}$.

   (b) The correlation effects of the two sets are asymmetric. Specifically, one element in the ancestor set could join with all elements in the descendant set, while one element in the descendant set can only join with up to H elements in the ancestor

set, where H is the height of the XML data tree $\mathcal{T}$.

Now we will briefly review previous models and show that their methods are not well suited to capture the above-mentioned features.

For one-dimensional histogram based methods, they are based on the model of *frequency distribution table* (or *data distribution*), which contains the set of pairs $T = \{(v_1, f_1), \ldots, (v_D, f_D)\}$, where v_i is the value of the attribute in question and f_i is the number of tuples in the relation that have value v_i in that attribute [25]. However, such model cannot be directly extended to join predicates that involve multiple attributes. Multidimensional histograms were also proposed, but their ability to capture correlation among dimensions is still questionable, let alone the unique correlation pattern in XML data. Last, histogram based methods are best suited for answering range/point queries, but not joins.

Both sampling and sketch based methods are based on statistic theorems. There has been no reported work on using sampling techniques in the presence of inequality joins. Sketch based methods, while performing well for joins and self-joins, cannot be easily extended to handle inequality joins either.

There are only few works for spatial join size estimation. All the existing works make some assumption on the data distribution of two joining datasets. Either power-law compliant assumption [11] or multidimensional uniform / independence assumptions are used. Those assumptions are not suitable for XML data either.

Wavelet or Fourier transformation based methods were successfully used in estimation problems in one-dimensional or multidimensional context. However, one unsolved difficulty is that it is still unknown how to find good transformation for data distribution with certain constraints. For example, a large portion of approximated data could be invalid due to their violation of the strictly nested property. We defer studies in this direction to our future research.

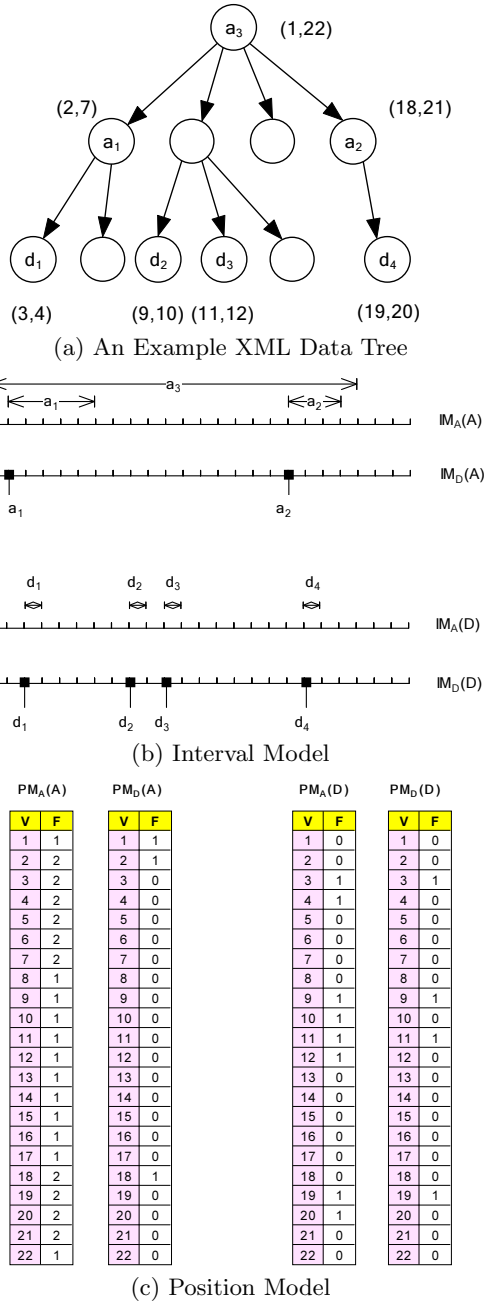
3.2 Interval Model

The interval model is a one-dimensional model. Conceptually, each element set corresponds to two sets: interval set and point set. An interval set contains intervals and a point set contains points. The interval set of element set S, denoted as $IM_A(S)$, is used when S acts as an ancestor set in a containment join. For each element e in S, it corresponds to an interval of $[e.start, e.end]$. The point set of element set S, denoted as $IM_D(S)$, is used when S acts as a descendant set in a containment join. For each element e in S, it corresponds to a point $e.start$.

Theorem 1 indicates that the original estimation problem can be converted into the problem of estimating overlapping interval-point pairs.

THEOREM 1. *Under the interval model, the result size of the containment join between A and D is equal to the number of overlapping interval-point pairs between $IM_A(A)$ and $IM_D(D)$.*

Example 1. We show an example XML data tree in Figure 1(a). We choose three nodes as the ancestor set A and four nodes as the descendant set D. Their region codes are also shown respectively. For example, node a_1 has region code (2, 7).



(a) An Example XML Data Tree

(b) Interval Model

(c) Position Model

Figure 1: Illustration of Two Models

The interval sets and the point sets of the ancestor set and the descendant set are shown in Figure 1(b). Node a_1, for example, corresponds to an interval from 2 to 7 in $IM_A(A)$ and node d_2 corresponds to a point at 9 in $IM_D(D)$.

3.3 Position Model

The position model is also a one-dimensional model. Conceptually, each element set corresponds to two tables: *covering table* and *start table*. Both tables have the same schema: (V, F), where $dom(V) = [c_{min}, c_{max}]$ and $dom(F) = \{x \in Z | x \geq 0\}$. The covering table of element set S, denoted as $PM_A(S)$, is used when S acts as an ancestor set in a containment join. For every tuple $t \in PM_A(S)$, $t.F = |\{e | e.start \leq$

$t.V \leq e.end\}|$. That is, the F field records the number of elements whose regions *cover* the corresponding V value. The start table of element set S, denoted as $PM_D(S)$, is used when S acts as a descendant set in a containment join. For every tuple $t \in PM_D(S)$, $t.F = |\{e|e.start = t.V\}|$. That is, the F field records the number of elements whose *start* values are equal to the V value. Note that, since the codes are distinct, the value of F for any tuple in $PM_D(S)$ can be either 0 or 1, as there could only be 0 or 1 element with a given *start* value.

Theorem 2 indicates that the original estimation problem can be converted into the estimation of the size of an equi-join.

THEOREM 2. *Under the position model, the result size of the containment join between A and D is equal to the inner product of of $PM_A(A)$ and $PM_D(D)$, i.e., $\sum_{c_{min} \leq i \leq c_{max}} (PM_A(A)[i] \cdot PM_D(D)[i])$.*

Example 2. We show the covering and start tables for both ancestor and descendant sets in Figure 1(c). For example, position 19 is covered by two ancestor nodes, namely a_2 and a_3, therefore, its F value is 2. The inner product of $PM_A(A)$ and $PM_D(D)$ is 6, which is equal to the result size of the containment join between A and D.

4. ESTIMATION VIA HISTOGRAM

Based on the interval model, we propose a new histogram based estimation method. In our new method, each descendant element d will correspond to a point in $IM_D(D)$ and each ancestor element a will be an interval in $IM_A(A)$. We will use A to denote $IM_A(A)$ and D to denote $IM_D(D)$ throughout this section, as long as there is no ambiguity.

4.1 PL Histogram

The PL Histogram (Point-Line Histogram) is based on the interval model. The basic idea is to partition the whole workspace $[c_{min}, c_{max}]$ into buckets and estimate the overlapping (interval, point) pairs within each bucket according to some assumptions.

We choose to make the following two basic assumptions:

- Data distribution of A and D are *independent*.
- D conforms to *uniform* distribution in each bucket.

Note that

1. They are not unreasonable assumptions. Independence assumption and uniform assumption are often made in the relational context and spatial context as well. They enable us to do the estimation without any *a priori* knowledge about the data distribution or correlation of the joining data.

2. One-dimensional uniform assumption can be made approximately valid if the data is regular and bucket boundaries are carefully selected. This assumption is weaker than the two-dimensional uniform assumption adopted in the early work, i.e., the PH histogram in [28]. It can be shown that if the data distribution is two-dimensional uniform, it is also one-dimensional uniform. In addition, we only require the assumption on D's distribution.

We now introduce the PL histogram and its estimation algorithm. We first divide the whole workspace of the element set into b buckets. For each bucket, we keep the following statistics n, wss, wse, l. The definitions of the statistics are listed in Table 1, where R could be either A or D.

Table 1: Statistics for PL Histogram

Symbol	Definition
$n(R,i)$	number of intervals/points in the i-th bucket
$wss(R,i)$	start position of the i-th bucket
$wse(R,i)$	end position of the i-th bucket
$l(R,i)$	average length of the intervals in the i-th bucket (not valid for D)

Let us consider the case when the whole workspace is a bucket. For each $a \in A$, according to our assumptions, the expected number of D elements that lie inside the range of a is $\frac{l(a)}{wse-wss} \cdot n(D)$, where $l(a)$ is the length of an element $a \in A$. Therefore the total number of (a,d) pairs can be estimated as

$$\begin{aligned} \hat{X} &= \sum_{a \in A} \frac{l(a)}{wse - wss} \cdot n(D) \\ &= \frac{n(A) \cdot l}{wse - wss} \cdot n(D) \\ &= \frac{l}{wse - wss} \cdot n(A) \cdot n(D) \end{aligned}$$

We can generalize the above formula (for 1 bucket) to the case of b buckets if we use the same partitioning scheme for both A and D. We can then estimate the size of containment join as:

$$\begin{aligned} \hat{X} &= \sum_{1 \leq i \leq b} \hat{X}_i \\ &= \sum_{1 \leq i \leq b} \frac{l(A,i)}{wse(A,i) - wss(A,i)} \cdot n(A,i) \cdot n(D,i) \quad (1) \end{aligned}$$

Note that

1. Our formula is an *adaptive* one. Intuitively, if there is no overlap of a in A, $\frac{n(A)*l}{wse-wss} < 1$ and the result size is less than D. That is reasonable as each $d \in D$ can join with at most one $a \in A$. On the other hand, if elements in A are heavily nested, the results could be larger than D. On the contrary, the formula in PH histogram takes the form of $c \cdot n(A) \cdot n(D)$ for estimation within a bucket (grid), where c is a constant (such as $\frac{1}{4}$, $\frac{1}{12}$).

2. In order to avoid multiple counting when some element lies across bucket boundaries, we use the following rules: each a across multiple buckets will be counted multiple times, while each d across multiple buckets will be only counted in the first bucket, that is, the bucket where $d.start$ is located.

Algorithm 1 describes the procedure to estimate the join size with PL histograms built on both datasets. The algorithm iterates over all the b buckets of A, and for each bucket of A, it calculates the estimated number of joining pairs according to Equation 1 (Line 4). We sum up the estimated result for each bucket of A to get the final estimated result for the containment join between A and D.

Algorithm 1 PL-Hist-Est(A, D)

Input:

A and D are ancestor and descendant sets respectively and both are partitioned identically into b buckets.

Output:

return the estimated size of join results.

Description:

1: $est = 0$
2: **for** $i = 1$ to b **do**
3: $\quad est = est + \frac{l(A,i)}{wse(A,i) - wss(A,i)} \cdot n(A,i) \cdot n(D,i)$
4: **end for**
5: **return** est

4.2 The MRE Measure

Although our formula is accurate in the continuous domain, it can fail for a certain class of cases in the discrete domain when D is relatively sparse. To illustrate this, let us look at the examples in Figure 2. There are 5 uniformly distributed elements (points) from D and 1 element (interval) from A in both Figure 2(a) and (b). Let the interval move from the initial position (the solid line) to the new positions denoted in dashed lines. We calculate and draw in the figure the number of matches (i.e., overlapping interval-point pairs) during this process. It is clear that the number of matches will vary. Intuitively, this illustrates that relative offset of a and d can affect the accuracy of the estimation.

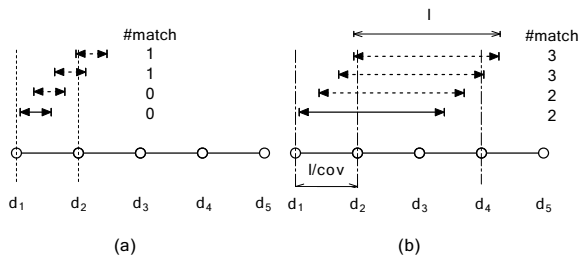


Figure 2: An Example that Shows the Effect of the Offset to the Join Size

Formally, in the case of discrete data (both a and d can only start/end at integer positions and d can never be redundant, i.e., having multiple d's with the same $Start$ position), the estimation formula is correct in the average case, but the real number (even the assumptions are valid) depends much on the relative offset of a and d. Let $cov = \frac{l}{\frac{wse - wss}{n(D)}} = \frac{l \cdot n(D)}{wwe - wws}$. Intuitively cov measures how many d's one a can cover on the average. From Figure 2, we can see that the join result will be $n(A) * \lceil cov \rceil$ with probability $(cov - \lfloor cov \rfloor)$ and be $n(A) * \lfloor cov \rfloor$ with probability $1 - (cov - \lfloor cov \rfloor)$. Traditionally, unlike sampling based methods, estimation given by histogram methods does not give any "confidence" measurement. Here, we define a measure named the *maximum relative error* to characterize the worst case estimation error as

$$\begin{aligned} MRE &= \max\{\frac{|x - \hat{x}|}{x}\} \\ &= \max\{\frac{\lceil cov \rceil - cov}{\lceil cov \rceil}, \frac{cov - \lfloor cov \rfloor}{\lfloor cov \rfloor}\} \end{aligned} \quad (2)$$

Intuitively, if the MRE value is large, we have the risk of having high inaccuracy for the estimation result. The worst case error will be big if cov is small. Specifically, if $cov < 1$, the maximum relative error could be unbounded. Even in terms of the *absolute* error, it could still be big when $cov < 1$. We show the MRE values for cov ranging from 1.0 to 10.0 in Figure 3. We can see that MRE values vary regularly with a period of 1 and the maximum value of MRE within each period goes down consistently when cov increases. We do not plot the cases when $cov < 1$ here, because the MRE values are unbounded.

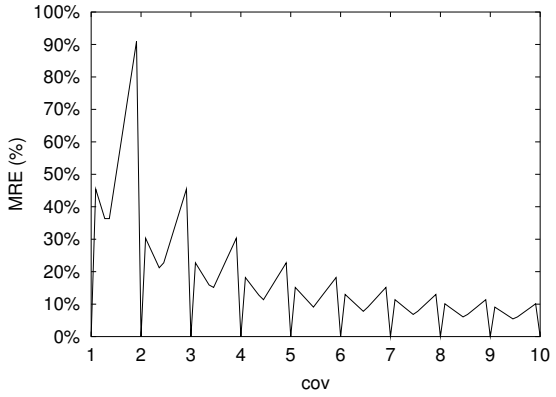


Figure 3: MRE vs cov for $cov > 1$ (MRE is unbounded when $0 < cov < 1$)

To address the above problem, we use the following two approaches simultaneously.

- We provide a "rough" *confidence* measure to the user about the estimated result: the MRE measure given in Equation 2. If $cov > 1$, it can be shown that $0 < MRE < 1$, and the maximum of MRE values over each period decreases when cov increases. Intuitively, MRE gives the bound of relative errors of the estimation when datasets conform well to the assumptions of our PL histogram. Note that the actual relative error could be even higher if the joining datasets do not conform to our assumptions well enough.

- The root of the problem is the inability for histograms to capture correlations. Therefore, we develop other estimation techniques (such as sampling based methods in the next section) which can provide accurate estimation with high *confidence* measures (although the two measures cannot be directly compared) even in the presence of correlation.

5. ESTIMATION VIA SAMPLING

In this section, we present a class of new sampling algorithms, designed to take into consideration unique features of XML data and the containment join.

Although the effectiveness and the practicality of sampling based estimation methods have been demonstrated by many previous research works, none of them considers the case of estimating the result size of complex inequality joins. Here, we consider the estimation problem under both the interval model and the position model. It will be shown that

with the help of the models, our new sampling based algorithms are capable of giving provably good estimation of the containment join size.

5.1 Interval Model Based Adaptive Sampling

In the interval based model, the analogy to the equality condition of two relations is the *overlapping* condition of intervals from $IM_A(A)$ table and points from $IM_D(D)$ table. Our algorithm is inspired by the *bifocal sampling* algorithm [13] as well as the following observation: An interval in $IM_A(A)$ could join with many points in $IM_D(D)$ (up to $|D|$), while a point in $IM_D(D)$ can only join with relatively much fewer intervals in $IM_A(A)$ (up to H, where H is the height of the XML data tree). Bifocal sampling will classify all subjoins between A and D into two classes, *DenseDense* subjoins and *SparseAny* subjoins, and distinct adaptive sampling algorithms are used for each class due to their different characteristics. H is usually a small constant and thus we assume $H < O(\sqrt{|A|})$. Under this assumption, the original bifocal sampling algorithm can be greatly simplified. The idea is that we only need to sample from D and all the subjoins will be sparse. Due to the fixed, small constant H, the variances among subjoin sizes are also well bounded, which provides good estimation accuracy and confidence.

The IM-DA-Est algorithm, simplified from the bifocal sampling algorithm according to our assumption, is shown in Algorithm 2. The algorithm takes m random points from the $IM_D(D)$ tables, then probes $IM_A(A)$ to calculate each subjoin size. The sizes of subjoins are summed up and scaled to get an estimate of the size of the containment join between A and D. The IM-DA-Est algorithm approximates the real size with high probability due to Theorem 3.

Algorithm 2 IM-DA-Est(A, D, m)

Input:
A and D are ancestor and descendant sets respectively. m is the number of sampled points from $IM_D(D)$ table.

Output:
Return the estimated size of containment join result between A and D.

Description:
1: $est = 0$
2: S^* = a random sample (of size m) from $IM_D(D)$.
3: **for all** point p in S^* **do**
4: x = the number of intervals in $IM_A(A)$ that join with p.
5: $est = est + x$
6: **end for**
7: **return** $est \cdot (|D|/m)$

THEOREM 3. *Let the size of the containment join between A and D be X, and let the estimate given by the IM-DA-Est algorithm be $\hat{X}$. Let $n = |D|$. We have $E[\hat{X}] = X$, with high probability $\hat{X} = \Theta(X) + O(n)$.*

PROOF. (sketch) Let $anc_A(d)$ denote the number of intervals a point $d \in IM_D(D)$ will stab, or the number of ancestors that will join with the element corresponding to d. Let $J(d)$ be the contribution of $d \in IM_D(D)$ to the estimate $\hat{X}$. It is obvious that $\hat{X} = \sum_{d \in IM_D(D)} J(d)$.

$$J(d) = \begin{cases} \frac{n}{m} \cdot anc_A(d) & \text{, if } d \in S^* \\ 0 & \text{, otherwise} \end{cases}$$

where S^* is the set of samples with size equal to m. The expected value of $J(d)$ is

$$\begin{aligned} E[J(d)] &= \frac{n}{m} \cdot anc_A(d) \cdot Pr[d \in S^*] + 0 \cdot Pr[d \notin S^*] \\ &= anc_A(d) \end{aligned}$$

Therefore,

$$\begin{aligned} E[\hat{X}] &= E\left[\sum_{d \in IM_D(D)} J(d)\right] = \sum_{d \in IM_D(D)} E[J(d)] \\ &= \sum_{d \in IM_D(D)} anc_A(d) = X \end{aligned}$$

Moreover, $J(d)$ is a random variable taken from domain $[0, z]$, where $z = n/m \cdot H$. By Hoeffding bounds, $X = \Theta(E[X])$ with probability $e^{-\Theta(E[X/z])}$. Let $m = n^\alpha$, where $0 < \alpha < 1$, then the above probability is high if $X = \Omega(n)$. □

Notice that our confidence of the estimated result is high even when the real result is $O(n)$. This is an improvement upon the previous requirement in [13], which is $O(n \log n)$. This improvement is due to our effort to exploit the unique feature of containment join for XML datasets.

5.2 Position Model Based Adaptive Sampling

In the position model, tables ($PM_A(S)$ and $PM_D(S)$) are generated for each dataset describing its positional information within the workspace under different circumstances. We have shown that the size of the containment join between A and D is equal to the inner product of $PM_A(A)$ and $PM_D(D)$. Hence, we choose to use a simplified bifocal sampling method to do the estimation for the equijoin. The simplification also comes from the observation that the size of subjoins are bounded the constant H, the height of the XML data tree. The PM-Est algorithm is shown in Algorithm 3. It takes m random points from the workspace, then probes both $PM_A(A)$ and $PM_D(D)$ to calculate each subjoin size. The sizes of subjoins are summed up and scaled to get an estimate of the size of containment join between A and D. The PM-Est algorithm approximates the real size with high probability due to Theorem 4.

THEOREM 4. *Let the size of the containment join between A and D be X, and let the estimate given by the PM-Est algorithm be $\hat{X}$. Let $n = c_{max} - c_{min} + 1$. We have $E[\hat{X}] = X$, with high probability $\hat{X} = \Theta(X) + O(n)$.*

Note that the n in the above theorem is the length of the workspace, which is no less than $|A| + |D|$. Each subjoin size in PM-Est method is also bounded by H. Therefore, we expect the performance of this method will be inferior to IM-DA-Est. This is further verified by our experiment result.

5.3 Discussions

Algorithm 3 PM-Est(A, D, m)

Input:

A and D are ancestor and descendant set respectively. m is the number of sampled points from the workspace $[c_{min}, c_{max}]$. w is the length of the workspace, i.e., $c_{max} - c_{max} + 1$.

Output:

Return the estimated result size of the containment join between A and D.

Description:

1: $est = 0$
2: $S^* =$ a random sample (of size m) from the workspace $[c_{min}, c_{max}]$.
3: **for all** value v in S^* **do**
4: $\quad x = PM_A[v]$
5: $\quad y = PM_D[v]$
6: $\quad est = est + x \cdot y$
7: **end for**
8: **return** $est \cdot (w/m)$

5.3.1 Indexes

In both IM-DA-Est and PM-Est methods, we need to calculate the subjoin size for every sample. We consider building auxiliary indexes for the datasets to accelerate such probing operations.

For IM-DA-Est algorithm, we need to answer the following type of queries: *given a point, return the number of intervals that overlap the point.* We consider two kinds of indexes. The first index is the *XR-Tree* [19], which is a dynamic external index structure capable of answering such *stabbing* queries (as well as range queries) efficiently. Efficient containment join algorithms have been developed based on the XR-Tree. Probing the XR-Tree index might not be optimal for the stabbing count query here, as we do not actually need to return all intervals that the query point stabs. However, probing in the XR-Tree will cost only several page accesses in the worst case (when none of the nodes is in the buffer) and, more importantly, such probing helps to load part of the XR-Tree index into the buffer, which will benefit the later containment join processing based on XR-Tree. The other kind of index is based on the observation that the $PM_A(S)[x]$ (if we build PM_A for the ancestor set S) is exactly the number of intervals that overlap point x. Instead of recording the whole $PM_A(S)$, we only need to index every turning point K of $PM_A(S)$ as well as the associated $PM_A(S)[K]$ value via a B$^+$-tree. A turning point K is a point where $PM_A(S)[K] \neq PM_A(S)[K-1]$. It can be shown that there are only $O(|S|)$ such turning points and $PM_A(S)$ values between two adjacent turning points are the same. We refer a B$^+$-tree built on those turning points and their values as a *T-tree*. We can look up the value of $PM_A(S)[q]$ by 1) find $K_i \leq q < K_{i+1}$ in the B$^+$-tree and 2) return the value associated with K_i.

Figure 4 shows an example T-tree constructed for the ancestor dataset in Figure 1(c). If the query point is 6, we can use the B$^+$-tree to search for the largest key that is no larger than the query key. In this example, we find 2 and return the frequency value associated with 2, which is also 2.

For PM-Est algorithm, we need to answer the following type of query: *given a point, return the value associated with the point in either $PM_A(S_1)$ or $PM_D(S_2)$.* The above mentioned T-tree is capable of answering such queries efficiently. Notice that for the probing query on $PM_D(S_2)$, any other index structures built on the *start* field of elements are capable to answer it, e.g. B$^+$-tree, XR-Tree or hash indexes etc. If the query key is found in the index, then its value is 1, otherwise, the value is 0.

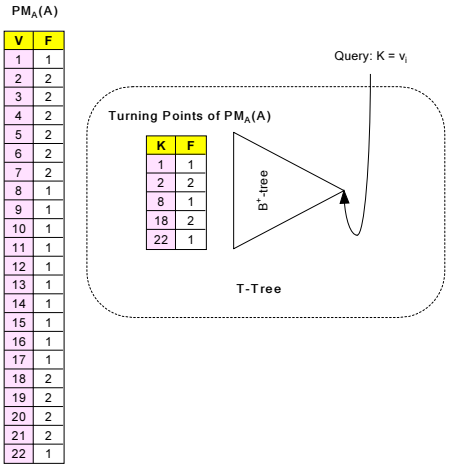


Figure 4: An Example T-Tree

5.3.2 Boosting

The probabilistic boosting method [4] can also be applied to our sampling methods. We can prepare $s_1 \cdot s_2$ independently generated random samples and get their estimated results. We then take the average of every s_1 estimated results and take the median of the s_2 averages as the final result.

6. EXPERIMENTAL STUDY

In this section, we present the results of our experiment study of the newly proposed algorithms. We also give some suggestions on the choice of the estimation algorithms based on the analysis of our experiment results.

6.1 Experiment Setup

We ran experiments on a PC with AMD Atholon 900 MHz CPU, 512M RAM and 40G hard disk. The operating system is Microsoft Windows 2000. We have implemented PL-Hist-Est (PL), IM-DA-Est (IM) and PM-Est algorithms (PM). We also implemented positional histogram/coverage histogram (PH), which is the previous work on the same problem proposed in [28].

We used both synthetic and real datasets in our experiments. We used one real dataset, DBLP and two benchmark datasets, XMARK [26] and XMACH [7]. We will focus on our results using XMARK and DBLP datasets because the results on XMACH datasets are very similar to those on XMARK datasets. DBLP dataset was chosen because (1) It is used in the experiment of PH algorithm [28] and (2) Compared to XMARK, the schema of DBLP data is relatively simple and its data distribution is more regular. XMARK is complex and was also chosen as the experiment dataset in [12] because the XML data tree features both large fanouts and deep nesting in the element structure. For each dataset, we selected a set of predicates, usually the element tag name

Table 2: Statistics for Datasets

(a) Statistics for XMARK

Predicate Name	Node Count	Overlap Property
item	8700	no overlap
desp	17800	no overlap
parlist	8419	N/A
listitem	24544	N/A
text	42314	no overlap
open_auction	4800	no overlap
keyword	28058	no overlap
name	19300	no overlap
mailbox	8700	no overlap
reserve	2355	no overlap
bidder	23521	no overlap
increase	23521	no overlap

(b) Statistics for DBLP

Predicate Name	Node Count	Overlap Property
inproceeding	10350	no overlap
author	21700	no overlap
title	10378	no overlap
cite	3805	no overlap
sup	42	no overlap
label	340	no overlap

(c) Statistics for XMACH

Predicate Name	Node Count	Overlap Property
host	1803	N/A
path	20235	N/A
doc_info	10000	no overlap
doc_id	10000	no overlap
chapter	313	no overlap
section	3338	N/A
head	3651	no overlap
paragraph	8328	no overlap
link	407	no overlap

Table 3: Queries for Datasets

(a) Queries for XMARK

Query	Ancestor	Descendant
Q1	item	name
Q2	item	mailbox
Q3	text	keyword
Q4	desp	parlist
Q5	desp	listitem
Q6	parlist	text
Q7	listitem	keyword
Q8	parlist	listitem
Q9	open_auction	text
Q10	open_auction	reserve
Q11	bidder	increase

(b) Queries for DBLP

Query	Ancestor	Descendant
Q1	inproceeding	author
Q2	inproceeding	title
Q3	inproceeding	cite
Q4	inproceeding	label
Q5	title	sup
Q6	cite	label

(c) Queries for XMACH

Query	Ancestor	Descendant
Q1	host	path
Q2	path	doc_inf
Q3	doc_inf	doc_id
Q4	chapter	section
Q5	section	head
Q6	section	paragraph
Q7	paragraph	link

and derived a set of containment joins accordingly. Table 2 shows the selected predicates and detailed statistics for each dataset. Notice that we also gathered the *overlap property* information for each element set. Such information (especially the information that a dataset has no-overlap property) is key to effectively estimating results of many queries for the PH algorithm. Table 3 shows the set of queries for each dataset.

We use *relative error* as the metric to judge the quality of the estimated result. Formally, it is defined as $\frac{|x-\hat{x}|}{x} \times 100\%$, where x is the real value and $\hat{x}$ is the estimated value. The performance of sampling based algorithms were obtained by averaging over multiple runs under the same setting.

6.2 Overall Performance

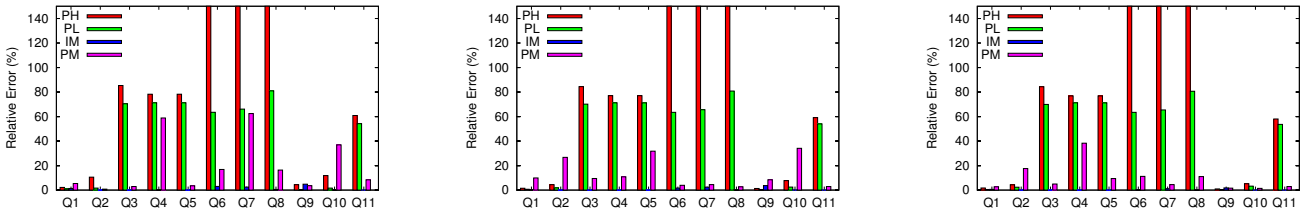
We show the relative estimation error for all the queries on XMARK and DBLP datasets in Figure 5 and Figure 6 respectively. For each dataset, we repeated the experiment for three different settings of different space budgets: 200, 400 and 800 bytes. Note that even 800 bytes is well below 1% of the size of joining element sets. These settings roughly correspond to using 25, 50, 100 buckets for PH histogram method, 10, 20, 40 buckets for PL histogram method and 25, 50, 100 samples for the sampling methods.

From the figures, we can see that under the same amount of space budget, IM algorithm always achieves the best accuracy among the four algorithms. In fact, its relative errors are close to zero for most queries, and up to one order of magnitude smaller than the PH algorithm. Another finding is that sampling based algorithms (IM and PM), in general, achieve lower relative errors than the histogram based algorithms. In all queries in the two datasets, the relative errors of sampling algorithms are below 60% (usually below 20%), while PH algorithm gives very bad result for some queries (37514.55% for Q7 in XMARK).

In the following two subsections, we focus on relative performance of two alternative algorithms for different classes of algorithms (histogram based and sampling based).

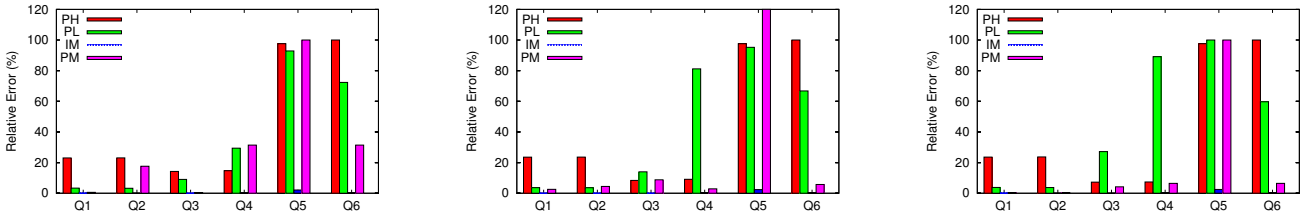
6.3 Histogram Based Algorithms

Figure 7(a) and 7(b) show the relative errors of PH and PL algorithms on the XMARK dataset for different number of buckets respectively. The relative errors of PH algorithm for Q6, Q7 and Q8 are omitted in Figure 7(a) because they are extremely large (1600% to 37500%). Figure 7(c) reexamines



(a) Space Limit = 200 Bytes (b) Space Limit = 400 Bytes (c) Space Limit = 800 Bytes

Figure 5: Overall Performance on XMARK



(a) Space Limit = 200 Bytes (b) Space Limit = 400 Bytes (c) Space Limit = 800 Bytes

Figure 6: Overall Performance on DBLP

their accuracy .

Several observations can be made from the figures:

- Neither histogram based algorithm is sensitive to the number of buckets. In particular, queries having large relative errors cannot be significantly improved by allocating more buckets. This is partly due to the fact that histogram based methods cannot capture correlation between joining datasets and thus cannot reduce the inaccuracy that is due to correlation.
- PL algorithm does not require overlap information of the ancestor set, and is more robust even if such information is available to PH algorithm. Notice that PH algorithm will give extremely erroneous estimation if the no-overlap property is not known beforehand.
- As discussed in the previous sections, PL algorithm bears high risk when the descendant set is relatively sparse, in which case the *cov* value is small and thus MRE value is high. We have listed the average *cov* values for all the queries on the DBLP dataset in Table 4. We can see that *cov* values for Q4, Q5 and Q6 of the DBLP dataset are extremely small (< 0.033) compared to the rest of the queries. Therefore, we expect the real join sizes will be quite sensitive to the data correlation and our PL estimates might be instable. This conjecture is proved in Figure 6, as the relative errors of PL algorithm for Q4, Q5 and Q6 are notably higher than those for Q1, Q2 and Q3. However, PL algorithm still outperforms PH algorithm in all but one query (Q4).

6.4 Sampling Based Algorithms

Figure 8(a) and 8(b) show the relative errors of IM and PM algorithms for different number of samples respectively. Figure 8(c) reexamines their accuracy on the XMARK dataset.

Several observations can be made from the figures:

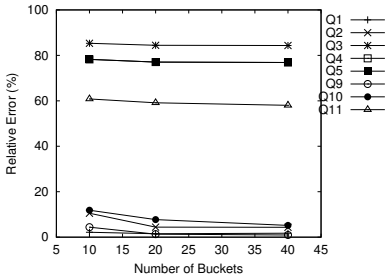
Table 4: Average *cov* Values for Queries on the DBLP Dataset

Query	*Cov*
Q1	2.0520
Q2	0.9814
Q3	0.3598
Q4	0.0322
Q5	0.0003
Q6	0.0201

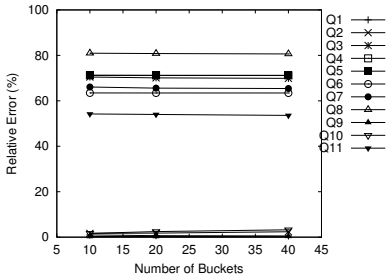
- IM algorithm steadily improves its accuracy with more sample points, while PM algorithm still shows some fluctuation for certain queries. For all the queries on XMARK dataset, IM algorithm has lower relative error rate than PM algorithm. These are mainly because PM algorithm requires more sample points than IM algorithm to achieve a good level of accuracy with high probability. This observation coincides with our prediction on the inferior performance of PM algorithm to the IM algorithm. Nevertheless, the relative error rates of the PM algorithm are still much lower than either of the histogram based methods (PL or PH) (See Figure 5).
- Both algorithms give good estimation under a small space budget. For many queries, IM algorithm can already give very accurate estimation on XMARK dataset with 25 samples (i.e., 200 bytes). With 100 samples (i.e., 800 bytes), IM algorithm can give estimated results within 2% of the real values for all queries and PM algorithm can still give estimated results within 40% of the real values.

6.5 Summary

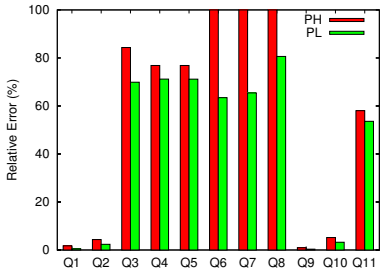
To sum up, IM algorithm achieves the best performance



(a) Tradeoff between Accuracy and Space for PH Algorithm (XMARK Dataset)

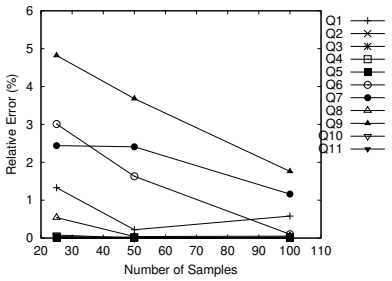


(b) Tradeoff between Accuracy and Space for PL Algorithm (XMARK Dataset)

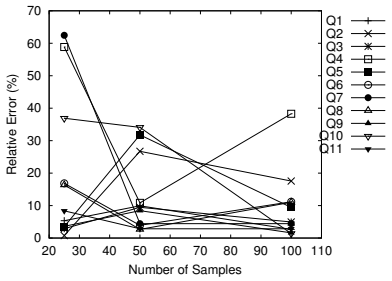


(c) PH vs. PL in terms of Accuracy (XMARK Dataset)

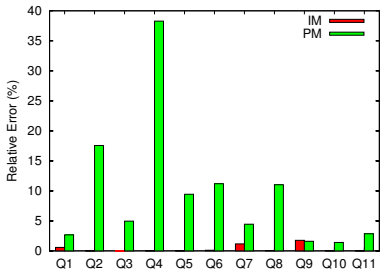
Figure 7: Performance of Histogram Based Algorithms



(a) Tradeoff between Accuracy and Space for IM Algorithm (XMARK Dataset)



(b) Tradeoff between Accuracy and Space for PM Algorithm (XMARK Dataset)



(c) IM vs. PM in terms of Accuracy (XMARK Dataset)

Figure 8: Performance of Sampling Based Algorithms

with a small space requirement and sampling algorithms tend to behave better than histogram based algorithms, especially in the presence of strong correlation between the ancestor and descendant sets.

Therefore, we recommend that a system should use PL histograms (with few buckets only) rather than PH histograms mainly for their robustness against the worst case, if there is no stringent requirement on the accuracy. On the other hand, in case when highly accurate estimation is required, or when the *cov* value is small and MRE value is high or unbounded, the interval model based sampling algorithm is the best choice.

7. CONCLUSIONS

In this paper, we have reported our comprehensive study of size estimation problem for containment joins on XML data. Accurate estimation of the containment join size is a key part for XML query optimization. The unique features of XML data and containment joins impose great challenges on the estimation problem and have rendered previous approaches inapplicable. We proposed two models, the interval model and the position model, in which the original containment join estimation problem are made tractable. Two classes of estimation algorithms, based on histogram and sampling techniques respectively were proposed. The new PL histogram based method makes minimal assumptions about the data distribution and correlation and is shown to be more robust than previous histogram based methods. We also proposed sampling based estimation methods that have provable accuracy with high probability. Various optimizations of the new algorithms are discussed as well.

Extensive experimental evaluation using both real and benchmark datasets has demonstrated the effectiveness of the proposed methods. Our new algorithms usually outperform previous methods up to an order of magnitude and are more robust. We have also observed interesting tradeoffs among various factors for different algorithms.

Our future work includes investigating alternative methods for the containment join size estimation problem. We are interested in applying other existing techniques, such as wavelet approximation and sketch, to this problem.

8. REFERENCES

[1] Ashraf Aboulnaga, Alaa R. Alameldeen, and Jeffrey F. Naughton. Estimating the selectivity of XML path expressions for internet scale applications. In *Proceedings of 27th International Conference on Very Large Data Bases*, pages 591–600, 2001.

[2] Shurug Al-Khalifa, H. V. Jagadish, Nick Koudas, Jignesh M. Patel, Divesh Srivastava, and Yuqing Wu. Structural joins: A primitive for efficient XML query pattern matching. In *Proceedings of the International Conference on Data Engineering*, 2002.

[3] Noga Alon, Phillip B. Gibbons, Yossi Matias, and Mario Szegedy. Tracking join and self-join sizes in limited storage. In *Proceedings of the Eighteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 10–20, 1999.

[4] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 20–29, 1996.

[5] Ning An, Zhen-Yu Yang, and Anand Sivasubramaniam. Selectivity estimation for spatial joins. In *Proceedings of the 17th International Conference on Data Engineering*, pages 368–375. IEEE Computer Society, 2001.

[6] Walid G. Aref and Hanan Samet. A cost model for query optimization using R-Trees. In *Proceedings of the Second ACM Workshop on Advances in Geographic Information Systems*, pages 60–67, 1994.

[7] Timo Böhme and Erhard Rahm. XMach-1: A benchmark for XML data management. In *Proc. of Datenbanksysteme in Büro, Technik und Wissenschaft (BTW)*, pages 264–273, Oldenburg, Germany, 2001.

[8] Zhiyuan Chen, H. V. Jagadish, Flip Korn, Nick Koudas, S. Muthukrishnan, Raymond T. Ng, and Divesh Srivastava. Counting twig matches in a tree. In *Proceedings of the 17th International Conference on Data Engineering*, pages 595–604, 2001.

[9] E. Cohen. Size-estimation framework with applications to transitive closure and reachability. *Journal of Computer System Science*, 55:441–453, 1997.

[10] Alin Dobra, Minos N. Garofalakis, Johannes Gehrke, and Rajeev Rastogi. Processing complex aggregate queries over data streams. In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, 2002.

[11] Christos Faloutsos, Bernhard Seeger, Agma J. M. Traina, and Caetano Traina Jr. Spatial join selectivity using power laws. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, volume 29, pages 177–188, 2000.

[12] Juliana Freire, Jayant R. Haritsa, Maya Ramanath, Prasan Roy, and Jérôme Siméon:. Statix: making XML count. In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, pages 181–191, 2002.

[13] Sumit Ganguly, Phillip B. Gibbons, Yossi Matias, and Abraham Silberschatz. Bifocal sampling for skew-resistant join size estimation. In H. V. Jagadish and Inderpal Singh Mumick, editors, *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, pages 271–281, 1996.

[14] Peter J. Haas, Jeffrey F. Naughton, S. Seshadri, and Arun N. Swami. Fixed-precision estimation of join selectivity. In *Proceedings of the Twelfth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 190–201, 1993.

[15] Banchong Harangsri, John Shepherd, and Anne H. H. Ngu. Selectivity estimation for joins using systematic sampling. In *DEXA Workshop 1997*, pages 384–389, 1997.

[16] Wen-Chi Hou, Gultekin Özsoyoglu, and Baldeo K. Taneja. Statistical estimators for relational algebra expressions. In *Proceedings of the Seventh ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 276–287, 1988.

[17] Wen-Chi Hou, Gultekin Özsoyoglu, and Baldeo K. Taneja. Processing aggregate relational queries with hard time constraints. In James Clifford, Bruce G. Lindsay, and David Maier, editors, *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data*, pages 68–77, 1989.

[18] Yannis E. Ioannidis. Universality of serial histograms. In Rakesh Agrawal, Seán Baker, and David A. Bell, editors, *Proceedings of the 19th International Conference on Very Large Data Bases*, pages 256–267, 1993.

[19] Haifeng Jiang, Hongjun Lu, Wei Wang, and Beng Chin Ooi. XR-Tree: Indexing XML data for efficient structural joins. In *Proceedings of the 19th International Conference on Data Engineering*, 2003.

[20] Quanzhong Li and Bongki Moon. Indexing and querying XML data for regular path expressions. In *Proceedings of the 27th International Conference on Very Large Data Bases*, 2001.

[21] Richard J. Lipton and Jeffrey F. Naughton. Query size estimation by adaptive sampling. In *Proceedings of the Ninth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 40–46, 1990.

[22] James F. Lynch. Analysis and application of adaptive sampling. In *Proceedings of the nineteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 260–267, 2000.

[23] Nikos Mamoulis and Dimitris Papadias. Selectivity estimation of complex spatial queries. In *Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases*, pages 155–174, 2001.

[24] Neoklis Polyzotis and Minos N. Garofalakis. Statistical synopses for graph-structured XML databases. In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, pages 358–369, 2002.

[25] Viswanath Poosala, Yannis E. Ioannidis, Peter J. Haas, and Eugene J. Shekita. Improved histograms for selectivity estimation of range predicates. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, pages 294–305, 1996.

[26] Albrecht Schmidt, Florian Waas, Martin Kersten, Daniela Florescu, Loana Manolescu, Michael J. Carey, and Ralph Busse. The XML benchmark project. Technical report, CWI, 2001.

[27] Chengyu Sun, Divyakant Agrawal, and Amr El Abbadi. Selectivity estimation for spatial joins with geometric selections. In *Proceedings of the 8th International Conference on Extending Database Technology*, pages 609–626, 2002.

[28] Yuqing Wu, Jignesh M. Patel, and H. V. Jagadish. Estimating answer sizes for xml queries. In *8th International Conference on Extending Database Technology*, pages 590–608, 2002.

[29] Chun Zhang, Jeffrey F. Naughton, David J. DeWitt, Qiong Luo, and Guy M. Lohman. On supporting containment queries in relational database management systems. In *Proceedings of the 27th ACM SIGMOD International Conference on Management of Data*, 2001.