

Index-Based Approximate XML Joins

Sudipto Guha

University of Pennsylvania
sudipto@cis.upenn.edu

Nick Koudas

AT&T Labs-Research
koudas@research.att.com

Divesh Srivastava

AT&T Labs-Research
divesh@research.att.com

Ting Yu

University of Illinois
tingyu@cs.uiuc.edu

Abstract

XML data integration tools are facing a variety of challenges for their efficient and effective operation. Among these is the requirement to handle a variety of inconsistencies or mistakes present in the data sets.

In this paper we study the problem of integrating XML data sources through index assisted join operations, using notions of approximate match in the structure and content of XML documents as the join predicate. We show how a well known and widely deployed index structure, namely the R-tree, can be adopted to improve the performance of such operations. We propose novel search and join algorithms for R-trees adopted to index XML document collections. We also propose novel optimization objectives for R-tree construction, making R-trees better suited for this application.

1 Introduction

Currently XML and related technologies are increasingly recognized in a variety of industries as the means for data exchange and integration. XML based data integration tools however, are facing a variety of challenges for their efficient and effective operation. A major challenge stems from a requirement, intrinsic to any data integration problem, namely the ability to cope with a multitude of inconsistencies present in the data sets to be integrated.

Such inconsistencies are relevant in the XML domain as well. First, literals of string type constitute typical content of XML elements. Second, two documents (having the same or different DTD's) might convey exactly or approximately the same information but have differences in their underlying structure. The flexible structure of XML documents makes it imperative to consider approximate match in structure, in addition to content, for effectively quantifying approximate match between pairs of XML documents.

A fundamental requirement towards the goal of automatically identifying structural differences in XML, is the existence of predicates that quantify the distance or closeness

of a pair of documents. We refer to such predicates as *approximate XML match* predicates. Various XML integration and cleansing problems can be formulated using correlations between XML data sources realized as join operations making use of approximate XML match predicates as join conditions. We refer to these join operations as *approximate XML joins*. In analogy to related join operations known from relational databases [GIJ⁺01], the study of efficient algorithms for executing such operations is a pressing concern. Indexing has been widely utilized by join algorithms in the relational world as well as in spatial and multidimensional applications, for improved performance. In this line of work, we investigate the use of indices to improve the performance of approximate XML join operations and we make the following contributions:

- Rather than proposing new indices tailored to this specific problem, we opt for a design that is capable of incorporating a very large class of approximate XML match predicates utilizing well known and widely accepted index structures.
- In particular, we instantiate our design by showing how R-trees [Gut84], which is one of the most popular index structures, can be used to improve the performance of approximate XML join operations for a wide class of approximate XML match predicates, namely *metric* approximate XML match predicates.
- We optimize the indexing structures applied to the approximate XML join problem, by proposing modifications to the basic R-tree construction algorithms that can improve substantially the performance of R-trees in this context. Moreover, we propose R-tree based search and join algorithms for the approximate XML join problem.

Our complete technical results are available elsewhere [GKSY]. This brief technical note is organized as follows: In section 2 we present background material on approximate XML joins. Section 3 formally defines the problem we address. Section 4 shows how R-trees can be adapted for

this problem and identifies basic properties that lead to the design of efficient algorithms for approximate XML joins using R-trees.

2 Background on Approximate XML Joins

Given two XML data sources S_1 and S_2 , let DIST be a predicate that assesses the distance between two documents $D_1 \in S_1$ and $D_2 \in S_2$ for a suitably defined notion of distance. A variety of choices exist for predicate DIST , for example tree edit distance¹ [ZS97], tree difference distance [MACM01] etc. Such predicates quantify closeness of documents using a series of operations transforming one document to the other.

We choose to make our presentation generic, without restricting it to a particular distance predicate. We require however that the predicate of choice be a metric.² This requirement makes the distance predicate and its interpretation very intuitive, since the predicate is symmetric, non negative and obeys the triangle inequality.

For two XML data sources, S_1, S_2 , the approximate XML join problem [GJK⁺02], returns in the output all pairs of documents (D_1, D_2) , $D_1 \in S_1, D_2 \in S_2$ such that $\text{DIST}(D_1, D_2) \leq k$, for a user-specified distance threshold k . Various solutions for this problem are derived in [GJK⁺02] using a mapping of XML documents to vectors obtained via the notion of *Reference Sets*. A reference set RS is a set of XML documents sampled from $S_1 \cup S_2$. Let $R_1, R_2, \dots, R_d, d = |RS|$ be an arbitrary ordering of the elements of RS . Using such an ordering every XML document $D \in S_1 \cup S_2$ can be mapped into a d -dimensional vector V_D by populating the i -th coordinate of V_D with the distance of D to the i -th document in the reference set, namely $V_D[i] = \text{DIST}(D, R_i), 1 \leq i \leq d$. We refer to such vectors as *XML Document Distance Vectors* (XDD Vectors). Since DIST is a metric, one can reason about the distance between two XML documents by manipulating the XDD vectors corresponding to the documents. For all $D_i, D_j, 1 \leq i \leq |S_1|, 1 \leq j \leq |S_2|$, and a user specified threshold k , two observations are immediate by application of the triangle inequality on the XDD vectors:

Observation 1 *if $\max_{1 \leq p \leq d} |V_{D_i}[p] - V_{D_j}[p]| > k$ then $\text{DIST}(D_i, D_j) > k$.*

Observation 2 *if $\min_{1 \leq p \leq d} |V_{D_i}[p] + V_{D_j}[p]| \leq k$ then $\text{DIST}(D_i, D_j) \leq k$.*

¹Defined as the minimum number of operations (node insert, node delete and relabeling of a node) to transform one document to the other.

²Namely, for any documents D_i, D_j, D_k , DIST has the the following properties (a) $\text{DIST}(D_i, D_j) = \text{DIST}(D_j, D_i)$ (b) $\text{DIST}(D_i, D_j) > 0$ (when $D_i \neq D_j$, otherwise $\text{DIST}(D_i, D_i) = 0$) (c) $|\text{DIST}(D_i, D_k) - \text{DIST}(D_k, D_j)| \leq \text{DIST}(D_i, D_j) \leq \text{DIST}(D_i, D_k) + \text{DIST}(D_k, D_j)$

Pruning efficiency is tightly related to the accuracy of the assessment of distance between pairs of documents is, using the triangle inequality. The choice of the reference set affects this accuracy. In [GJK⁺02] efficient sampling based techniques were proposed to identify reference sets with good pruning properties and also determine the optimum size of the reference set.

3 Our Problem Statement

Indexing has been widely utilized to speed up the performance of various operators in a variety of data management scenarios. In this paper, we investigate indexing in the context of approximate XML join operations. This gives rise to the main problem addressed in this paper which we define formally as:

Problem 1 (Index Based Approximate XML Joins)

Given two XML data sources S_1, S_2 , a metric distance predicate DIST assessing the distance between pairs of XML documents and a user defined threshold distance k , utilize indexing structures in order to report efficiently all pairs of documents $(D_1, D_2) \in S_1 \times S_2$, such that $\text{DIST}(D_1, D_2) \leq k$.

4 Indexing XDD Vectors Using R-trees

Let S_1, S_2 be two XML document sources of interest and $RS \subset S_1 \cup S_2$ be a reference set of size $d = |RS|$. The particular set RS of optimum size d can be obtained using the sampling based algorithms presented in [GJK⁺02]. With a single scan in each source one can produce the XDD vectors corresponding to each XML document of each source. Let $V_{D_j}^{S_i}, 1 \leq j \leq |S_i|, 1 \leq i \leq 2$ denote the XDD vectors so produced.

R-trees fall in the *entity grouping* category of multi-dimensional access structures, namely they organize the underlying space by grouping a number of entities (objects) together, deriving suitable descriptors of the portion of space occupied by these entities, commonly their *minimum bounding rectangle* (MBR). R-tree structures are suitable for indexing collections of multidimensional points. The collection of XDD vectors can be viewed as a collection of points in d -dimensional space and an R-tree can be constructed on this collection using traditional R-tree construction algorithms [Gut84].

Assume a collection of m XDD vectors corresponding to documents $D_j, 1 \leq j \leq m$ from a source S is stored in the same leaf page of an R-tree. Let r represent the MBR of that collection. The MBR is represented as pairs $((l_1, u_1), \dots, (l_d, u_d))$, where $l_i, u_i, 1 \leq i \leq d$ are r 's coordinates in dimension i . The first coordinate in each pair is $l_i = \min_{1 \leq j \leq m} V_{D_j}^S[i]$ and the second coordinate is

$u_i = \max_{1 \leq j \leq m} V_{D_j}^S[i]$; they represent the minimum and maximum of distances to the i -th document in the reference set RS , for documents $D_j, 1 \leq j \leq m$, from source S . For an MBR r we use $lower_i(r)$ to represent l_i and $upper_i(r)$ to represent u_i . An R-tree can be constructed using such MBRs.

4.1 R-tree Construction on XDD Vectors

Various optimization objectives (eg., *minVolume*) during R-tree construction have been formulated in folklore uses of R-trees [Gut84], since the *combination of all dimensions* is of interest when formulating queries. The main utility of XDD vectors however, is their ability to obtain a bound on the DIST-distance between two XML documents. XDD vectors enable utilization of distances to elements of the reference set *in isolation* (i.e., examining each dimension individually) seeking a useful bound on DIST in order to facilitate pruning with respect to the distance threshold. This fundamental difference in the way index page information is used in the case of range search on XDD vectors suggests a different optimization criterion during R-tree construction. A way to make pruning more effective is to enforce a tighter approximation to DIST using the triangle inequality. The values of $upper_j, lower_j$ for dimension j are crucial for pruning. In particular, the smaller $upper_j$ is, the tighter the upper bound becomes. Similarly, the larger $lower_j$ is or the smaller $upper_j$ is the tighter the lower bound becomes. This effectively means that the sides of the MBRs along a dimension j should have as small a length as possible. One can achieve this by assuring that the MBRs of the two new index nodes resulting from a split of an index node during R-tree construction have the *minimum possible length* along at least one side. One side is sufficient, as at least one successful application of the triangle inequality to a reference set point (out of d points in total) with respect to the query distance threshold is necessary and sufficient for pruning. We refer to this new optimization objective as *minSide*.

4.2 R-tree Join on XDD Vector Collections

Given R-tree structures T_1 and T_2 constructed on the XDD vectors obtained from XML data sources S_1 and S_2 , we have designed a join algorithm that utilizes both R-trees providing an efficient solution to our original problem. The algorithm performs a synchronized depth-first traversal of both trees utilizing the triangle inequality on MBRs of nodes of T_1 and T_2 to reason and effectively prune XDD vector collections *in bulk* from both trees.

At the index nodes of both trees, pairs of MBRs, are considered for pruning using the triangle inequality. When considering the triangle inequality on a pair of MBRs one has to

take into account the set of XDD vectors represented by the MBRs. In particular, the upper bound pruning condition on the actual DIST-distance between any pair of vectors (documents) represented by MBRs mbr_i, mbr_l becomes:

$$\exists j \text{ } upper_j(mbr_i) + upper_j(mbr_l) \leq k, 1 \leq j \leq d \quad (1)$$

and the lower bound condition:

$$\exists j \text{ } lower_j(mbr_i) - upper_j(mbr_l) > k, \quad (2)$$

$$\text{if } lower_j(mbr_i) > upper_j(mbr_l)$$

or

$$lower_j(mbr_l) - upper_j(mbr_i) > k, \quad (3)$$

$$\text{if } lower_j(mbr_l) > upper_j(mbr_i), 1 \leq j \leq d$$

If there exists a dimension $j, 1 \leq j \leq d$ such that equation 1 becomes true, then for that pair of MBRs, all pairs of XDD vectors in the leaf pages of the subtrees rooted at mbr_i, mbr_j are included in the result; if a dimension makes equation 2 true, the pairs of XDD vectors in the leaf pages of the corresponding subtrees can be safely ignored for this specific pair of MBRs. Note that equation 2 is suitably modified to account for the partial ordering of the projection of two MBRs in the j -th dimension.

5 Conclusions

In this technical note, we have identified basic properties that lead towards the design of efficient R-tree based algorithms for searching and joining XML data sources approximately. Complete details about our techniques are available elsewhere [GKSY].

References

- [GIJ⁺01] L. Gravano, P. Ipeirotis, H.V. Jagadish, N. Koudas, S. Muthukrishnan, and D. Srivastava. Approximate strings joins in a database (almost) for free. *Proceedings of VLDB*, 2001.
- [GJK⁺02] S. Guha, H. V. Jagadish, N. Koudas, D. Srivastava, and T. Yu. Approximate XML Joins. *Proceedings of ACM SIGMOD*, June 2002.
- [GKSY] S. Guha, N. Koudas, D. Srivastava, and T. Yu. Index-Based Approximate XML Joins. *ATT Labs Research Technical Report*.
- [Gut84] A. Guttman. R-trees : A Dynamic Index Structure for Spatial Searching. *Proceedings of ACM SIGMOD*, pages 47–57, June 1984.
- [MACM01] A. Marian, S. Abideboul, G. Cobena, and L. Mignet. Change Centric Management of Versions in an XML Warehouse. *Proceedings of VLDB, Rome Italy*, 2001.
- [ZS97] K. Zhang and D. Shasha. Tree Pattern Matching. *Pattern Matching Algorithms, Apostolico and Galil Editors, Oxford University Press*, 1997.