Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 5, May 2015, pg.1101 – 1107

RESEARCH ARTICLE

Mapping of XML to Sparse Table Using Relational Engine for P2P Databases: Implementation

SAURABH VYAS[*], SHRUTI KOLTE, SONALI BODKHE

PG SCHOLAR, Department of CSE

PROFESSOR, Department of CSE

HOD, Department of CSE

GHRAET, NAGPUR, INDIA

saurabh.vyas7890@gmail.com shrutikolte32@gmail.com sonali.mahure@gmail.com

***Abstract-** In this module, user has to give the query for the further propose and to obtain the optimized query. The wide spread use of the Web has originated several new data management problems, such as extracting data from Web pages and making databases accessible from Web browsers, and has renewed the interest in problems that had appeared before in other contexts, such as querying graphs, semi structured data and structured documents. Several systems and languages have been proposed for solving each of these Web-data management problems, but none of these systems addresses all the problems from a unified perspective. XML is commonly supported by SQL database systems. However, existing mappings of XML to tables can only deliver satisfactory query performance for limited use cases. Most of the existing link analysis algorithms treat a web page as a single node in the web graph. However, in most cases, a web page contains multiple semantics and hence the web page might not be considered as the atomic node. In this paper, the web page is partitioned into blocks using the vision-based page segmentation algorithm. Comparing to other existing techniques, our approach is independent to underlying documentation representation such as HTML and works well even when the HTML structure is far different from layout structure. Genetic Algorithm can be used in this query technique.*

***Keywords:** VIPS, DOM Trees, Rewriter, Data Region Extraction.*

I. INTRODUCTION

PROBLEM STATEMENT

This paper focuses on the problem of automatically extracting data records that are encoded in the query result pages generated by web databases. In general, a query result page contains not only the actual data, but also other information, such as navigational panels, advertisements, comments, information about hosting sites, and so on. The major issues in Web data extraction and describe an XML-based methodology whose goal extends far beyond simple "screen scraping." An ideal data extraction process is able to digest target Web databases that are visible only as HTML pages, and create a local, identical replica of those databases as a result. What is needed in this process is much more than a Web crawler and set of Web site wrappers. A comprehensive data extraction process needs to deal with such roadblocks such as session identifiers, HTML forms, and client-side JavaScript, and data integration problems such as incompatible datasets and vocabularies, and missing and conflicting data. Proper data extraction also requires a solid data validation and error recovery service to handle data extraction failures, which are unavoidable [1].

EXISTING SYSTEM

In general, XML query languages contain unconventional constructs that bring more challenges. With XML schemas, the evaluation of such constructs can be greatly simplified: queries can be pre-evaluated against XML schemas. For this reason, schemas are even more important for query processing in XML than in RDBMSs, as some XML query constructs hide schema information and put an additional burden on query processing. Though normalization mappings provide schema- and normalization-based optimizations, they can only support a Limited number of use cases. In particular, we identify three use cases that are widely observed in enterprise applications but cannot be supported by normalization mappings efficiently. Flexible schema. XML's schema-less property is one of its major advantages over other schema-first data models. In normalization mappings, schema design of primary-foreign keys requires full knowledge of the XML schema, and cannot be applied to this use case. While the XML schema can be summarized from the initial document set , later updates may cause expensive schema evolution and table repartitioning. Many small XML documents. Enterprise applications often use many small XML documents rather than a few large ones. Normalization mappings typically shred one XML document over many tables, which causes fragmentation of small XML documents. As a consequence, the cost of XML reconstruction can be high. Many optional elements/attributes. By data normalization, XML attributes are stored as additional columns in the tables of their parent elements. This strategy, however, results in many sparse tables and incurs storage. Schema and normalization have a profound impact on query processing in database systems. While XML is an ideal model for applications with flexible schemas, conventional query processing for such XML data loses all schema and normalization-based optimizations. In this paper, we proposed a novel mapping that resembles normalization for XML with flexible schemas. The key is to retain normalization through logical sub-tables in one wide sparse table, which is physically represented by interpreted storage. Logical sub-tables are maintained as metadata—the annotated DG—to reduce the cost of schema evolution. At query compilation time, the annotated DG is used to eliminate joins. Experimental results demonstrate significant improvements of query performance under the new mapping.[1][2]

DISADVANTAGES

- Extract method extracts only the actual data not extract other information's such as navigational panels, advertisements. Comments and so on.
- Does not remove irrelevant information from the query result page.
- It is not straightforward to determine which index will have the best performance for each query. In fact, index selection is a difficult problem for query processing of the node-encoding mapping.

PROPOSED SYSTEM

Converting an XML encoded dataset associated with each identified hierarchical structure, wherein for each identified hierarchical structure said converting step includes the further steps of: determining a node element set for said identified hierarchical structure of said XML encoded dataset, wherein each node element in said node element set is a discrete level of said identified hierarchical structure of said dataset; determining one or more nodes of said XML encoded dataset each node being an instance of a node element; allocating to each node a unique node identifier; and generating an SQL node table containing one or more records, each record corresponding to a respective one of said allocated node identifiers.[9]

According to a second aspect of the invention, there is provided an apparatus for converting an XML encoded dataset into a minimal set of SQL tables, the apparatus including: a device for identifying at least one hierarchical structure in the XML encoded dataset; and a device for converting an XML encoded dataset associated with each identified hierarchical structure, the device including:

- A device for determining a node element set for the identified hierarchical structure of the XML encoded dataset, wherein each node element in the node element set is a discrete level of the identified hierarchical structure of the dataset;
- A device for determining one or more nodes of the XML encoded dataset, each node being an instance of a node element;
- A device for allocating to each node a unique node identifier; and
- A device for generating an SQL node table containing one or more records, each record corresponding to a respective one of the allocated node identifiers

According to another aspect of the invention there is provided a computer program product including a computer readable medium having recorded thereon a computer program for implementing the method described above.

For each round, the DOM tree with its visual information corresponded to the current block (page for the first round) is obtained from a web browser. Then, from the root node(s) of the DOM tree (e.g. DOM node), the block extraction process is started to extract blocks from the DOM tree based on visual cues. Every DOM node is checked to judge whether it forms a single block or not. If not its children will be processed in the same way. Separators among these blocks are identified and the weight of a separator is set based on properties of its neighbouring blocks. The layout hierarchy was constructed based on these separators. After constructing the layout hierarchy of the current round, each leaf node of the content structure is checked to see whether or not it meets the granularity requirement. If not, this leaf node will be treated as a sub-page and will be further segmented similarly. After all blocks are processed, the final vision-based content structure for the webpage is out putted. In the above example, we finally obtain a vision-based content structure tree[6].

ADVANTAGES

- Accurate data extraction method.
- New techniques are proposed to handle the situation when data records are not contiguous.

- Pair wise alignment is proposed for both tag structure similarity and data value similarity.
- Very Efficient. It is Flexible.

II. ALGORITHM AND DESCRIPTION

A. VIPS

In this paper, we propose VIPS (Vision-based Page Segmentation) algorithm to extract the semantic structure for a webpage. Such semantic structure is a hierarchical structure in which each node will correspond to a block. Each no de will be assigned a value (Degree of Coherence) to indicate how co he rent of the content in the block based on visual perception. The VIPS algorithm makes full use of page layout feature: it first extracts all the suitable blocks from the html DOM tree, then it tries to find the separators between these extracted blocks. Here, separators denote the horizontal or vertical lines in a webpage that visually cross with no blocks. Finally, based on these separators, the semantic structure for the webpage is constructed. VIPS algorithm employs a top-down approach, which is very effective.[6]

B. Mapping

The Data Region Extraction module identifies all possible data regions, which usually contain dynamically generated data, top down starting from the root node. [7]We first assume that some child sub trees of the same parent node form similar data records, which assemble a data region. Many query result pages some additional item that explains the data records, such as a recommendation or comment, often separates similar data records. Hence, we propose a new method to handle non-contiguous data regions so that it can be applied to more web databases. Finding the data segmentation, data region identification.[7][2][1]

III. METHODOLOGIES

Methodologies are the process of retrieving exact information what the user looking for. Here are the modules and explanation for the modules given for the effective function of the system.

Modules

A. *Pre-processing*
B. *Query Initialization*
C. *Rewriter*
D. *DOM Tree Construction*
E. *Data Region Extraction*

Modules Description

A. Pre-processing

Data Preparation and filtering steps can take considerable amount of processing time. Includes cleaning, normalization, transformation, feature extraction and selection etc. Analyzing data that has not been carefully screened for such problems can produce misleading results. Thus, the representation and quality of data is first and foremost before running an analysis.[1][2]

B. Query Initialization

In this module, user has to give the query for the further propose and to obtain the optimized query. Here we consider the static tables and data's. The table names and attributes are prefix, name, sex, dob, addr, city, zip, mailid, ph, date, Age, problem, Height, Weight, BP_Before, BP_After.

C. Rewriter

In this module, have to rewrite the user given query into the representation format based on the selection, project and joint. Based on this rewrites query only have to prepare the execution plans. The selection is represented by sigma then the projection is represented by pi then the joint is represented by ><. [4][3]

D. DOM Tree Construction

Get the Input Query Result Page from the User. Given a query result page, the DOM Tree Construction module first constructs a DOM tree for the page rooted in the <HTML> tag. Each node represents a tag in the HTML page and its children are tags enclosed inside it. Each internal node n of the tag tree has a tag string tsn, which includes the tags of n and all tags of n's descendants, and a tag path tpn, which includes the tags from the root to n.[5][6]

E. Data Region Extraction

The Data Region Extraction module identifies all possible data regions, which usually contain dynamically generated data, top down starting from the root node. We first assume that some child sub trees of the same parent node form similar data records, which assemble a data region. Many query result pages some additional item that explains the data records, such as a recommendation or comment, often separates similar data records. Hence, we propose a new method to handle non-contiguous data regions so that it can be applied to more web databases. The data region Extraction algorithm discovers data regions in a top-down manner. Starting from the root of the query result page DOM tree, the data region identification algorithm is applied to a node n and recursively to its children n_i, i =1 . . .m. Compute the similarity sim_{ij} of each pair of nodes n_i and n_j, i , j = 1 . . .m and i # j, using the node similarity calculation method. The data region identification algorithm is recursively applied to the children of n_i only if it does not have any similar siblings. Segment the data region into data records using the record segmentation algorithm.[7]

IV. System Architecture

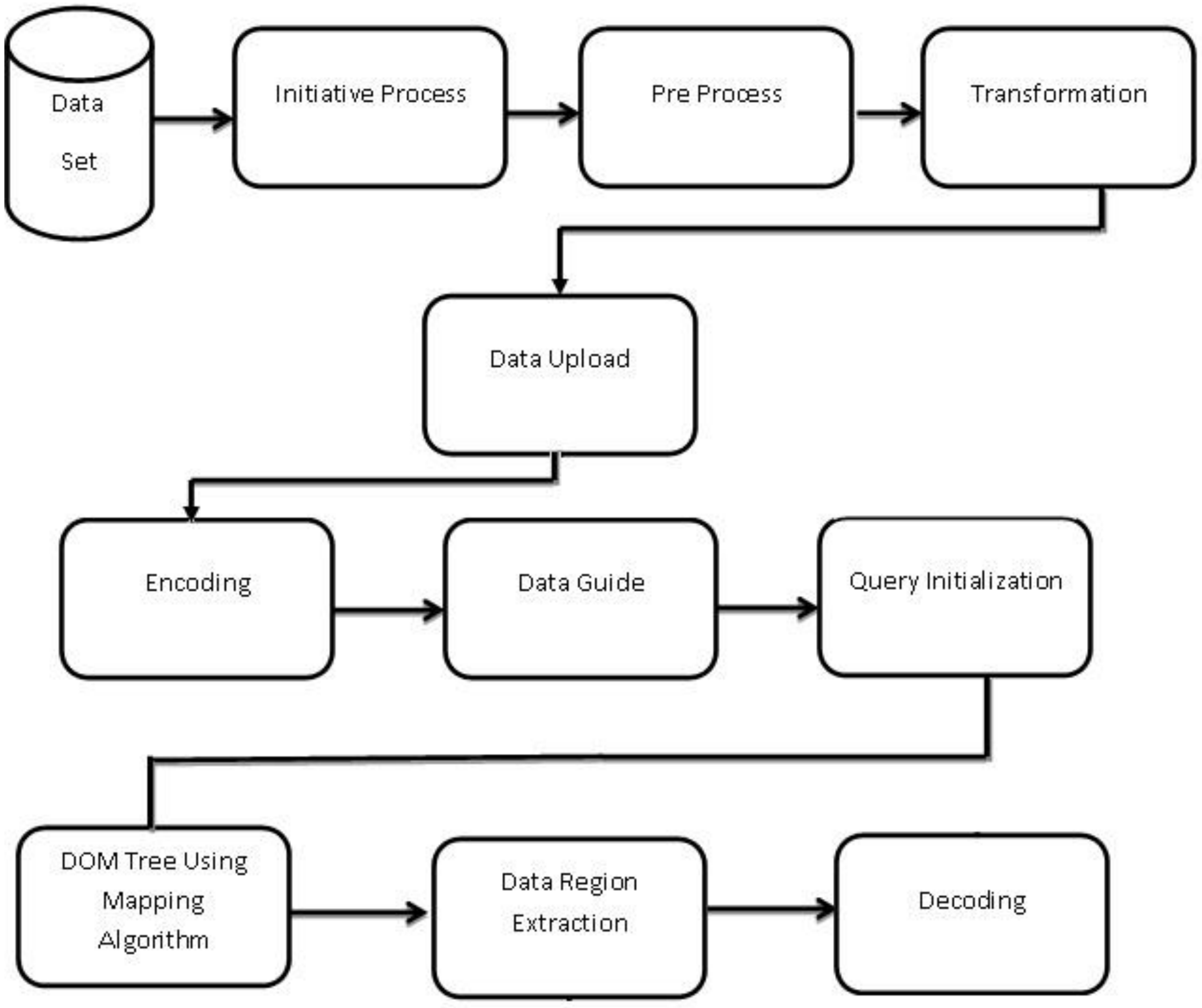

Fig. 1 Implementation Flow of the system.

V. Conclusion

In this paper a new approach for extracting web content structure based on visual representation was proposed. The produced web content structure is very helpful for applications such as web adaptation, information retrieval and information extraction. By identifying the logic relationship of web content based on visually out information, web content structure can effectively represent the semantic structure of the webpage. An automatic top- down, tag-tree independent and scalable algorithm to detect web content structure was presented. It simulates how a user understands the layout structure of a web page based on its visual representation. Compared with traditional DOM based segmentation method, our scheme utilizes useful visual cues to obtain a better partition of a page at the semantic level. It is also independent of physical realization and works well even when the physical structure is far different from visual presentation. The algorithm is evaluated manually on a large dataset, and also used for selecting good expansion terms in a pseudo-relevance feedback process in web information retrieval, both of which achieve very satisfactory performance.

it would be useful to study for which combinations of documents and workload this mapping is and is not well suited. Even more useful would be a system that automatically selects a mapping and translate queries with minimum user input (sample documents, sample query workload). The sparse mapping can have many variations. For example, instead of mapping an entire XML document, the system could map only parts of the document that are queried.

REFERENCES

[1] L. J. Chen et al. "Mapping XML to a wide sparse table," in IEEE 28th ICDE, Washington, DC, USA, 2012.

[2] Saurabh Vyas et al. "A revierw paper for mapping of XML Data to Relational Table" in International Journal of Computer Science and mobile Computing, vol. 3 Issue. 12 December 2014.

[3] H. Georgiadis and V. Vassalos, "XPath on steroids: Exploiting relational engines for XPath performance," in *Proc. SIGMOD*, Beijing, China, 2007.

[4] M. J. Carey, L. Ling, M. Nicola, and L. Shao, "EXRT: Towards a simple benchmark for XML readiness testing," in *Proc. 2nd TPCTC*, Singapore, 2010.

[5] Z. Chen, H. V. Jagadish, L. V. S. Lakshmanan, and S. Paparizos, "From tree patterns to generalized tree patterns: On efficient evaluation of XQuery," in *Proc. 29th Int. Conf. VLDB*, Berlin, Germany, 2003.

[6] P. YESURAJU* et al, "A LANGUAGE INDEPENDENT WEB DATA EXTRACTION USING VISION BASED PAGE SEGMENTATION ALGORITHM," IJRET., Volume: 2 Issue: 4, pp. 635–639, 2012103.

[7] G. H. L. Fletcher, D. V. Gucht, Y. Wu, M. Gyssens, S. Brenes, and J. Paredaens, "A methodology for coupling fragments of XPath with structural indexes for XML documents," *Inf. Syst.*, vol. 34, no. 7, pp. 657–670, Nov. 2009.

[8] P. A. Boncz et al "MonetDB/XQuery: A fast XQuery processor powered by a relational engine," in Proc. SIGMOD, Chicago, IL, USA, 2006.

[9] S. Acharya, et al., "Relational support for flexible schema scenarios," Proc. VLDB, vol. 1, no. 2, pp. 1289–1300, Aug. 2008.

[10] Amjad Qtaish, Kamsuriah Ahmad, "Model-Mapping Approaches for Storing and Querying XML Documents in Relational Database.

[11] A. Arion, A. Bonifati, et al, "Efficient query evaluation over compressed XML data," in Proc. EDBT, Heraklion, Greece, 2004.

[12] K. S. Beyer et al. "System RX: One part relational, one part XML," in Proc. SIGMOD, Baltimore, MD, USA, 2005.

[13] P. Bohannon, J. Freire, P. Roy, and J. Sim´eon. From XML schema to relations: A cost-based approach to XML storage. In Proceedings of the International Conference on Data Engineering (ICDE), pages 64–75, 2002.

[14] P. Bohannon, J. Freire, P. Roy, and J. Siméon, "From XML schema to relations: A cost-based approach to XML storage," in Proc. 18th ICDE, San Jose, CA, USA, 2002.

[15] S. Amer-Yahia, F. Du, and J. Freire. A generic and flexible framework for mapping XML documents into relations.

[16] Technical report, OGI/OHSU, 2004 [15] Z. Chen, H. V. Jagadish, L. V. S. Lakshmanan, and S. Paparizos, "From tree patterns to generalized tree patterns: On efficient evaluation of XQuery," in Proc. 29th Int. Conf. VLDB, Berlin, Germany, 2003. 1414 IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 26, NO. 6, JUNE 2014

[17] D. DeHaan, D. Toman, M. P. Consens, and M. T. Özsu, "A comprehensive XQuery to SQL translation using dynamic interval encoding," in Proc. SIGMOD, San Diego, CA, USA, 2003.