Extracting XML Schema from Multiple Implicit XML Documents Based on Inductive Reasoning

Masaya Eki, Tadachika Ozono and Toramatsu Shintani
Nagoya Insutitute of Technology
Gokiso-cho, Showa-ku, Nagoya, Aichi, Japan
{eki, ozono, tora}@ics.nitech.ac.jp

ABSTRACT

We propose a method of classifying XML documents and extracting XML schema from XML by inductive inference based on constraint logic programming. The goal of this work is to type a large collection of XML approximately but efficiently. This can also process XML code written in a different schema or even code which is schema-less. Our approach is intended to achieve identification based on the syntax and semantics of the XML documents by information extraction using ontology, and to support retrieval and data management. Our approach has three steps. The first step is XML to predicates, the second step is to compare predicates and classifies structures which represent similar meanings in different structures , and the last step is predicates to rules by using ontology and to maintain XML Schema. We evaluate similarity of data type and data range by using an ontology dictionary, and XML Schema is made from results of second and last step.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Experimentation

Keywords

XML, predicate logic, inductive reasoning

1. INTRODUCTION

The Semantic Web is an evolving extension of the World Wide Web (WWW) in which web contents can be expressed not only in natural language but also in a form that can be understood, interpreted, and used by software agents, thereby permitting such agents to find, share, and integrate information more easily.

In general, semistructured data are characterized by the lack of fixed and rigid schema, although, typically, the data have some implicit structure. We propose a method of classifying XML documents and extracting XML schema from XML documents through inductive inference based on constraint logic programming; our goal is to be able to type a large collection of XML approximately but efficiently.

2. EXTRACT XML SCHEMA

[4] proposed an approach to extract schema from semi-structured data. Our research is mainly based on that study. Constraint logic programming is a paradigm that builds a mechanism of restriction cancellation into logic programming; [3] presents a concept of constraint into a relative connotation. [2] proposed to identify relationships between attributes or classes in different database schemas. Details of our research are described in [1].

Our approach has three steps. The first step is XML to predicates, the second step is predicates to rules by induction inference, and the last step is predicates to rules by using ontology. We evaluate similarity of data type and data range by using an ontology dictionary, and XML Schema is made from results of second and last step.

2.1 XML to Predicates

The first step is production of constraints from XML documents. The standard manner is to contain a link-structure and values to extract data into XML (the order of elements is not guaranteed), and to represent the XML using seven base relations defined as follows:

Here, Object is O, Class is C, FromObject is FO, ToObject is TO, Label is L, and Value is V.

class(O, C) :- The class(O, C) corresponds to O being C.

link(FO, TO, L) :- The link(FO, TO, L) corresponds to an edge labeled L from FO to TO. There is at most one such edge labeled L.

element(O, V) :- The element(O, V) corresponds to O being element and having V.

attribute(O, V1, V2) :- The attribute(O, V1, V2) corresponds to O being an attribute and having V1 and V2.

root(O) :- The root(O) corresponds to O being a root-tag.

branch(O) :- The branch(O) corresponds to O being a branch.

leaf(O) :- The root(O) corresponds to O being a leaf tag.

Next, our approach generates predicates for the data type or data range. In our study, we assumed that the data type and data range were similar to a tag that expresses the same meaning. Moreover, these are expressed by the data type and the regular expression of XML schema. The generated predicate becomes two kinds of the following.

datatype(O, V). The data type of O is V. Type includes decimal, string, Boolean, etc.

datarange(O, V). The data range of O is V. V records the range of character length or the value. For instance, the numeric character data and ranges of 1000–9999, are described as follows.
xsd:minInclusive value="1000",
xsd:maxInclusive value="9999" .

WWW 2008, April 21–25, 2008, Beijing, China.
ACM 978-1-60558-085-2/08/04.

Table 1: Experimental Data.

Test Case	Schema ?	Implisit Structure ?	Instance Filename	Average Instance Size (Kbytes)	Average Number of Nodes
1	Y	Y	testcase1_1--100.xml	22	112
2	N	Y	testcase2_1--100.xml	21	112
3	N	N	testcase3_1--100.xml	22	112

2.2 Predicates to Rules by Induction Inference

We use Progol, which is a system of typical induction logic programming, to analyze classification rules. The background knowledge used for the inductive inference is a hierarchical relation of the amount of the multistep floor features and the connotation relation between the amount of characteristics. In this case, two classification rules were derived using a study that addressed both data of a negative example and a positive example.

It is impossible to detect differences from the study data even a little, but the accuracy of the classified category generation is high because usual studies that use data of both a positive example negative example suggest a classification rule that the restriction is strong compared to studies that use only a positive example. The accuracy of the category generation worsens because the restriction of a study that uses only a positive example is weak and one-sided, although it is possible to correspond when using only study data with little difference.

Our approach to improving case-sensitive matching is taken by obtaining the retrieval result because the detection results by the restriction of study that uses only a positive example are integrated based on the restriction of a usual study that uses both data of the positive example and the negative example in this study.

3. EXPERIMENTAL RESULTS

The dataset comprises 300 collected XML files, e.g., News-ML and weblogs, from which 81 unique XML documents were made. The results of running our typing algorithm for several synthetic data sets are presented in Table 2. We conducted 100 experiments per test case. Here, XML data were acquired from about a 110-node instance size. 'Schema?' in Table 1 shows whether the test case has a scheme, e.g. DTD. 'Implicit Structure?' in Table 1 shows whether the test case has implicit structures. Evaluation Value is a ratio that uses data of both the positive example and negative example; for the negative example, in the case of 0.8, the data only of 80% and a positive example takes the meaning of 20%. Here, the value of the Evaluation Value has three patterns: 0.8, 0.6, and 0.4 for each Test Case.

The results of experiments are presented in Table 2. Their corresponding graphs are shown in Fig. 1. Test Number in Table 2 and Test Number on the horizontal axis of Fig. 1 mutually correspond. When the Evaluation Value was one or less, it did not become 100%, although Test Case1 with the schema was a very high recall ratio in this experiment (The result shown in Table 2 is displayed to three significant digits). In the experiment, the generated restriction was queried, and a method of obtaining the set of necessary data was described. Results confirmed whether man was correct for the relevance ratio and the recall ratio. Test Case 2 with implicit structure : though it doesn't have the schema. The relevance ratio is about 60% when the recall ratio is about 74%. Furthermore, it is low when the Evaluation Value is improving. The prevailing purpose of this study is Test Case 2. Future studies are intended to design a technique for displaying the relevance ratio with high accuracy simultaneously to boosting the recall ratio of this case as high as possible.

Table 2: Experimental Results A.

Test Number	Test Case	Evaluation Value	Average Constraint Count	Average Precision(%)	Average Recall(%)
1	1	0.8	11	91.0	100
2	1	0.6	11	78.8	100
3	1	0.4	11	78.6	100
4	2	0.8	21	80.2	74.1
5	2	0.6	21	64.7	82.3
6	2	0.4	21	59.7	83.9
7	3	0.8	35	38.6	25.6
8	3	0.6	35	35.4	32.5
9	3	0.4	35	22.7	49.2

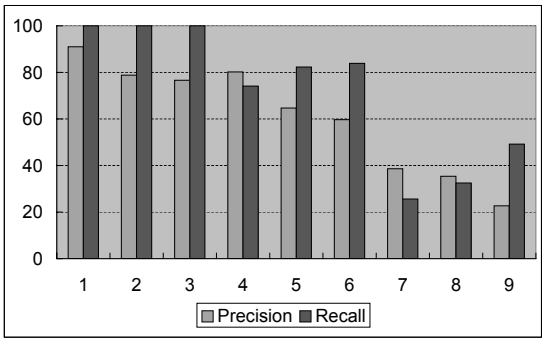

Figure 1: Experimental Results B.

4. CONCLUSIONS

To summarize, this paper proposes a framework for XML retrieval which can find similar meanings of XML within different XML structures. Our approach uses syntax and semantics search; it also uses predicate logic. Results of experiments using real and synthetic data confirmed that our system can compare many kinds of XML documents. The verification of effectiveness for processing the tag structures which have only small differences is a task of future studies, although the present study achieved a highly accurate classification and retrieval for tag structures with a large difference of features.

5. REFERENCES

[1] Masaya Eki, Tadachika Ozono, Toramatsu Shintani, 'On an XML Database System Based on Constraint Logic Programming', WorldComp ICAI'07, pages 859-865, 2007.

[2] Wen-Syan Li, Chris Clifton, 'SEMINT: a tool for identifying attribute correspondences in heterogeneous databases using neural networks', Data & Knowledge Engineering, Volume 33, Issue 1, Pages 49-84, Apr 2000.

[3] Fumio Mizoguchi, Hayato Ohwada, 'Constraint relative least general generalization for inducing constraint logic programs', New Generation Computing, pages 335-368, 1995.

[4] Svetlozar Nestorov, Serge Abiteboul, Rajeev Motwani, 'Extracting Schema from Semistructured Data', SIGMOD'98, pages 295-306, 1998.