Schema Extraction from XML Collections

Boris Chidlovskii
Xerox Research Centre Europe, Grenoble Laboratory
6, Chemin de Maupertuis, F–38240 Meylan, France
childovskii@xrce.xerox.com

ABSTRACT

XML Schema language has been proposed to replace Document Type Definitions (DTDs) as schema mechanism for XML data. This language consistently extends grammar-based constructions with constraint- and pattern-based ones and have a higher expressive power than DTDs. As schemas remain optional for XML, we address the problem of XML Schema extraction. We model the XML schema as extended context-free grammars and develop a novel extraction algorithm inspired by methods of grammatical inference. The algorithm copes also with the schema determinism requirement imposed by XML DTDs and XML Schema languages.

Categories and Subject Descriptors

H.3 [**Information Systems**]: Information Storage and Retrieval

General Terms

Design, Management

Keywords

Schema extarction, XML Schemas, context-free grammars

1. INTRODUCTION

The Extensible Markup Language (XML) is a standard for exchange and manipulation of structured documents. The structure of XML documents is often described by Document Type Definitions (DTDs). Unfortunately, DTD capabilities appear to be limited for many application domains. Most important limitations of DTDs are a non-XML syntax, a limited set of datatypes and loose structured constraints [3, 4]. As result, new XML schema languages, such as DSD, Schematron and, in particular, XML Schema, have been recently proposed to replace DTDs. These languages extend the DTD model with novel important features, such as simple and complex types, rich datatype sets, occurrence constraints and inheritance. As the XML Schema language [1] is widely accepted as the DTD successor, we align our approach with this language.

Problem statement. Despite the importance of schema information, schema definitions are not obligatory in XML documents. This raises the *problem of extracting the schematic information from XML documents*. Ideally, the extracted schema should, on one side, *tightly represent* the data, and be *concise and compact*, on the other side. As the two requirements essentially contradict each other, finding an optimal tradeoff is a difficult and challenging task.

Automatic schema extraction is highly helpful in many situations. First, it can be used for real world *XML data with complex structure*; for such data, schema design is a difficult and error-prone work that often results in a large number of badly designed DTDs [5]. Second, the automatic schema extraction can be beneficial in mediator systems that cope with *heterogeneous collections* of XML documents and need a common schema for all collections. Finally, the *semi-automatic schema extraction* may assist the designer in the schema design. It consists in analyzing available XML data, refining the existing schema patterns and finding new ones.

Our contribution in this paper is three-fold. First, we adopt **the XML schema formalism** based on *extended context-free grammars* (ECFG) with *range regular expressions* allowed in nonterminal productions; such regular expressions combine grammatical forms and constraints for nonterminals and element groups similarly to constructions of XML Schema language.

Second, with the proposed schema model, we address the **problem of schema extraction** from XML collections. The ECFG-based schema model makes the extraction problem more complex than in the DTD extraction case [2], so we identify three important components, namely, (1) *induction the context-tree grammars* from XML documents represented as structured examples, (2) *generalization of content strings* into regular expressions, and (3) *constraining datatypes* for simple XML elements. The second problem is the same as with the DTD extraction, but the first and third ones are relevant to the powerful schema mechanisms offered by novel XML schema languages. For the first problem, we extend the method of CFG inference from structural examples [6]. For the datatypes constraining, we develop an algorithm based on the subsumption relationships among elementary datatypes in XML Schema language. For the content generalization, we propose a solution alternative to the DTD extraction [2], in order to cope with the occurrence constraints in schemas.

Third, we address the **determinism** requirement, imposed by both XML DTDs and XML Schema to easily validate XML data against corresponding schema. Determinism can essentially constrain the power of ECFG model, as a large part of grammars does not provide the feature. We study both *horizontal and vertical determinism* that address the ease of vertical and horizontal navigation in an XML document tree.

2. XML SCHEMA FORMALISM

We model XML schema as range extended context-free gram-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
JCDL '02, July 13-17, 2002, Portland, Oregon, USA
Copyright 2002 ACM 1-58113-513-0/02/0007 ...$5.00.

mars. *Range regular expression* is a regular expression over an alphabet Σ, where each term is defined with the range $[l : r]$ of possible occurrences, where l and r are nonnegative integers, $0 \leq l \leq r$, r can be ∞. Using the range notation, the Kleen closure $a*$ is equivalent to $a[0 : \infty]$, and $a?$ is equivalent to $a[0 : 1]$.

The benefit of using range regular expressions in the schema formalism is two-fold. First, range regular expressions extend the schema model with occurrence constraints in the same way as the XML Schema language does. Second, it prompts the definition of *limited element occurrences* instead of unlimited ones in DTDs ($a*$ and $a+$), which is highly valuable for query formulation and optimization of XML data storage.

A *(range) extended context free grammar* (ECFG) is defined by 5-tuple $G = (T, N, D, \delta, Start)$, where T, N and D are disjoint sets of terminals, nonterminals and datatypes; $Start$ is an initial nonterminal and δ is a finite set of production rules of the form $A \rightarrow \alpha$ for $A \in N$, where α is a range regular expression over *terms*, where one term is a terminal-nonterminal-terminal sequence like $t\ B\ t'$, briefly $t : B$, where $t, t' \in T$ and $B \in N \cup D$.

The ECFG-based schema model targets a core set of features of the XML Schema language [1] so that any XML Schema definition corresponds to some extended context free grammar and vice versa. The feature set includes *elementary datatypes* for simple elements, *complex types* for complex elements, *sequence* and *choice* groups of elements, *occurrence constraints* for elements and groups.

There exists a correspondence between components an ECFG G and XML Schema definition S. The terminal and datatype sets T and D in G correspond to element names and elementary datatypes in S. The nonterminal set N in G corresponds to the set of complex types in S, both named and abstract. Productions in G corresponds to definitions of the complex types. One production is a *sequence* or *choice* group of typed elements or other (nested) groups.

For the schema determinism, we follow [3] in distinguishing between *vertical* and *horizontal* determinism. The *vertical determinism* requires that at any complex element, the rule for any subelement should be determined with one-token lookahead. Formally, two terms $t : A$ and $t' : A'$ are *ambiguous* if $t = t'$ but $A \neq A'$. Ambiguous terms in productions make difficult validating XML documents; on the other hand, the absence of ambiguous terms in productions guarantees the vertical determinism of an ECFG and gives the *sufficient* condition for inferring deterministic ECFGs.

PREPOSITION 1. *An ECFG G guarantees the vertical determinism if no production in G contains ambiguous terms.*

3. SCHEMA INDUCTION OVERVIEW

The extraction algorithm comprises several important steps; below we report on the core step of ECFG inference, details on tightening and content generalization can be found in the full version.

Algorithm 1. Schema extraction from XML data.

```
0. Represent XML documents as set $I$ of structured examples.
1. Induce an extended context-free grammar $G$ from $I$:
   1.1. Create the initial set of nonterminals $N$;
   1.2. Merge nonterminals in $N$ with the similar content and context;
   1.3. Determine tight datatypes for terminals in $G$;
   1.4. Generalize contents in nonterminals into range REs.
2. Transform the result ECFG $G$ into an XML Schema definition $S$.
```

First, we represent XML documents as a *structured example* of an (unknown) ECFG, where structured examples are derivation trees of a CFG with all nonterminal labels removed. In [6], a context-free grammar is inferred from structural examples by merging nonterminals with *equivalent content* and *context* as follows:

[**1. Content equivalence:**] $A \rightarrow \alpha, B \rightarrow \alpha$ in $\delta \implies A = B$,
[**2. Context equivalence:**] $A \rightarrow \alpha B \beta$ and $A \rightarrow \alpha C \beta$ in $\delta \implies B = C, \alpha, \beta \in (N \cup D)^*$.

For ECFG-based schema model, these two equivalence rules should be properly extended to meet the determinism requirement. Also, the content equivalence condition appears to be too strong as it fails to merge nonterminals whose right parts are instances of one regular expression. Therefore, we replace the equivalence with a weaker condition of *similarity*. For the determinism requirement, we implement Preposition 1 for the vertical determinism as generalization of the context equivalence rule (2):
[**3. Context similarity:**] $A \rightarrow \alpha\ t{:}B\ \beta\ t{:}C\ \gamma \implies B = C$.

The merge of nonterminals with similar content requires a special metric that can control the merge accordingly to the user's perception. We consider two contents as strings over terminal alphabet T and use the vector space model to measure the string similarity.We generalize the model by considering n-grams in content strings, where n-gram is a sequence of n consequent elements in the content, $n = 1, 2, 3, \ldots$. We denote the set of n-grams in a content string s as $P_n(s)$ and count n-grams for the content string c; thus c is a normalized vector $c = \{c^i\}, i = 1, 2, \ldots$ of n-gram frequencies, $\sum c^i = 1$. The *similarity measure* between two content strings c_1 and c_2 is given by $\mathcal{M}(c_1, c_2) = \sum c_1^i \cdot c_2^i$. Two contents c_1 and c_2 are *similar* if $\mathcal{M}(c_1, c_2) > th$, where $0 \leq th \leq 1$ is a *threshold value* set by user. The choice of th is important; values close to 0 may result in merging most of nonterminals, while values close to 1 make difficult the merge of even visibly similar contents.

If nonterminals are given by two disjunction of contents, their similarity is given by the maximal similarity over pairs of contents from corresponding conjunctions. Formally, if two nonterminals $A \rightarrow \alpha$ and $B \rightarrow \beta$ are such that $\alpha = c_1|c_2|\ldots$ and $\beta = c'_1|c'_2|\ldots$, then $\mathcal{M}(\alpha, \beta) = max_{c_i \in \alpha, c'_j \in \beta} \mathcal{M}(c_i, c'_j)$. Thus we can rewrite the equivalence condition (1) with a weaker similarity one:
[**4. Content similarity:**] For $A \rightarrow \alpha$ and $B \rightarrow \beta$ in Σ, such that M$(\alpha, \beta) > th, \implies A = B$.

4. CONCLUSION

We have developed a novel schema extraction from XML documents. To our knowledge, this is the first attempt to induce XML schema that unifies the expressive power of ECFGs and the determinism requirement. We have identified three important components of the extraction algorithm, namely, the grammar induction itself, content generalization and tight datatype identification and have developed sophisticated solutions for each of them.

5. REFERENCES

[1] D.C Fallside. XML Schema Part 0: Primer. W3C Candidate Recommendation. http://www.w3.org/TR/xmlschema-0/.

[2] M. N. Garofalakis, A. Gionis, R. Rastogi, et al. XTRACT: A System for Extracting Document Type Descriptors from XML Documents. In *Proc. ACM SIGMOD*, pp. 165–176, 2000.

[3] D. Lee, M. Mani, M. Murata. Reasoning about XML Schema Languages using Formal Language Theory. Technical report, IBM Almaden Research Center, 2000.

[4] Y. Papakonstantinou, V. Vianu. DTD Inference for Views of XML Data. In *Proc. ACM PODS*, pp. 35–46, 2000.

[5] A. Sahuguet. Everything you ever wanted to know about dtds, but were afraid to ask. In *Proc. WebDB Workshop*, 2000.

[6] Y. Sakakibara. Efficient Learning of Context-Free Grammars from Positive Structural Examples. *Information and Computation*, 97(1), pp. 23–60, 1992.