# XML Validation for Context-Free Grammars

Yasuhiko Minamide[1] and Akihiko Tozawa[2]

[1] Department of Computer Science
University of Tsukuba
[2] IBM Research,
Tokyo Research Laboratory, IBM Japan, ltd.

**Abstract.** String expression analysis conservatively approximates the possible string values generated by a program. We consider the validation of a context-free grammar obtained by the analysis against XML schemas and develop two algorithms for deciding inclusion $L(G_1) \subseteq L(G_2)$ where $G_1$ is a context-free grammar and $G_2$ is either an XML-grammar or a regular hedge grammar. The algorithms for XML-grammars and regular hedge grammars have exponential and doubly exponential time complexity, respectively. We have incorporated the algorithms into the PHP string analyzer and validated several publicly available PHP programs against the XHTML DTD. The experiments show that both of the algorithms are efficient in practice although they have exponential complexity.

## 1 Introduction

String expression analysis conservatively approximates the possible string values generated by a program [CMS03b]. Minamide adopted context-free grammars as a foundation of string expression analysis and developed a string analyzer for PHP [Min05]. We consider the validation of a context-free grammar obtained by the analysis against XML schemas and develop two algorithms for deciding inclusion $L(G_1) \subseteq L(G_2)$ where $G_1$ is a context-free grammar and $G_2$ is either an XML-grammar or a regular hedge grammar, which are subclasses of context-free grammars theoretically corresponding to XML schema languages such as Document Type Definition (DTD) and RELAX NG [CM01].

To simplify the discussion on XML validation, we consider languages over a paired alphabet. Context-free languages with parentheses or paired alphabets were studied extensively in the 1960s and 1970s [McN67, Knu67, Tak75]. Let $A$ be a base alphabet. Then, we introduce a paired alphabet consisting of two sets $\acute{A}$ and $\grave{A}$:

$$\acute{A} = \{\, \acute{a} \mid a \in A \,\} \qquad \grave{A} = \{\, \grave{a} \mid a \in A \,\}$$

where $\acute{A}$ and $\grave{A}$ correspond to the set of start tags and the set of end tags, respectively. We consider that $\acute{a}$ and $\grave{a}$ match. We write $\Sigma$ for $\acute{A} \cup \grave{A}$. This notation is based on Takahashi's work on context-free grammars [Tak75].

The fundamental notion on a string over a paired alphabet is whether it is balanced. For example, $\acute{a}\acute{b}\grave{b}\acute{c}\grave{c}\grave{a}$ and $\acute{a}\grave{a}\acute{b}\grave{b}$ are balanced, but $\acute{a}\grave{b}$ and $\acute{a}\acute{b}\grave{b}$ are not.

N. Kobayashi (Ed.): APLAS 2006, LNCS 4279, pp. 357–373, 2006.

This notion of balanced strings corresponds to well-formed documents in XML. We call the set of all balanced strings $B(\Sigma)$ the Dyck set over $\Sigma$.

As a balanced subclass of context-free languages, Berstel and Boasson proposed XML-grammars modeling DTDs and studied their formal properties [BB02]. An XML-grammar consists of a set of terminals $\Sigma = \acute{A} \cup \grave{A}$, a set of nonterminals $V$ in one-to-one correspondence with base alphabet $A$, a start nonterminal $S$, and a set of productions. For each $a \in A$, there must be a unique production of the following form:

$$X_a \to \acute{a} R_a \grave{a}$$

where $X_a$ is the nonterminal corresponding to $a$ and $R_a$ is a regular expression over $V$.

*Example 1.* Consider the following DTD, taken from [BB02].

```
<!DOCTYPE a [
   <!ELEMENT a ((a|b),(a|b)) >
   <!ELEMENT b (b)* >
]>
```

This DTD can be represented by an XML-grammar with the following productions:

$$X_a \to \acute{a}(X_a|X_b)(X_a|X_b)\grave{a}$$
$$X_b \to \acute{b} X_b^* \grave{b}$$

where $X_a$ and $X_b$ are the nonterminals corresponding to $a$ and $b$, respectively, and $X_a$ is the start symbol.

In this formal setting, validating a context-free grammar against a DTD corresponds to checking $L(G) \subseteq L(G_{\mathsf{xml}})$ for a context-free grammar $G$ and an XML-grammar $G_{\mathsf{xml}}$. To develop an algorithm checking this inclusion, we exploit locality in DTDs and XML-grammars. They have locality in the sense that they can only describe a relation between an element and its children as can seen in the definition of XML-grammars. The algorithm has exponential time complexity and is presented in Section 4.

There is a larger class of grammars called *regular hedge grammars* corresponding to regular tree languages over unranked alphabets [Mur99]. The class of regular hedge grammars can be formulated as an extension of XML-grammars where each production has the following form:

$$X \to R$$

where $R$ is an arbitrary regular expression over $\acute{a} Y \grave{a}$. Also there is an alternative formulation of regular hedge grammars. We obtain grammars of the same expressiveness by restricting each production to one of the following forms:

$$X \to \acute{a} Y \grave{a} Z \quad \text{or} \quad X \to \epsilon.$$