XML Grammars

Jean Berstel[1] and Luc Boasson[2]

[1] Institut Gaspard Monge (IGM), Université Marne-la-Vallée,
F-77454 Marne-la-Vallée Cédex 2
http://www-igm.univ-mlv.fr/~berstel
[2] Laboratoire d'informatique algorithmique: fondements et applications (LIAFA),
Université Denis-Diderot,
F-75251 Paris Cédex 05
http://www.liafa.jussieu.fr/~lub

Abstract. XML documents are described by a document type definition (DTD). An XML-grammar is a formal grammar that captures the syntactic features of a DTD. We investigate properties of this family of grammars. We show that an XML-language basically has a unique XML-grammar. We give two characterizations of languages generated by XML-grammars, one is set-theoretic, the other is by a kind of saturation property. We investigate decidability problems and prove that some properties that are undecidable for general context-free languages become decidable for XML-languages.

1 Introduction

XML (eXtensible Markup Language) is a format recommended by W3C in order to structure a document. The syntactic part of the language describes the relative position of pairs of corresponding tags. This description is by means of a document type definition (DTD). In addition to its syntactic part, each tag may also have attributes. If the attributes in the tags are ignored, a DTD appears to be a special kind of context-free grammar. The aim of this paper is to study this family of grammars.

One of the consequences will be a better appraisal of the structure of XML documents. It will also show some limitations of the power of expression of XML. Consider for instance an XML-document that consists of a sequence of paragraphs. A first group of paragraphs is being typeset in bold, a second one in italic, and there should be as many paragraphs in bold than in italic. As we shall see (Example 4.6), it is not possible to specify this condition by a DTD. This is due to the fact that the context-free grammars corresponding to DTD's are rather restricted.

The main results in this paper are two characterizations of XML-languages. The first (Theorem 4.1) is set-theoretic. It shows that an XML-language is the biggest language in some class of languages. It relies on the fact that, for each XML-language, there is only one XML-grammar that generates it. The second characterization (Theorem 4.3) is syntactic. It shows that XML-languages have a kind of "saturation property".

M. Nielsen and B. Rovan (Eds.): MFCS 2000, LNCS 1893, pp. 182–191, 2000.

As usual, these results can be used to show that some languages cannot be XML. This means in practice that, in order to achieve some features of pages, additional nonsyntactic techniques have to be used.

The paper is organized as follows. The next section contains the definition of XML-grammars and their relation to DTD. Section 3 contains some elementary results, and in particular the proof that there is a unique XML-grammar for each XML-language. It appears that a new concept plays an important role in XML-languages. This is the notion of surface. The surface of an opening tag a is the set of sequences of opening tags that are children of a. The surfaces of an XML-language must be regular sets, and in fact describe the XML-grammar. The characterization results are given in Section 4. They heavily rely on surfaces, but the second one also uses the syntactic concept of a context.

Section 5 investigates decision problems. It is shown that it is decidable whether the language generated by a context-free language is well-formed, but it is undecidable whether there is an XML-grammar for it. On the contrary, it is decidable whether the surfaces of a context-free grammar are finite. The final section is a historical note. Indeed, several species of context-free grammars investigated in the sixties, such as parenthesis grammars or bracketed grammars are strongly related to XML-grammars. This relationship is sketched.

2 Notation

An XML document [6] is composed of text and of tags. The tags are opening or closing. Each opening tag has a unique associated closing tag, and conversely. There are also tags called empty tags, and which are both opening and closing. In a canonical XML [7], these tags are always replaced by a sequence composed of an opening tag immediately followed by its closing tag. We do so here, and therefore assume that there are no empty tags.

Let A be a set of opening tags, and let $\bar{A}$ be the set of corresponding closing tags. Since we are interested in syntactic structure, we ignore any text. Thus, an XML document is a word over the alphabet $A \cup \bar{A}$.

A document x is *well-formed* if the word x is a correctly parenthesized word, that is if x is in the set of Dyck primes over $A \cup \bar{A}$. Observe that the word is a prime, so it is not a product of two well parenthesized words. Also, it is not the empty word.

An XML-*grammar* is composed of a terminal alphabet $A \cup \bar{A}$, of a set of variables V in one-to-one correspondence with A, of a distinguished variable called the *axiom* and, for each letter $a \in A$ of a regular set $R_a \subset V^*$ defining the (possibly infinite) set of productions

$$X_a \to am\bar{a}, \qquad m \in R_a, \quad a \in A$$

We also write for short

$$X_a \to aR_a\bar{a}$$

as is done in DTD's. An XML-*language* is a language generated by some XML-grammar.

It is well-known from formal language theory that if non-terminals in a context-free grammar are allowed to have infinite regular or context-free sets of productions, then the language generated is still context-free. Thus, any XML-language is context-free.

Example 2.1. The language $\{a^n\bar{a}^n \mid n > 0\}$ is a XML-language, generated by

$$X \to a(X|\varepsilon)\bar{a}$$

Example 2.2. The language of *Dyck primes* over $\{a, \bar{a}\}$ is a XML-language, generated by

$$X \to aX^*\bar{a}$$

Example 2.3. The language D_A of *Dyck primes* over $A \cup \bar{A}$ is generated by the grammar

$$\begin{aligned} X &\to \textstyle\sum_{a\in A} X_a \\ X_a &\to aX^*\bar{a}, \qquad a \in A \end{aligned}$$

This grammar is not XML. Each X_a in this grammar generates an XML-language, which is $D_A \cap a(A \cup \bar{A})^*\bar{a}$. However D_A is not an XML-language if A has more than one letter.

In the sequel, all grammars are assumed to be *reduced*, that is, every non-terminal is accessible from the axiom, and every non-terminal produces at least one terminal word. Note that, given a grammar with an infinite set of productions, the classical reduction procedure can be applied to get an equivalent reduced grammar when the sets of productions are recursive.

Given a grammar G over a terminal alphabet T and a nonterminal X we denote by

$$L_G(X) = \{w \in (A \cup \bar{A})^* \mid X \xrightarrow{*} w\}$$

the language generated by X in the grammar G.

Remark 2.4. The definition has the following correspondence to the terminology and notation used in the XML community ([6]). The grammar of a language is called a *document type definition* (DTD). The axiom of the grammar is qualified `DOCTYPE`, and the set of productions associated to a tag is an `ELEMENT`. The syntax of an element implies by construction the one-to-one correspondence between pairs of tags and non-terminals of the grammar. Indeed, an element is composed of a *type* and of a *content model*. The type is merely the tag name and the content model is a regular expression for the set of right-hand sides of the productions for this tag. For instance, the grammar

$$\begin{aligned} S &\to a(S|T)(S|T)\bar{a} \\ T &\to bT^*\bar{b} \end{aligned}$$

with axiom S corresponds to

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

Here, S and T stand for the nonterminals X_a and X_b respectively.

The regular expressions allowed for the content model are of two types : those called children, and those called mixed. In fact, since we do not consider text, the mixed expressions are no more special expressions.

In the definition of XML-grammars, we ignore *entities*, both general and parameter entities. Indeed, these may be considered as shorthands and are handled at a lexical level.

3 Elementary Results

We denote by D_a the language of *Dyck primes* starting with the letter a. This is the language generated by X_a in Example 2.3. We set $D_A = \cup_{a\in A} D_a$. This is not an XML-language if A has more than one letter. We call D_A the set of Dyck primes over A and we omit the index A if possible. The set D is known to be a *bifix* code, that is no word in D is a proper prefix or a proper suffix of another word in D.

Let L be any subset of the set D of Dyck primes over A. The aim of this section is to give a necessary and sufficient condition for L to be an XML-language.

We denote by $F(L)$ the set of factors of L, and we set $F_a(L) = D_a \cap F(L)$ for each letter $a \in A$. Thus $F_a(L)$ is the set of those factors of words in L that are also Dyck primes starting with the letter a.

Example 3.1. For the language

$$L = \{ab^{2n}\bar{b}^{2n}\bar{a} \mid n \geq 1\}$$

one has $F_a(L) = L$ and $F_b(L) = \{b^n\bar{b}^n \mid n \geq 1\}$.

Example 3.2. Consider the language

$$L = \{a(b\bar{b})^n(c\bar{c})^n\bar{a} \mid n \geq 1\}$$

Then $F_a(L) = L$, $F_b(L) = \{b\bar{b}\}$, $F_c(L) = \{c\bar{c}\}$.

The sets $F_a(L)$ are important for XML-languages and grammars, as illustrated by the following lemma:

Lemma 3.3. *Let G be an XML-grammar over $A \cup \bar{A}$ generating a language L, with nonterminals X_a, for $a \in A$. For each $a \in A$, the language generated by X_a is the set of factors of words in L that are Dyck primes starting with the letter a, that is*

$$L_G(X_a) = F_a(L)$$

Proof. Set $T = A \cup \bar{A}$. Consider first a word $w \in L_G(X_a)$. Clearly, w is in D_a. Moreover, since the grammar is reduced, there are words g, d in T^* such that $X \xrightarrow{*} gX_a d$, where X is the axiom of G. Thus w is a factor of L.

Conversely, consider a word $w \in F_a(L)$ for some letter a, let g, d be a words such that $gwd \in L$. Due to the special form of an XML-grammar, any letter a can only be generated by a production with non-terminal X_a. Thus, a left derivation $X \xrightarrow{*} gwd$ factorizes into

$$X \xrightarrow{k} gX_a\beta \xrightarrow{*} gwd$$

for some word β, where k is the number of letters in g that are in A. Next

$$gX_a\beta \xrightarrow{*} gw'\beta \xrightarrow{*} gwd$$

with $X_a \xrightarrow{*} w'$ and $w' \in D$. None of w and w' can be a proper prefix of the other, because D is bifix. Thus $w' = w$. This shows that w is in $L_G(X_a)$ and proves that $F_a = L_G(X_a)$. □

Corollary 3.4. *For any XML-language $L \subset D_a$, one has $F_a(L) = L$.* □

Let w be a Dyck prime in D_a. It has a unique factorization

$$w = au_{a_1}u_{a_2}\cdots u_{a_n}\bar{a}$$

with $u_{a_i} \in D_{a_i}$ for $i = 1, \ldots, n$. The *trace* of the word w is defined to be the word $a_1a_2\cdots a_n \in A^*$.

If L is any subset of D, and $w \in L$, then the words u_{a_i} are in $F_{a_i}(L)$. The *surface* of $a \in A$ in L is the set $S_a(L)$ of all traces of words in $F_a(L)$. Observe that the notion of surface makes sense only for subsets of D.

Example 3.5. For the language of Example 3.1, the surfaces are easily seen to be $S_a = \{b\}$ and $S_b = \{b, \varepsilon\}$.

Example 3.6. The surfaces of the language of Example 3.2 are $S_a = \{b^n c^n \mid n \geq 1\}$ and $S_b = S_c = \{\varepsilon\}$.

It is easily seen that the surfaces of the set of Dyck primes over A are all equal to A^*.

Surfaces are useful for defining XML-grammars. Let $\mathcal{S} = \{S_a \mid a \in A\}$ be a family of regular languages over A. We define an XML-grammar G associated to $\mathcal{S}$ called the *standard grammar* of $\mathcal{S}$ as follows. The set of variables is $V = \{X_a \mid a \in A\}$. For each letter a, we set

$$R_a = \{X_{a_1}X_{a_2}\cdots X_{a_n} \mid a_1a_2\cdots a_n \in S_a\}$$

and we define the productions to be

$$X_a \to am\bar{a}, \qquad m \in R_a$$

for all $a \in A$. Since S_a is regular, the sets R_a are regular over the alphabet V. By construction, the surface of the language generated by a variable X_a is S_a, that is $S_a(L_G(X_a)) = S_a$. For any choice of the axiom, the grammar is an XML-grammar.

Example 3.7. The standard grammar for the surfaces of Example 3.1 is

$$X_a \to aX_b\bar{a}$$
$$X_b \to b(X_b|\varepsilon)\bar{b}$$

The language generated by X_a is $\{ab^n\bar{b}^n\bar{a} \mid n \geq 1\}$ and is *not* the language of Example 3.1.

This construction is in some sense the only way to build XML-grammars, as shown by the following proposition.

Proposition 3.8. *For each XML-language L, there exists exactly one reduced XML-grammar generating L, up to renaming of the variables.*

Proof. (Sketch) Let G be an XML-grammar generating L, with nonterminals $V = \{X_a \mid a \in A\}$, and $R_a = \{m \in V^* \mid X_a \longrightarrow am\bar{a}\}$ for each $a \in A$. We claim that the mapping

$$X_{a_1}X_{a_2}\cdots X_{a_n} \mapsto a_1a_2\cdots a_n$$

is a bijection from R_a onto the surface $S_a(L)$ for each $a \in A$. Since the surface depends only on the language, this suffices to prove the proposition. □

It follows easily that

Corollary 3.9. *Let L_1 and L_2 be XML-languages. Then $L_1 \subset L_2$ iff $S_a(L_1) \subset S_a(L_2)$ for all a in A.*

This in turn implies

Proposition 3.10. *The inclusion and the equality of XML-languages are decidable.*

In particular, it is decidable whether an XML-language L is empty. Similarly, it is decidable whether $L = D_\alpha$.

XML-languages are not closed under union and difference. This will be an easy consequence of the characterizations given in the next section (Example 4.8).

Proposition 3.11. *The intersection of two XML-languages is an XML-language.*

4 Two Characterizations of XML-Languages

In this section, we give two characterizations of XML-language. The first (Theorem 4.1) is based on surfaces. It states that, for a given set of regular surfaces, there is only one XML-language with these surfaces, and that it is the maximal language in this family. The second characterization (Theorem 4.3) is syntactical and based on the notion of context.

Let $\mathcal{S} = \{S_a \mid a \in A\}$, be a family of regular languages, and fix a letter a_0 in A. Define $\mathcal{L}(\mathcal{S})$ to be the family of languages $L \subset D_{a_0}$ such that $S_a(L) = S_a$ for all a in A. Clearly, any union of sets in $\mathcal{L}(\mathcal{S})$ is still in $\mathcal{L}(\mathcal{S})$, so there is a maximal language (for set inclusion) in the family $\mathcal{L}(\mathcal{S})$. The *standard* language associated to $\mathcal{S}$ is the language generated by X_{a_0} in the standard grammar of $\mathcal{S}$.

Theorem 4.1. *The standard language associated to $\mathcal{S}$ is the maximal element of the family $\mathcal{L}(\mathcal{S})$. This language is XML, and it is the only XML-language in the family $\mathcal{L}(\mathcal{S})$.*

Example 4.2. The standard language associated to the sets $S_a = \{b\}$ and $S_b = \{b, \varepsilon\}$ of Example 3.1 is the language $\{ab^n\bar{b}^n\bar{a} \mid n \geq 1\}$ of Example 3.7. Thus, the language of Example 3.1 is not XML.

We now give a more syntactic characterization of XML-languages. For this, we define the set of *contexts* in L of a word w as the set $C_L(w)$ of pairs of words (x, y) such that $xwy \in L$.

Theorem 4.3. *A language L over $A \cup \bar{A}$ is an XML-language if and only if*
(i) *$L \subset D_\alpha$ for some $\alpha \in A$,*
(ii) *for all $a \in A$ and $w, w' \in F_a(L)$, one has $C_L(w) = C_L(w')$,*
(iii) *the set $S_a(L)$ is regular for all $a \in A$.*

Before giving the proof, let us compute one example.

Example 4.4. Consider the language L generated by the grammar

$$S \to aTT\bar{a}$$
$$T \to aTT\bar{a} \mid b\bar{b}$$

with axiom S. This grammar is not XML. Clearly, $L \subset D_a$. Also, $F_a(L) = L$. There is a unique set $C_L(w)$ for all $w \in L$, because at any place in a word in L, a factor w in L can be replaced by another factor w' in L. Finally, $S_a(L) = (a \cup b)^2$ and $S_b(L) = \varepsilon$. The theorem claims that there is an XML-grammar generating L.

Proof. We write F_a, S_a and $C(w)$, with the language L understood. We first show that the conditions are sufficient.

Let G be the XML-grammar defined by the family S_a and with axiom X_α. We prove first $L_G(X_a) = F_a$ for $a \in A$. We let the proof of the inclusion $F_a \subset L_G(X_a)$ to the reader. Next, we prove the inclusion $F_a \supset L_G(X_a)$ by induction on the derivation length k. Assume $X_a \xrightarrow{k} w$. Then $w = au\bar{a}$ for some word u. If $k = 1$, then the empty word is in S_a, which means that $a\bar{a}$ is in F_a. If $k > 1$, then the derivation factorizes in

$$X_a \to aX_{a_1} \cdots X_{a_n}\bar{a} \xrightarrow{k-1} au\bar{a}$$

for some production $X_a \to aX_{a_1} \cdots X_{a_n}\bar{a}$. Thus there is a factorization $u = u_1 \cdots u_n$ such that $u_i \in L_G(X_{a_i})$ for $i = 1, \ldots, n$. By induction, $u_i \in F_{a_i}$ for $i = 1, \ldots, n$. Moreover, the word $a_1 \cdots a_n$ is in the surface S_a. This means that there exist words u'_i in F_{a_i} such that the word $w' = au'_1 \cdots u'_n\bar{a}$ is in F_a. Let g, d be two words such that $gw'd$ is in the language L. Then the pair $(ga, u'_2 \cdots u'_n\bar{a}d)$ is a context for the word u'_1. By (ii), it is also a context for u_1. Thus $au_1u'_2 \cdots u'_n\bar{a}$ is in F_a. Proceeding in this way, one strips off all primes in the u's, and eventually $au_1u_2 \cdots u_n\bar{a}$ is in F_a. Thus w is in F_a. This proves the inclusion and therefore

the equality. Finally, by Corollary 3.4, one has $L_G(X_\alpha) = L$, and thus we have shown that the conditions are sufficient.

We now show that the conditions are necessary. Let G be an XML-grammar generating L, with productions $X_a \to aR_a\bar{a}$ and axiom X_α. Clearly, L is a subset of D_α. Next, consider words $w, w' \in F_a$ for some letter a, and let (g, d) be a context for w. Thus $gwd \in L$. By Lemma 3.3, we know that $F_a = L_G(X_a)$. Thus, there exist derivations $X_a \xrightarrow{*} w$ and $X_a \xrightarrow{*} w'$. Substituting the second to the first in

$$X_\alpha \xrightarrow{*} gX_ad \xrightarrow{*} gwd$$

shows that (g, d) is also a context for w'. This proves condition (ii).

Finally, since R_a is a regular set, the set S_a is also regular. □

Example 4.5. Consider the language L of Example 4.4. The construction of the proof of the theorem gives the XML-grammar

$$\begin{aligned} X_a &\to a(X_a|X_b)(X_a|X_b)\bar{a} \\ X_b &\to b\bar{b} \end{aligned}$$

Example 4.6. The language

$$\{a(b\bar{b})^n(c\bar{c})^n\bar{a} \mid n \geq 1\}$$

already given above is not XML since the surface of a is the nonregular set $S_a = \{b^nc^n \mid n \geq 1\}$. This is the formalization of the example given in the introduction, if the tag b means bold paragraphs, and the tag c means italic paragraphs.

Example 4.7. Consider again the language

$$L = \{ab^{2n}\bar{b}^{2n}\bar{a} \mid n \geq 1\}$$

of Example 3.1. We compute some contexts. First $C_L(b\bar{b}) = \{(ab^{2n-1}, a\bar{b}^{2n-1}\bar{a}) \mid n \geq 1\}$. Next $C_L(b^2\bar{b}^2) = \{(ab^{2n}, a\bar{b}^{2n}\bar{a}) \mid n \geq 0\}$. Thus there are factors with distinct contexts. This shows again that the language is not XML.

Finally, we give an example showing that XML-languages are not closed under union nor difference.

Example 4.8. Consider the sets $cL\bar{c}$ and $cM\bar{c}$, where $L = D^*_{\{a,b\}}$ is the set of products of Dyck primes over $\{a, b\}$, and $M = D^*_{\{a,d\}}$ is the set of products of Dyck primes over $\{a, d\}$. Each of these two languages is XML. However, the union $H = L \cup M$ is not. Indeed, the words $cab\bar{b}\bar{a}\bar{c}$ and $ca\bar{a}d\bar{d}\bar{c}$ are both in H. The pair $(c, d\bar{d}\bar{c})$ is in the context of $a\bar{a}$, so it has to be in the context of $ab\bar{b}\bar{a}$, but the word $cab\bar{b}\bar{a}d\bar{d}\bar{c}$ is not in H. Since XML-languages are closed under intersection, this proves that they are not closed under difference.

5 Decision Problems

As usual, we assume that languages are given in an effective way, in general by a grammar or an XML-grammar, according to the assumption of the statement.

Some properties of XML-languages, such as inclusion or equality (Proposition 3.10) are easily decidable because they reduce to decidable properties of regular sets. The problem is different if one asks whether a context-free grammar generates an XML-language. We have already seen in Example 4.4 that there exist context-free grammars that generate XML-languages without being XML-grammars. The following proposition and its proof are an extension of a result proved by Knuth [3] for parenthesis grammars.

Proposition 5.1. *Given a context-free language L over the alphabet $A \cup \bar{A}$, it is decidable whether $L \subset D_A$.*

In the same paper, Knuth proves also that it is decidable whether a context-free grammar generates a parenthesis language. In contrast to this decidability result, we have.

Proposition 5.2. *It is undecidable whether a context-free language is an XML-language. It is undecidable whether the surfaces of a context-free language are regular.*

This proposition is one reason to consider finite surfaces. Also, the associated XML-grammar is then a context-free grammar in the strict sense, that is with a finite number of productions for each nonterminal. Finally, XML-grammars with finite surfaces are very close to families of grammars that were studied a long time ago. They will be described in the historical note below. The main result is the following.

Theorem 5.3. *Given a context-free language L that is a subset of a set of Dyck primes, it is decidable whether L has all its surfaces finite.*

Conclusion. Given an ordinary context-free grammar G over $A \cup \bar{A}$, we have seen that it is decidable whether $L(G) \subset D_a$ for some letter $a \in A$. If the inclusion holds, we have shown that it is decidable whether $L(G)$ has finite surfaces. If this holds, one may proceed further. A generalization of an argument of Knuth [3] allows to build a balanced grammar G' such that $L(G) = L(G')$. Finally, using this grammar and extending a result of McNaughton [5], one can decide whether the language $L(G')$ is an XML-language.

6 Historical Note

There exist several families of context-free grammars related to XML-grammars that have been studied in the past. In the sequel, the alphabet of nonterminals is denoted by V.

Parenthesis grammars. These grammars have been studied by McNaughton [5] and by Knuth [3]. A *parenthesis grammar* is a grammar with terminal alphabet $T = B \cup \{a, \bar{a}\}$, and where every production is of the form $X \longrightarrow am\bar{a}$, with $m \in (B \cup V)^*$. A parenthesis grammar is *pure* if $B = \emptyset$. In a parenthesis grammar, every derivation step is marked, but there is only one kind of tag.

Bracketed grammars. These were investigated by Ginsburg and Harrison in [1]. The terminal alphabet is of the form $T = A \cup \bar{B} \cup C$ and productions are of the form $X \longrightarrow am\bar{b}$, with $m \in (V \cup C)^*$. Moreover, there is a bijection between A and the set of productions. Thus, in a bracketed grammar, every derivation step is marked, and the opening tag identify the production that is applied (whereas in an XML-grammar they only give the nonterminal).

Very simple grammars. These grammars were introduced in [4], and studied in depth later on. Here, the productions are of the form $X \longrightarrow am$, with $a \in A$ and $m \in V^*$. In a simple grammar, the pair (a, m) determines the production, and in a very simple grammar, there is only one production for each a in A.

Chomsky-Schützenberger grammars. These grammars are used in the proof of the Chomsky-Schützenberger theorem (see e. g. [2]), even if they were never studied for their own. Here the terminal alphabet is of the form $T = A \cup \bar{A} \cup B$, and the productions are of the form $X \longrightarrow am\bar{a}$. Again, there is only one production for each letter $a \in A$.

XML-grammars differ from all these grammars by the fact that the set of productions is not necessarily finite, but regular. However, one could consider a common generalization, by introducing *balanced grammars*. In such a grammar, the terminal alphabet is $T = A \cup \bar{A} \cup B$, and productions are of the form $X \longrightarrow am\bar{a}$, with $m \in (V \cup B)^*$. Each of the parenthesis grammars, bracketed grammars, Chomsky-Schützenberger grammars are balanced. If $B = \emptyset$, such a *pure* grammar covers XML-grammars with finite surfaces. If the set of productions of each nonterminal is allowed to be regular, one gets a new family of grammars with interesting properties.

References

1. S. Ginsburg and M. A. Harrison. Bracketed context-free languages. *J. Comput. Syst. Sci.*, 1:1–23, 1967.
2. Michael A. Harrison. *Introduction to Formal Language Theory.* Addison-Wesley, Reading, Mass., 1978.
3. D. E. Knuth. A characterization of parenthesis languages. *Inform. Control*, 11:269–289, 1967.
4. A. J. Korenjak and J. E. Hopcroft. Simple deterministic grammars. In *7th Switching and Automata Theory*, pages 36–46, 1966.
5. R. McNaughton. Parenthesis grammars. *J. Assoc. Mach. Comput.*, 14:490–500, 1967.
6. W3C Recommendation REC-xml-19980210. *Extensible Markup Language (XML) 1.0*, 10 February 1998. `http://www.w3.org/TR/REC-XML`.
7. W3C Working Draft. *Canonical XML Version 1.0*, 15 November 1999. `http://www.w3.org/TR/xml-c14n`.