МАТЕМАТИКА И МАТЕМАТИЧЕСКО ОБРАЗОВАНИЕ, 2004
MATHEMATICS AND EDUCATION IN MATHEMATICS, 2004
Proceedings of the Thirty Third Spring Conference of
the Union of Bulgarian Mathematicians
Borovets, April 1–4, 2004

SYNTHETIC XML DATA

Pavel K. Azalov, Fani I. Zlatarova

This paper discusses an approach for Extended Markup Language (XML) document synthesis. It is presumed that the structure of the XML documents is determined by a corresponding description that represents a document type definition with some restrictions. The approach proposed in the paper is used to implement a prototype system generating XML documents.

1. Introduction. The synthetic generation of data is appropriate in many cases. The application area of such data comprises not only the scientific research and analysis of the real program system features, but the teaching activity, also [4]. The data generated can be of different natures: program texts, tables, etc. The XML is emerging as the standard for data exchange on the Web. Because of this, the automated synthesis of XML documents is a topic of interest in the recent research.

The development and testing of systems processing XML documents imposes the generation of synthetic XML data. A similar problem also exists when researching the possibility for processing of XML documents to be accomplished by existing relational database systems.

Problem Definition. The main result presented in this paper consists in the description of an approach for synthetic XML document generation. The generation procedure is based on a description that could be considered to be a document type definition (DTD) with some constraints, and we therefore (by slight abuse) will continue to use this term. The basic constraints are reduced to the following:

– Recursion is not permitted in DTDs;
– DTDs do not include entity definitions;
– Mixed-content elements are not considered in DTDs.

The XML documents created in this way differ from each other. Random numbers control the process of generation. In general, the same DTD leads to production of different XML document sets. Therefore, the problem solved below is formulated as follows:

Problem: Suppose that t is a DTD and $S(t)$ is the set of all XML documents that conforms to t. Develop a procedure for random generation of a finite set S'(t) consisting of XML documents conforming to t, i.e. $S'(t) \subseteq S(t)$.

The maximum number of different XML documents having the same DTD is determined by both the corresponding DTD and the current values of certain parameters. This is discussed in detail in Section 4.

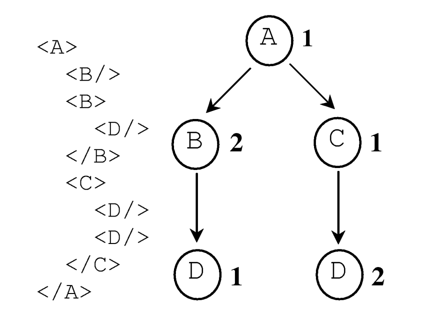
Related Work. There exist two basic approaches for XML document synthesis. The idea of the first approach consists in generating a tree that specifies the structure of the XML data. The idea of the second approach recommends using a DTD to generate XML documents that conform to it. The generator of XML documents developed by IBM [2] is of the second type. A variance of the first approach is presented in [1]. The so-called *path tree* is generated first. The next steps of the process assign a corresponding name and a positive integer, used as a node frequency, to each of the nodes. Figure 1 illustrates the example described in [1].

It shows the respective path tree and an XML document whose structure is determined by this path tree. A number beside the nodes indicates the corresponding node frequency. Two parameters of the generation process exist:

– the number of levels in the path tree;

– the minimum and maximum number of children for nodes at this level, considered for every level of the path tree, except the lowest level.

An essential advantage of this approach is the possibility to generate a path tree of random complexity.



```
<A>
  <B/>
  <B>
    <D/>
  </B>
  <C>
    <D/>
    <D/>
  </C>
</A>
```

Figure 1

2. Preliminaries. XML Trees. Our abstraction for an XML document is an XML tree. XML trees are labeled ordered trees. Every *node* of an XML tree is determined by its name and content. The *name* is a symbol from some finite *alphabet* Σ and is considered as a label of this node. The *content* represents a list (a totally ordered set, eventually empty) of children, each of them being an XML tree. An example of an XML tree is given in Figure 2.

XML trees have a commonly accepted linear representation as strings of elements [3]. Each of these elements is described through its *opening* and *closing tags.*

Suppose that $a$is a symbol of the Σ alphabet, $a \in \Sigma$. The corresponding opening and closing tags are noted here respectively as a and a°. The hierarchical structure from Figure 2 can then be represented in a linear form through the following string:

(1) $dUamm^\circ yy^\circ a^\circ U^\circ Namm^\circ a^\circ N^\circ d^\circ$

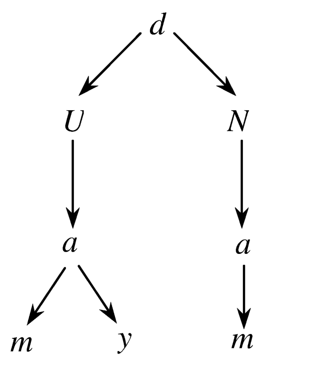


Figure 2

The DTD provides a typing mechanism for XML documents. DTDs use regular expressions to describe the allowed sequences of children of a node. Figure 3 shows the example considered in [6] with an insignificant modification of a DTD. This DTD matches the XML tree from Figure 2. According to [5], “the optional character following a name or list governs whether the element or the content particles in the list may occur one or more (+), zero or more (*), or zero or one times (?). The absence of such an operator means that the element or content particle must appear exactly once. Any content particle in a *choice* list ("|") may appear in the element content at the location where the choice list appears in the grammar. Content particles occurring in a *sequence* list (",") must each appear in the element content in the order given in the list."

DTD Expression Tree. Our abstraction for a DTD is a DTD expression tree. A *DTD expression tree* (DTDET) is a binary tree that contains all operators and names, determined in the corresponding DTD. The root and the internal nodes are operators, and the leaves are names. In the example given in Figure 3, the operators are "+", "*" (unary operators), "|", "," (binary operators), and the names are "dealer" (d), "UsedCars" (U), "NewCars" (N), "ad" (a), "model" (m), and "year" (y).

An XML tree over Σ *satisfies* a DTD named d if it is derived from the corresponding DTD expression tree [3].

3. Internal DTD Representation. The XML data synthesis is accomplished in a sequence of steps whose description is offered below. Transformations of the DTD occur first. As a result, the corresponding DTDET is obtained. These transformations are illustrated through the example from Figure 3 with certain modifications.

```
<DOCTYPE dealer [
  <!ELEMENT dealer (UsedCars, NewCars)>
  <!ELEMENT UsedCars (ad*)>
  <!ELEMENT NewCars (ad+)>
  <!ELEMENT ad ((model, year) | model)>
 ]>
```

Figure 3

DTD Table Representation. A text file t containing a DTD represents the input data for the procedure considered below. The data provided by this file is used for the construction of two tables: the element table ET(t) and attribute table AT(t). Each element declared in t is included in ET(t) together with the set (eventually empty) of its corresponding attributes. The attributes and their values are given in AT(t) table. The relationship between both tables is established through the attribute names.

First DTD String Representation. The right-hand sides of productions in a DTD are regular expressions over the terminals and non-terminals. The set consisting of four productions (Figure 3) is transformed into a single expression through multiple nesting in the following way:

(2) $$d(U(a^*((m,y)|m)),N(a+((m,y)|m))))$$

where only the first letters of the respective names were used. In this expression, there are included two binary operations, "," and "|", and two unary operations, "*" and "+". In the general case, one more unary operation is available, "?". By applying the unary operator "*", the expression (2) is transformed to the following expression:

(3) $$d(U((a((m,y)|m))^*),N((a((m,y)|m))+))$$

Second DTD Strings Representation. In expression (3), the binary operation linking the name of an element with its definition is implicitly used. If this operation is noted by “$\rightarrow$”, then expression (3) can be rewritten in the following way:

(4) $$(d \rightarrow (U \rightarrow ((a \rightarrow ((m, y)|m))^*), N \rightarrow ((a \rightarrow ((m, y)|m))+)))$$

DTD Tree Representation. The last expression (4) could be transformed in a DTDET. This could be accomplished in different ways. One of them is to use the Polish Notation first. The postfix form of this expression is the following:

(5) $$d U \, a \, m \, y, \, m| \rightarrow^* \rightarrow N \, a \, m \, y, \, m| \rightarrow + \rightarrow, \rightarrow \$$$

The unary operators are of highest priority. They are followed by the “$\rightarrow$” operator. The lowest priority is assigned to the “|” and “,” operators which are left associative.

Figure 4 shows the final result from the transformation of the DTD, considered in Figure 3, into a DTDET.

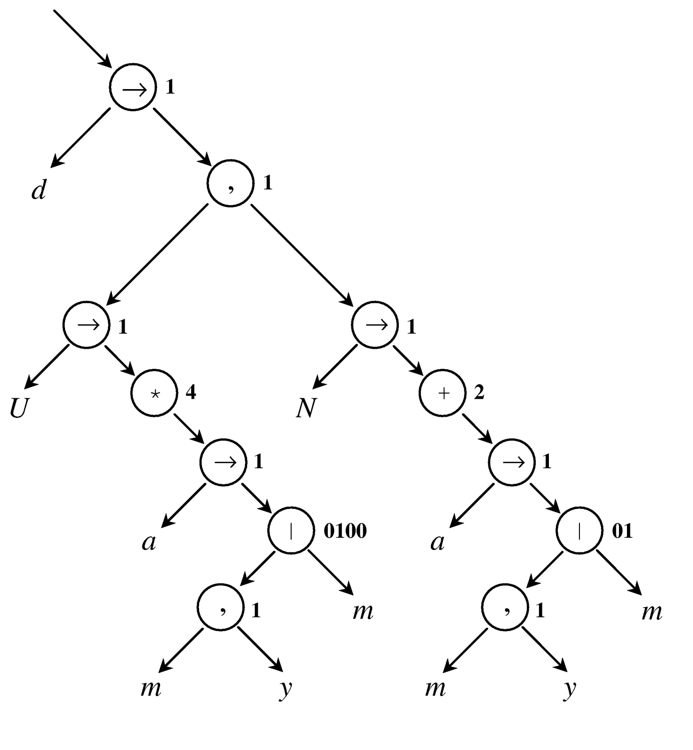
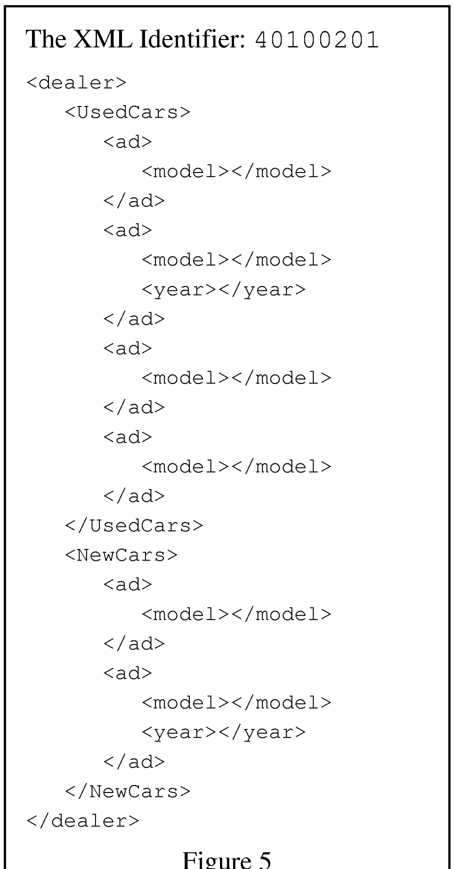


Figure 4. DTD expression tree

4. Generating XML Data. Randomization of XML Data Generation. A pair of nonnegative integers min and max is assigned to each inner node during the DTDET generation. The integer min has a determined value for each of the operator types. For example, its value is 0, 0, and 1 for the respective operators “?”, “*”, and “+”. These integers indicate the minimum number of repetitions for each unary operation. The max integer is generated randomly according to the constraint min $\leq$ max $\leq m$, where m is a parameter. This parameter has a given value determined by default, but it can take another value during the execution of the system. For unary operators, the pair of integers min and max determines the repetition coefficient of the subtree whose root is the unary operation considered. During the XML document generation, a 0 or 1 is assigned in a random way to the binary operator "|" each time it is met. The corresponding value 0 or 1 will determine which of both subtrees should be selected. The value of both integers min and max corresponding to the binary operators "**,**" and "$\rightarrow$" is always 1. It means that the trees having such roots are traversed only once. In this way, the generation of different XML documents is accomplished using the repetition coefficients introduced above and a modified procedure for inorder traversal of a DTDET.


The XML Identifier: `40100201`

```
<dealer>
   <UsedCars>
      <ad>
         <model></model>
      </ad>
      <ad>
         <model></model>
         <year></year>
      </ad>
      <ad>
         <model></model>
      </ad>
      <ad>
         <model></model>
      </ad>
   </UsedCars>
   <NewCars>
      <ad>
         <model></model>
      </ad>
      <ad>
         <model></model>
         <year></year>
      </ad>
   </NewCars>
</dealer>
```

Figure 5

The procedure, generating random XML data, is controlled by a special variable whose values are random numbers, too. When the system is activated at the very beginning, it generates the seed for this variable first. To avoid the external factor, when obtaining the seed, an expression, including the current time, is calculated. At the end of the execution, the current value of this variable is stored in a file. Later, it is used as the new seed for the generation of the next XML data set. A similar approach was used in [4].

XML Identifiers. The numbers associated to each of the inner nodes in the XML document generation process identify in a unique way each XML document obtained.

For example, given the sequence of numbers from Figure 4, it represents a number with base m. It will be the following: 1 1 1 $\underline{\mathbf{4}}$ 1 $\underline{0}$ $\underline{1}$ $\underline{0}$ $\underline{0}$ 1 $\underline{\mathbf{2}}$ 1 $\underline{0}$ $\underline{1}$ and is obtained through a modified procedure of inorder traversal of the respective DTDET. The digits, which are not underlined, are the same for all XML documents generated by the same execution. This allows to determine the number build only of the underlined digits **4** 0 1 0 0 **2** 0 1. This number is considered to represent the identifier of the corresponding XML document (Figure 5). The bolded digits indicate unary operators. For example, the digit **4** is the repetition coefficient of the subtree, whose root is the "*" operator. The next four digits, 0 1 0 0, are assigned to the "|" node. They indicate that first, the right subtree of "|" is selected. After that, its left subtree is selected. In the next two steps, the left subtree is selected. The digit **2** indicates the unary operator "+", and the sequence of the next two digits, 0 1, indicates the "|" operator.

5. XML-DG: AN XML Data Generator. The idea described here is used to develop a prototype of a system, XML-DG that generates XML documents. First, the DTD is input from a text file and is transformed into two tables: elements table and attributes table. After a sequence of other transformations, the DTD is converted into a DTDET. It is used to generate XML documents stored as text files that contain the structure of the respective documents.

REFERENCES

[1] J. Aboulnaga, J. Naughton, C. Zhang. Generating Synthetic Complex-structured XML Data, The Niagara Project: `http://www.cs.wisc.edu/niagara`, 2002.

[2] IBM XML generator, `http://www.alphaworks.ibm.com/tech/xmlgenerator`, 2001.

[3] L. Segouflin, V. Vianu. Validating Streaming XML Documents. Madison, WI: ACM PODS, 2002.

[4] P. Azalov, F. Zlatarova. SDG – A System for Synthetic Data Generation. IEEE Computer Society: Proceedings of the International Conference on Information Technology: Coding and Computing, Las Vegas, NV, April 2003, 69–75.

[5] T. Bray, J. Paoli, C. M. Sperberg-McQueen. Extensible Markup Language (XML) 1.0, `http://www.w3.org/TR/REC-xml.`

[6] Y. Papakonstantinou, V. Vianu. DTD Inference for Views of XML Data. Dallas, TX: ACM POD, 2000.

Pavel K. Azalov
Engineering Division, CWC
Pennsylvania State University
Hazleton, PA 18202, USA

Fani I. Zlatarova
Computer Science Department
Elizabethtown College
Elizabethtown, PA 17022, USA

СИНТЕЗ НА XML

Павел К. Азълов, Фани Й. Златарева

В настоящата статия е представен подход за синтез на XML (Extended Markup Language) данни. Предполага се, че структурата на XML данните е определена от съответна дефиниция на типа на документа (DTD) с известни ограничения. Предлаганият подход е използван за реализация на прототип на програмна система за генериране на XML данни.