XML Data – The Current State of Affairs

Kamil Toman and Irena Mlynkova

{kamil.toman,irena.mlynkova}@mff.cuni.cz

Charles University
Faculty of Mathematics and Physics
Department of Software Engineering
Malostranske nam. 25
118 00 Prague 1, Czech Republic

Abstract. At present the eXtensible Markup Language (XML) is used almost in all spheres of human activities. Its popularity results especially from the fact that it is a self-descriptive metaformat that allows to define the structure of XML data using other powerful tools such as DTD or XML Schema. Consequently, we can witness a massive boom of techniques for managing, querying, updating, exchanging, or compressing XML data.
On the other hand, for majority of the XML processing techniques we can find various spots which cause worsening of their time or space efficiency. Probably the main reason is that most of them consider XML data too globally, involving all their possible features, though the real data are often much simpler. If they do restrict the input data, the restrictions are often unnatural.
In this contribution we discuss the level of complexity of real XML collections and their schemes, which turns out to be surprisingly low. We involve and compare results and findings of existing papers on similar topics as well as our own analysis and we try to find the reasons for these tendencies and their consequences.

1 Introduction

Currently XML and related technologies [7] have already achieved the leading role among existing standards for data representation and are used almost in all spheres of human activities. They are popular for various reasons, but especially because they enable to describe the allowed structure of XML documents using powerful tools such as DTD [7] or XML Schema [9, 20, 6]. Thus we can witness a massive boom of various XML techniques for managing, processing, exchanging, querying, updating, and compressing XML data that mutually compete in speed, efficiency, and minimum space and/or memory requirements.

On the other hand, for majority of the techniques we can find various critical spots which cause worsening of their time and/or space efficiency. In the worst and unfortunately quite often case such bottlenecks negatively influence directly the most interesting features of a particular technique.

If we study the bottlenecks further, we can distinguish two typical problematic situations. Firstly, we can distinguish a group of general techniques that take into account all possible features of input XML data – an approach that is at first glance correct. Nevertheless the standards were proposed as generally as possible enabling future users to choose what suits them most, whereas the real XML data are usually not as "rich" as they could be – they are often surprisingly simple. Thus the effort spent on every possible feature is mostly useless and it can even be harmful.

Secondly, there are techniques that do restrict features of input XML data in some way. Hence it is natural to expect the bottlenecks to occur only in situations when given data do not correspond to the restrictions. The problem is that such restrictions are often "unnatural". They do not result from inherent features of real XML data collections but from other, more down-to-earth, reasons, e.g. limitations of the basic proposal of a particular technique, complexity of such solution etc.

A solution to the given problems could be a detailed analysis of real XML data and their classification. Up to now, there are several works which analyze real XML collections from various points of view. All the papers have the same aim – to describe typical features and complexity of XML data – and all conclude that the real complexity is low indeed. In this paper we briefly describe, discuss, and compare results and findings of the papers as well as our own analysis. We try to find the reasons for these tendencies, their consequences and influence on future processing.

The paper is structured as follows: The first section introduces the considered problems. The following, second, section contains a brief overview of formalism used throughout the paper. Section 4 classifies, describes, and discusses XML data analyses. The last, fifth, section provides conclusions.[1]

2 Formal Definitions

For structural analysis of XML data it is natural to view XML documents as ordered trees and DTDs or XSDs (i.e. XML Schema definitions) as sets of regular expressions over element names. Attributes are often omitted for simplicity. We use notation and definitions for XML documents and DTDs from [8] and [5]. (For XSDs are often used the same or similar ones – we omit them for the paper length.)

Definition 1. An XML document *is a finite ordered tree with node labels from a finite alphabet Σ. The tree is called Σ-*tree.

Definition 2. A DTD *is a collection of element declarations of the form $e \rightarrow \alpha$ where $e \in \Sigma$ is an element name and α is its content model, i.e. regular*

[1] We will not describe neither the basics of XML, DTD, or XML Schema. We suppose that XML and DTD have already become almost a common knowledge whereas description of XML Schema is omitted for the paper length.

expression over Σ. The content model α is defined as $\alpha := \epsilon \mid pcdata \mid f \mid \alpha_1, \alpha_2, ..., \alpha_n \mid \alpha_1|\alpha_2|...|\alpha_n \mid \beta* \mid \beta+ \mid \beta?$, where ϵ denotes the empty content model, pcdata denotes the text content, f denotes a single element name, "," and "|" stand for concatenation and union (of content models $\alpha_1, \alpha_2, ...\alpha_n$), and "", "+", and "?" stand for zero or more, one or more, and optional occurrence(s) (of content model β). One of the element names $s \in \Sigma$ is called* a start symbol.

Definition 3. *A Σ-tree* satisfies the DTD *if its root is labeled by start symbol s and for every node n and its label e, the sequence $e_1, e_2, ...e_k$ of labels of its child nodes matches the regular expression α, where $e \rightarrow \alpha$.*

Basic analyses of XML data usually focus on depth of content models and/or XML documents, reachability of content models and/or elements, types of recursion, types of paths and cycles, fan-ins and fan-outs. They are usually similar for both XML documents and XML schemes (regardless the used language).

Definition 4. Depth of a content model α *is inductively defined as follows:*
$depth(\epsilon) = 0$;
$depth(pcdata) = depth(f) = 1$;
$depth(\alpha_1, \alpha_2, ..., \alpha_n) = depth(\alpha_1|\alpha_2|...|\alpha_n) = max(depth(\alpha_i)) + 1;\ 1 \leq i \leq n$
$depth(\beta*) = depth(\beta+) = depth(\beta?) = depth(\beta) + 1$.

Definition 5. Distance of elements e_1 *and* e_2 *is the number of edges in Σ-tree separating their corresponding nodes.*

Level of an element *is distance of its node from the root node. The level of the root node is 0.*

Depth of an XML document *is the largest level among all the elements.*

Definition 6. *An element name e' is* reachable from *e, denoted by $e \Rightarrow e'$, if either $e \rightarrow \alpha$ and e' occurs in α or $\exists\ e''$ such that $e \Rightarrow e''$ and $e'' \Rightarrow e'$.*

A content model α is derivable*, denoted by $e \Rightarrow \alpha$, if either $e \rightarrow \alpha$ or $e \Rightarrow \alpha'$, $e' \rightarrow \alpha''$, and $\alpha = \alpha'[e'/\alpha'']$, where $\alpha'[e'/\alpha'']$ denotes the content model obtained by substituting α'' for all occurrences of e' in α'.*

An element name e is reachable*, if $r \Rightarrow e$, where r is the name of root element. Otherwise it is called* unreachable.

Definition 7. *An element e is* recursive *if there exists at least one element d in the same document such that d is a descendant of e and d has the same label as e.*

The element-descendant association is called an ed-pair.

Definition 8. *An element e is called* trivially recursive *if it is recursive and for every α such as $e \Rightarrow \alpha$ e is the only element that occurs in α and neither of its occurrences is enclosed by "*" or "+".*

An element e is called linearly recursive *if it is recursive and for every α such as $e \Rightarrow \alpha$ e is the only recursive element that occurs in α and neither of its occurrences is enclosed by "*" or "+".*

An element e is called purely recursive *if it is recursive and for every α such as $e \Rightarrow \alpha$ e is the only recursive element that occurs in α.*

An element that is not purely recursive is called generally recursive *element.*

Definition 9. Simple path *(in a non-recursive DTD) is a list of elements* e_1, e_2,... e_k, *where* $e_i \rightarrow \alpha_i$ *and* e_{i+1} *occurs in* α_i *for* $1 \leq i < k$. *Parameter* k *is called* length of a simple path.

Simple cycle *is a path in the form* e_1, e_2,... e_k, e_1, *where* e_1, e_2,... e_k *are distinct element names.*

Chain of stars *is a a simple path of elements* e_1, e_2,... e_{k+1}, *where* e_{i+1} *is in the corresponding* α_i *followed by "*" or "+" for* $1 \leq i \leq k$. *Parameter* k *is called* length of a chain of stars.

Definition 10. Fan-in *of an element* e *is the cardinality of the set* $\{e' \mid e' \rightarrow \alpha'$ *and* e *occurs in* $\alpha'\}$. *An element name with large fan-in value is called* hub.

Definition 11. Element fan-out *of element* e *is the cardinality of the set* $\{e' \mid e \rightarrow \alpha$ *and* e' *occurs in* $\alpha\}$.

Minimum element fan-out *of element* e *is the minimum number of elements allowed by its content model* α.

Maximum element fan-out *of element* e *is the maximum number of elements allowed by content model* α.

Attribute fan-out *of element* e *is the number of its attributes.*

There are also XML constructs, that can be called advanced, such as types of mixed content, DNA patterns, or relational patterns.

Definition 12. *An element is called* trivial *if it has an arbitrary amount of attributes and its content model* $\alpha := \epsilon \mid pcdata$.

A mixed content of element is called simple *if it consist only of trivial elements. Otherwise it is called* complex.

Definition 13. *An nonrecursive element* e *is called* DNA pattern *if its content model* α *is not mixed and consists of a nonzero amount of trivial elements and just one nontrivial and nonrecursive element which is not enclosed by "*" or "+". The nontrivial subelement is called degenerated branch.*

Depth *of a DNA pattern* e *is the maximum depth of its degenerated branch.*

Definition 14. *A nonrecursive element* e *is called* relational pattern *if it has an arbitrary amount of attributes and its content model* $\alpha := (e_1, e_2, ..., e_n)* \mid (e_1, e_2, ..., e_n)+ \mid (e_1|e_2|...|e_n)* \mid (e_1|e_2|...|e_n)+$ *and it is not mixed.*

A nonrecursive element e *is called* shallow relational pattern *if it has an arbitrary amount of attributes and its content model* $\alpha := f* \mid f+$ *and it is not mixed.*

3 Analyses and Results

Up to now several papers have focused on analysis of real XML data. They analyze either the structure of DTDs, the structure of XSDs, the structure of XML data regardless their schema, or the structure of XML documents in relation to corresponding schema. The sample data usually essentially differ.

3.1 DTD Analysis

Probably the first attempt to analyze the structure of XML data can be found in [18]. The paper is relatively old (especially with regard to the fast development of XML standards) and it contains a quantitative analysis of 12 DTDs and a general discussion of how they are (mis)used.

The analysis involves:

- the size of DTDs, i.e. the number of elements, attributes, and entity references,
- the structure of DTDs, i.e. the number of root elements and depth of content models, and
- some specific aspects, i.e. the use of mixed-content, `ANY`, `ID`s and `IDREF`s, or the kind of attribute decorations used (i.e. `implied`, `required`, and `fixed` attributes).

The discussion of current (mis)using of DTDs brings various conclusions, involving especially shortcomings of DTD. Most of them have already been overcome in XML Schema – e.g. the necessity to use XML itself for description of the structure of XML data, the missing operator for unordered sequences, insufficient tools for inheritance and modularity, the requirement for typed `IDREF`s (i.e. those which cannot refer to any `ID`) etc.

There are also interesting observations concerning structure of the data, especially the finding that content models have the depth less than 6 and that `ID`s and `IDREF`s are not used frequently (probably due to the above mentioned problem with typing). According to the author the most important conclusion is that DTDs are usually incorrect (both syntactically and semantically) and thus are not a reliable source of information.

Second paper [8] that also focuses on DTDs describes analyses which are more statistical than in the previous case. It analyzes 60 DTDs further divided according to their intended future use into three categories:

- *app*, i.e. DTDs designed for data interchange,
- *data*, i.e. DTDs for data that can easily be stored in a database, and
- *meta*, i.e. DTDs for describing the structure of document markup.

The statistics described in the paper focus on graph theoretic properties of DTDs and can be divided into:

- *local*, i.e. describing kinds of content models found at individual element declarations (e.g. the number of mixed-content elements) and
- *global*, i.e. describing graph structure of the DTD (e.g. the maximum path length allowed by the DTD).

Local properties focus on four types of features – content model classifications, syntactic complexity, non-determinism, and ambiguity. The classification of content models involves *pcdata*, ϵ, *any*, mixed content, "|" only (but not mixed) content, "," only content, complex content (i.e. with both "|"s and ","s), list content (i.e. the usage of "+" or"*" for one element), and single content (i.e. the optional usage of "?" for one element); the syntactic complexity is expressed by the previously defined *depth* function. The question of both non-determinism and ambiguity (i.e. a special case of non-determinism) of content models is a bit controversial since non-deterministic content models are not allowed by the XML standards. The most important findings for local properties are that the content model of DTDs is usually not complex (the maximum depth is 9, whereas its mode is even 3) and that despite the standards, there are both non-deterministic and ambiguous content models.

Global properties discussed in the paper involve reachability, recursions, simple paths and cycles, chains of stars and hubs. The most important findings are listed below.

- Unreachable elements are either root elements or useless, whereas the mode of their number is 1, i.e. the root element is usually stated clearly.
- There are no linear recursive elements, whereas the number of non-linear recursive elements is significant (i.e. they occur in 58% of all DTDs).
- The maximum length of simple path is surprisingly small (mostly less than 8), whereas on the other hand the number of simple paths as well as simple cycles is either small (less than 100) or large (more than 500).
- The length of the longest chain of stars is usually small (its mode is 3).
- Hubs exist in all categories od DTDs and their number is significant.

Last found paper [12] which focuses on DTD analysis is trying to adapt software metrics to DTDs. It defines five metrics, also based on their graph representation – i.e. size, complexity, depth, fan-in, and fan-out, whereas all of them were already defined and discussed. Regrettably, there are just 2 DTD examples for which the statistics were counted.

3.2 DTD vs. XML Schema

With the arrival of XML Schema, as the extension of DTD, has arisen a natural question: Which of the extra features of XML Schema not allowed in DTD are used in practise? Paper [5] is trying to answer it using analysis of 109 DTDs and 93 XSDs. Another aim of the paper is to analyze the real structural complexity for both the languages, i.e. the degree of sophistication of regular expressions used.

The former part of the paper focuses on analysis of XML Schema features. The features and their resulting percentage are:

- extension[2] (27%) and restriction (73%) of simple types,
- extension (37%) and restriction (7%) of complex types,

[2] Extension of a simple type means adding attributes to the simple type, i.e. creating a complex type with simple content.

- `final` (7%), `abstract` (12%), and `block` (2%) attribute of complex type definitions,
- substitution groups (11%),
- unordered sequences of elements (4%),
- `unique` (7%) and `key/keyref` (4%) features,
- namespaces (22%), and
- redefinition of types and groups (0%).

As it is evident, the most exploited features are restriction of simple types, extension of complex types, and namespaces. The first one reflects the lack of types in DTD, the second one confirms the naturalness of object-oriented approach (i.e. inheritance), whereas the last one probably results from mutual modular usage of XSDs. The other features are used minimally or are not used at all.

Probably the most interesting finding is, that 85% of XSDs define so called *local tree languages* [17], i.e. languages that can be defined by DTDs as well, and thus that the expressiveness beyond local tree grammars is needed rarely.

3.3 XML Schema Analysis

Paper [5] mentioned in the previous section analyzed the properties of DTDs and XSDs together. Nevertheless its first part focused only on statistical analysis of real usage of new XML Schema features. Paper [14] has a similar aim – it defines 11 metrics of XSDs and two formulae that use the metrics to compute complexity and quality indices of XSDs. The metrics are:

- the number of (both globally and locally defined) complex type declarations, which can be further divided into text-only, element-only, and mixed-content,
- the number of simple type declarations,
- the number of annotations,
- the number of derived complex types,
- the average number of attributes per complex type declaration,
- the number of global (both simple and complex) type declarations,
- the number of global type references,
- the number of unbounded elements,
- the average bounded element multiplicity size, where multiplicity size is defined as *(maxOccurs - minOccurs + 1)*,
- the average number of restrictions per simple type declaration,
- *element fanning*, i.e. the average fan-in/fan-out.

On the basis of the experience in analyzing many XSDs the authors define two indices for expressing their quality and complexity.

Definition 15. Quality Index = *(Ratio of simple to complex type declarations) * 5 + (Percentage of annotations over total number of elements) * 4 + (Average restrictions per simple type declarations) * 4 + (Percentage of derived complex type declarations over total number of complex type declarations) * 3 (Average* *bounded element multiplicity size) * 2 (Average attributes per complex type declaration) * 2*

Complexity Index = *(Number of unbounded elements) * 5 + (Element fanning) * 3 + (Number of complex type declarations) + (Number of simple type declarations) + (Number of attributes per complex type declaration)*

Unfortunately, there is just 1 XSD example for which the statistics were counted.

3.4 XML Document Analysis

Previously mentioned analyses focused on descriptions of the allowed structure of XML documents. By contrast paper [15] (and its extension [2]) analyzes directly the structure of their instances, i.e. XML documents, regardless eventually existing DTDs or XSDs.[3] It analyzes about 200 000 XML documents publicly available on the Web, whereas the statistics are divided into two groups – statistics about the Web and statistics about the XML documents.

The Web statistics involve:

- clustering of the source web sites by zones consisting of Internet domains (e.g. *.com*, *.edu*, *.net* etc.) and geographical regions (e.g. Asia, EU etc.),
- the number and volume (i.e. the sum of sizes) of documents per zone,
- the number of DTD (48%) and XSD (0.09%) references,
- the number of namespace references (40%),
- distribution of files by extension (e.g. *.rdf*, *.rss*, *.wml*, *.xml* etc.), and
- distribution of document *out-degree*, i.e. the number of `href`, `xmlhref`, and `xlink:href` attributes.

Obviously most of them describe the structure of the XML Web and categories of the source XML documents.

Statistics about the structure of XML documents involve:

- the size of XML documents (in bytes),
- the amount of markup, i.e. the amount of element and attribute nodes versus the amount of text nodes and the size of text content versus the size of the structural part,
- the amount of mixed content elements,
- the depth of XML documents and the distribution of node types (i.e. element, attribute, or text nodes) per level,
- element and attribute fan-out
- the number of distinct strings, and
- recursion.

The most interesting findings of the research are as follows:

[3] The paper just considers whether the document does or does not reference a DTD or an XSD.

– The size of documents varies from 10B to 500kB; the average size is 4,6kB.
– For documents up to 4kB the number of element nodes is about 50%, the number of attribute nodes about 30%. Surprisingly, for larger documents the number of attribute nodes rises to 50%, whereas the number of element nodes declines to 38%. The structural information still dominates the size of documents.
– Although there are only 5% of all elements with mixed content, they were found in 72% of documents.
– Documents are relatively shallow – 99% of documents have fewer than 8 levels, whereas the average depth is 4.
– The average element fan-out for the first three levels is 9, 6, and 0.2; the average attribute fan-out for the first four levels is 0.09, 1, 1.5, and 0.5. Surprisingly, 18% of all elements have no attributes at all.

A great attention is given to recursion which seems to be an important aspect of XML data. The authors mention the following findings:

– 15% of all XML documents contain recursive elements.
– Only 260 distinct recursive elements were found. In 98% of recursive documents there is only one recursive element used.
– 95% of recursive documents do not refer to any DTD or XSD.
– Most elements in ed pairs have the distance up to 5.
– The most common average fan-outs are 1 (60%) and 2 (37%), the average recursive fan-out is 2.2.

Lastly, paper [13] that focuses on analysis of XML documents consists of two parts – a discussion of different techniques for XML processing and an analysis of real XML documents. The sample data consists of 601 XHTML web pages, 3 documents in DocBook format[4], an XML version of Shakespeare's plays[5] (i.e. 37 XML documents with the same simple DTD) and documents from *XML Data repository* project[6]. The analyzed properties are the maximum depth, the average depth, the number of simple paths, and the number of unique simple paths; the results are similar to previous cases.

3.5 XML Documents vs. XML schemes

The work initiated in the previously mentioned articles is taken up recently by paper [16]. It enhances the preceding analyses and defines several new constructs for describing the structure of XML data (e.g. DNA or relational patterns). It analyzes XML documents together with their DTDs or XSDs eventually that were collected semi-automatically with interference of human operator. The reason is that automatic crawling of XML documents generates a set of documents that are unnatural and often contain only trivial data which cause misleading

[4] `http://www.docbook.org/`
[5] `http://www.ibiblio.org/xml/examples/shakespeare/`
[6] `http://www.cs.washington.edu/research/xmldatasets/`

results. The collected data consist of about 16 500 XML documents of more than 20GB in size, whereas only 7.4% have neither DTD nor XSD. Such low ratio is probably caused by the semi-automatic gathering.

The data were first divided into following six categories:

- *data-centric documents*, i.e. documents designed for database processing (e.g. database exports, lists of employees etc.),
- *document-centric documents*, i.e. documents which were designed for human reading (e.g. Shakespeare's plays, XHTML [1] documents etc.)
- *documents for data exchange* (e.g. medical information on patients etc.),
- *reports*, i.e. overviews or summaries of data (usually of database type),
- *research documents*, i.e. documents which contain special (scientific or technical) structures (e.g. protein sequences, DNA/RNA structures etc.), and
- *semantic web documents*, i.e. RDF [4] documents.

The statistics described in the paper are also divided into several categories. They were computed for each category and if possible also for both XML documents and XML schemes and the results were compared. The categories are as follows:

- *global statistics*, i.e. overall properties of XML data (e.g. number of elements of various types such as empty, text, mixed, recursive etc., number of attributes, text length in document, paths and depths etc.),
- *level statistics*, i.e. distribution of elements, attributes, text nodes, and mixed contents per each level,
- *fan-out statistics*, i.e. distribution of branching per each level,
- *recursive statistics*, i.e. types and complexity of recursion (e.g. exploitation rates, depth, branching, distance of ed-pairs etc.),
- *mixed-content statistics*, i.e. types and complexity of mixed contents (e.g. depth, percentage of simple mixed contents etc.),
- *DNA statistics*, i.e. statistics focussing on DNA patterns (e.g. number of occurrences, width, or depth), and
- *relational statistics*, i.e. statistics focussing on both relational and shallow relational patterns (e.g. number of occurrences, width, or fan-out).

Most interesting findings and conclusions for all categories of statistics are as follows:

- The amount of tagging usually dominates the size of document.
- The lowest usage of mixed-content (0.2%) and empty (26.8%) elements can be found in data-centric documents.
- The highest usage of mixed-content elements (77%) can be found in document-centric documents.
- Documents of all categories are typically shallow. (For 95% of documents the maximum depth is 13, the average depth is about 5.)
- The highest amounts of elements, attributes, text nodes, and mixed contents as well as fan-outs are always at first levels and then their number of occurrences rapidly decreases.

– Recursion occurs quite often, especially in document-centric (43%) and exchange (64%) documents, although the number of distinct recursive elements is typically low (for each category less than 5).
– Recursion, if used, is rather simple – the average depth, branching as well as distance of ed-pairs is always less than 10.
– The most common types of recursion are linear (20% for document-centric and 33% for exchange documents) and pure (19% for document-centric and 23% for exchange documents).
– Unlike document instances almost all schemes specify usually only the most general type of recursion.
– The percentage of simple mixed contents is relatively high (e.g. 79% for document-centric or even 99% for exchange documents) and thus the depth of mixed contents is generally low (on the average again less than 10).
– The number of occurrences of DNA patterns is rather high, especially for research, document-centric, and exchange documents. On the other hand the average depth and width is always low (less than 7).
– The number of occurrences of relational patterns is high, especially for semantic-web, research, and exchange documents. The complexity (i.e. depth and width) is again quite low.
– XML schemes usually provide too general information, whereas the instance documents are much simpler and more specific.

3.6 Discussion

The previous overview of existing analyses of XML data brings various interesting information. In general, we can observe that the real complexity of both XML documents and their schemes is amazingly low.

Probably the most surprising findings are that recursive and mixed-content elements are not as unimportant as they are usually considered to be. Their proportional representation is more than significant and in addition their complexity is quite low. Unfortunately, effective processing of both the aspects is often omitted with reference to their irrelevancy. Apparently, the reasoning is false whereas the truth is probably related to difficulties connected with their processing.

Another important discovery is that the usual depth of XML documents is small, the average number is always less than 10. This observation is already widely exploited in techniques which represent XML documents as a set of points in multidimensional space and store them in corresponding data structures, e.g. R-trees [11], UB-trees [3], BUB-trees [10] etc. The effectiveness of these techniques is closely related to the maximum depth of XML documents or maximum number of their simple paths. Both of the values should be of course as small as possible.

Next considerable fact is that the usage of schemes for expressing allowed structures of XML documents is not as frequent as it is expected to be. The situation is particularly wrong for XSDs which seem to appear sporadically. And even if they are used, their expressive power does not exceed the power of

DTDs. The question is what is the reason for this tendency and if we can really blame purely the complexity of XML Schema. Generally, the frequent absence of schema is of course a big problem for methods which are based on its existence, e.g. schema-driven database mapping methods [19].

Concerning the XML schemes there is also another important, though not surprising finding, that XML documents often do not fully exploit the generality allowed by schema definitions. It is striking especially in case of types of recursion but the statement is valid almost generally. Extreme cases are of course recursion that theoretically allows XML documents with infinite depth or complete subgraphs typical for document-centric XML documents. This observation shows that although XML schemes provide lots of structural information on XML documents they can be too loose or even inaccurate.

The last mentioned analysis indicates, that there are also types of constructs (such as simple mixed contents, DNA patterns, or relational patterns etc.), that are quite common and can be easily and effectively processed using, e.g., relational databases. Hence we can expect that a method that focuses on such constructs would be much more effective than the general ones.

Last but not least, we must mention the problem of both syntactic and semantic incorrectness of analyzed XML documents, DTDs, and XSDs. Authors of almost all previously mentioned papers complain of huge percentage of useless sample data – an aspect which unpleasantly complicates the analyses. A consequent question is whether we can include schema non-determinism and ambiguity into this set of errors or if it expresses a demand for extension of XML recommendations.

4 Conclusion

The main goal of this paper was to briefly describe, discuss, and classify papers on analyses of real XML data and particularly their results and findings. The whole overview shows that the real data show lots of regularities and pattern usages and are not as complex as they are often expected to be. Thus there exists plenty of space for improvements in XML processing based on this enhanced categorization.

Acknowledgement

This work was supported in part by the National Programme of Research (Information Society Project 1ET100300419).

References

1. *The Extensible HyperText Markup Language (Second Edition).* W3C Recommendation, August 2002. `http://www.w3.org/TR/xhtml1/`.

2. D. Barbosa, L. Mignet, and P. Veltri. Studying the XML Web: Gathering Statistics from an XML Sample. In *World Wide Web*, pages 413–438, Hingham, MA, USA, 2005. Kluwer Academic Publishers.
3. R. Bayer. The Universal B-Tree for Multidimensional Indexing: General Concepts. In *WWCA '97, Worldwide Computing and Its Applications, International Conference*, pages 198–209, Tsukuba, Japan, 1997. Springer.
4. D. Beckett. *RDF/XML Syntax Specification (Revised).* W3C Recommendation, February 2004. `http://www.w3.org/TR/rdf-syntax-grammar/`.
5. G. J. Bex, F. Neven, and J. Van den Bussche. DTDs versus XML Schema: a Practical Study. In *WebDB '04, Proceedings of the 7th International Workshop on the Web and Databases*, pages 79–84, New York, NY, USA, 2004. ACM Press.
6. P. V. Biron and A. Malhotra. *XML Schema Part 2: Datatypes Second Edition.* W3C Recommendation, October 2004. `www.w3.org/TR/xmlschema-2/`.
7. T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau. *Extensible Markup Language (XML) 1.0 (Third Edition).* W3C Recommendation, February 2004. `http://www.w3.org/TR/REC-xml/`.
8. B. Choi. What are real DTDs like? In *WebDB '02, Proceedings of the 5th International Workshop on the Web and Databases*, pages 43–48, Madison, Wisconsin, USA, 2002. ACM Press.
9. D. C. Fallside and P. Walmsley. *XML Schema Part 0: Primer Second Edition.* W3C Recommendation, October 2004. `www.w3.org/TR/xmlschema-0/`.
10. R. Fenk. The BUB-Tree. In *VLDB '02, Proceedings of 28th International Conference on Very Large Data Bases*, Hong Kong, China, 2002. Morgan Kaufman Publishers.
11. A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *SIGMOD'84, Proceedings of Annual Meeting*, pages 47–57, Boston, Massachusetts, 1984. ACM Press.
12. M. Klettke, L. Schneider, and A. Heuer. Metrics for XML Document Collections. In *XMLDM Workshop*, pages 162–176, Prague, Czech Republic, 2002.
13. J. Kosek, M. Kratky, and V. Snasel. Struktura realnych XML dokumentu a metody indexovani. In *ITAT 2003 Workshop on Information Technologies Applications and Theory*, High Tatras, Slovakia, 2003. (in Czech).
14. A. McDowell, C. Schmidt, and K. Yue. Analysis and Metrics of XML Schema. In *SERP '04, Proceedings of the International Conference on Software Engineering Research and Practice*, pages 538–544. CSREA Press, 2004.
15. L. Mignet, D. Barbosa, and P. Veltri. The XML Web: a First Study. In *WWW '03, Proceedings of the 12th international conference on World Wide Web, Volume 2*, pages 500–510, New York, NY, USA, 2003. ACM Press.
16. I. Mlynkova, K. Toman, and J. Pokorny. *Statistical Analysis of Real XML Data Collections.* Technical report 2006/5, Charles University, June 2006.
17. M. Murata, D. Lee, and M. Mani. Taxonomy of XML Schema Languages using Formal Language Theory. In *Extreme Markup Languages*, Montreal, Canada, 2001.
18. A. Sahuguet. Everything You Ever Wanted to Know About DTDs, But Were Afraid to Ask (Extended Abstract). In *Selected papers from the 3rd International Workshop WebDB 2000 on The World Wide Web and Databases*, pages 171–183, London, UK, 2001. Springer-Verlag.
19. J. Shanmugasundaram, K. Tufte, C. Zhang, G. He, D. J. DeWitt, and J. F. Naughton. Relational Databases for Querying XML Documents: Limitations and Opportunities. In *VLDB'99, Proceedings of 25th International Conference on Very Large Data Bases*, pages 302–314, Edinburgh, Scotland, UK, 1999. Morgan Kaufmann.

20. H. S. Thompson, D. Beech, M. Maloney, and N. Mendelsohn. *XML Schema Part 1: Structures Second Edition.* W3C Recommendation, October 2004. `www.w3.org/TR/xmlschema-1/`.