Statistical Analysis of Real XML Data Collections
(Technical Report)

Irena Mlynkova, Kamil Toman, and Jaroslav Pokorny

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

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

Abstract. Recently XML has achieved the leading role among languages for data representation and thus we can witness a massive boom of corresponding techniques for managing XML data. Most of the processing techniques however suffer from various bottlenecks worsening their time and/or space efficiency. We assume that the main reason is they consider XML collections too globally, involving all their possible features, although real data are often much simpler. Even though some techniques do restrict the input data, the restrictions are often unnatural.
In this paper we analyze existing XML data, their structure and real complexity in particular. We have gathered more than 20GB of real XML collections and implemented a robust automatic analyzer. The analysis considers existing papers on similar topics, trying to confirm or confute their observations as well as to bring new findings. It focuses on frequent but often ignored XML items (such as mixed content or recursion) and relationship between schemes and their instances.

1 Introduction

It is a well known fact (not only in the scientific world) that XML [39] has undoubtedly achieved the leading role among existing standards for data representation and its popularity is further spreading. Arm in arm with this tendency 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. But for overwhelming majority of these techniques there can be found certain bottlenecks that cause quite significant problems resulting in worsening of their most interesting features, generally speaking effectiveness. But where do these bottlenecks come from?

Under a closer investigation we can distinguish two situations. On one hand there is a group of general techniques that take into account all possible features of input XML data. This idea is obviously correct but the problem is that the standards were proposed as generally as possible so future users can choose what suits them most. But 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. It can even be harmful.

On the other hand there are techniques that somehow do restrict features of the input XML data. Then it is natural to expect bottlenecks to occur only in situations when given data do not correspond to these restrictions. The problem is that such restrictions are often “unnatural”. They do not result from features of real XML data but from other more down-to-earth reasons, e.g. due to limitations of the basic proposal of a particular technique, complexity of such solution, irregularities etc.

We can naturally pose two apparent questions:

1. Is it necessary to take into account a feature that will be used minimally or will not be used at all?
2. If so, what are these features?

The answer for the first question obviously depends on the particular situation, the answer for the second one is the main topic of this paper. And we assume that it can be answered through statistical analysis of real XML data collections.

Our analysis focuses on usual features, structure, and complexity of existing XML data collected all over the Internet. Primarily we result from existing papers on similar topics, whose most interesting observations we discuss and either confirm or confute. The main part of our research consist of detailed analysis of XML features that seem to be important but are often omitted for the complexity of their processing (e.g. mixed content or recursion), analysis of several new constructs we have defined (e.g. DNA patterns or relational patterns), and analysis of relationship between XML documents and XML schemes.

The paper is structured as follows: The first section introduces the considered problems. The following, second section contains an overview of formalism used throughout the paper. Section 3 discusses existing related works and their findings and observations. Section 4 describes and classifies the analyzed data in general and Section 5 describes the key analyses and the most interesting results. The last, sixth section provides conclusions and possible future work.[1]

2 Formal Definitions

For structural analysis of XML data it is natural to view XML documents as ordered trees and DTDs [39] or XSDs (i.e. XML Schema [41, 53, 37] definitions) as sets of regular expressions over element names. Attributes are often omitted

[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.

for simplicity. We use notation and definitions for XML documents and DTDs from [40] and [36]. (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* $T = (\Sigma, N, E, r)$ *where* Σ *is a finite alphabet,* N *is a set of nodes of the tree,* E *is a set of edges of the tree, and* $r \in N$ *denotes* a root element *of the tree. Each node* $n \in N$ *is associated with* a type of the node *which can be one the following:* element, attribute, text, processing instruction, *or* comment. *Nodes with element or attribute type are also associated with a node label* $l \in \Sigma$ *called* an element name *or* an attribute name *respectively.*

The tree T *is called* Σ*-tree.*

Note 1. For simplicity, we use the terms *element* and *attribute* for nodes of the respective type.

Definition 2. A DTD *is a collection of element declarations of the form* $e \to \alpha$ *where* $e \in \Sigma$ *is an element name and* α *is its content model, i.e. regular 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* α_1, α_2, ..., α_n*), and "*", "+", and "?" stand for zero or more, one or more, and optional occurrence(s) (of content model* β*) respectively.*

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* $n \in N$ *and its label* e*, the sequence* e_1, e_2, ..., e_k *of labels of its child nodes matches the regular expression* α*, where* $e \to \alpha$.

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

Definition 4. A 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. A distance of elements e_1 *and* e_2 *is the number of edges in* Σ*-tree separating their corresponding nodes.*

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

A 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* $e'' \Rightarrow e'$.

An element e' *is* a descendant of element e*, if* l' *is element name of* e', l *is element name of* e *and* $l \Rightarrow l'$.

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* $s \Rightarrow e$*, where* s *is the start symbol. Otherwise it is called* unreachable.

Definition 7. *A content model* α *is* mixed*, if*
$\alpha = (\alpha_1|...|\alpha_n|pcdata)* \mid (\alpha_1|...|\alpha_n|pcdata)+$ *where* $n \geq 1$ *and for* $\forall i$ *such that* $1 \leq i \leq n$ *content model* $\alpha_i \neq \epsilon \bigwedge \alpha_i \neq pcdata$.

An element e *is called* mixed-content element *if its content model* α *is mixed.*

Definition 8. *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 element name as* e.

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

Definition 9. A 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* $1 \leq i < k$*. The number of elements in the list is called* a length of a simple path.

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

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

Definition 10. A fan-in *of an element* e *is the cardinality of the set* $\{f \mid f \rightarrow \alpha'$ *and the element name* e' *of element* e *occurs in* $\alpha'\}$.

An element with large fan-in value is called a hub.

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

An attribute fan-out *of an element* e *is the number of its attributes.*

Definition 12. A base symbol *is a regular expression of the form* a, $a?$*, or* $a*$*, where* $a \in \Sigma$.

A factor *is a regular expression of the form* e, $e?$*, or* $e*$*, where* e *is a disjunction of base symbols.*

A simple regular expression *is* ε, $\emptyset$*, or a sequence of factors.*

Several existing papers focus on more detailed analysis and classification of complexity and types of recursion. For this purpose they define new constructs, e.g. types of recursion, types of its representation, regularity of recursion, or recursive fan-out.

Definition 13. *An element e is* linearly recursive *if it is recursive and for every α such that $e \Rightarrow \alpha$ e is the only recursive element that occurs in α and neither of its occurrences is enclosed by "*" or "+".*

An element is non-linear recursive *if it is recursive but it is not linear recursive.*

Definition 14. A recursive XML tree *is an XML tree that is rooted at a recursive element and whose leaves are descendants of the root node which are also recursive.*

An all-descendants (AD) interpretation *of a recursive XML tree involves all the recursive descendants of the root node.*

A closest-descendants (CD) interpretation *of a recursive XML tree involves only the closest descendants of the root.*

Definition 15. A regularity of recursion *of element e is the average distance between elements in all ed pairs of the document in CD interpretation.*

Definition 16. An AD (CD) recursive fan-out *of element e is the number of its recursive descendants in AD (CD) interpretation.*

3 Related Work

Up to now only a few papers have focused on analysis of real XML data. They analyze either the structure of DTDs, the structure of XSDs, or the structure of XML documents (regardless their schema). The sample sets of XML data usually differ.

3.1 DTD Analysis

Probably the first attempt to analyze the structure of XML data (in this particular case the structure of DTDs) can be found in [51]. 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, e.g. the number of elements, attributes, or entity references,
- the structure of DTDs, e.g. the number of root elements or depth of content models, and
- some specific aspects, e.g. the use of mixed contents, `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 observations important for our research, 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 related paper [40] also focuses on DTDs, but its analysis is 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 DTDs of all categories).

- 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.

The last mentioned paper which focuses on DTD analysis is trying to adapt software metrics to DTDs [43]. It defines five metrics also based on their graph representation – i.e. size, complexity, depth, fan-in, and fan-out, whereas all of them have already been 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 constructs of XML Schema not allowed in DTD are used in practice? Paper [36] 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 used regular expressions.

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

- extension[2] (27%) and restriction (73%) of simple types,
- extension (37%) and restriction (7%) of complex types,
- `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%) constructs,
- namespaces (22%), and
- redefinition of types and groups (0%).

It is evident that 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. The concluding finding is, that 85% of XSDs define so called *local tree languages* [49], i.e. languages that can be defined by DTDs as well, and thus that the expressiveness beyond local tree grammars is needed rarely.

Second part of the paper focuses on real complexity of both DTDs and XSDs. Unfortunately only 30 out of the 93 XSDs could be used for these analyses, because the others did not adhere to W3C specifications. Both the DTDs and

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

XSDs were first preprocessed (i.e. the entities were resolved, conditional sections were included/excluded, multiplicity constraints were replaced using "?" operator etc.) and transformed into so-called *canonical form*, which abstracts away the actual element names and replaces them with canonical names c_1, c_2, ... to preserve the structure related information. For example

```
<!ELEMENT lib ((book|journal)*)>
```

is represented by canonical form $(c_1|c_2)^*$. The resulting number of distinct canonical forms is 750 among DTDs and 138 among XSDs. The subsequent analysis focuses on their real complexity and its most interesting findings are listed below.

- The vast majority of regular expressions (i.e. 92% for DTDs and 97% of XSDs) are simple regular expressions.
- There are no significant differences between expressions used in DTDs and XSDs. The most striking ones are that XSDs have:
  - 14% more `simpleType` content models and
  - 18% less expressions of the form a or $(a_1|a_2|...|a_n)^*$, where a, a_1, a_2, ..., $a_n \in \Sigma$.

The paper also focuses on chains of stars and ambiguity. The results are similar to the previous cases.

3.3 XML Schema Analysis

Paper [36] 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 constructs. Paper [47] 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 17. 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 [48] (and its extension [34]) 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 Internet, whereas the statistics are divided into two groups – statistics about the XML web and statistics about the XML documents.

The XML 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. Nevertheless, more important for our research are statistics about the XML documents which 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

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

– the number of distinct strings, and
– recursion.

The most interesting findings of the research are as follows:

– 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 regularity of most XML documents is 1. More than 97% of recursive documents are highly regular.
– The most common average fan-outs are 1 (60%) and 2 (37%), the average recursive fan-out is 2.2.

The last mentioned paper [44] 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.

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

3.5 Conclusions

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. 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, UB-trees or BUB-trees [45, 46]. 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 structure 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 a schema is of course a big problem for methods which are based on its existence, e.g. schema-driven database mapping methods [52].

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 non-determinism and ambiguity into this set of errors or if it expresses a demand for extension of XML recommendations.

3.6 Aims

In this paper we take up work initiated in the previously mentioned existing articles. We focus on analysis of XML data aspects, which are (in our opinion) important for effective data processing and querying. In contrast to existing papers we omit e.g. the geographical or Internet source of XML data collections or secondary XML items such as namespace references, document out-degree, usage of `include` and `import` elements in XSDs etc. Generally speaking, our analysis focuses on aspects which influence the structural complexity of XML data or carry additional important information while it ignores items that are rather relevant to semantic web [31] or items that can be even qualified as "syntactic sugar".

First of all, we repeat the most interesting analyses on our sample data and compare the results with the existing ones. The reason is that our analyzed data differ from the previously used ones – they are larger and often even much more natural. The main part of our research focuses on new types of analyses, with emphasis on previously determined frequent aspects such as mixed content and recursion. Last but not least we compare the the results for XML documents with corresponding results for their schemes where it is possible.

4 Sample XML Data Collections

We have collected a huge amount of XML documents and their DTDs/XSDs. In contrast to existing papers the XML collections were not only collected automatically using a crawler but also manually from sources offering their data natively in XML format (government sites, open document repositories, web site XML exports etc), Internet catalogues and semantic web resources. The respective schemes for data sets were often searched out later in separate because they had been missing in the original sources. Then the collections were categorized and the duplicate documents identified by a simple hashing algorithm (disregarding white spaces) and subsequently removed. Also computer-generated or random-content XML files were eliminated.

The reason for using more reliable and/or categorized sources is that automatic crawling of XML documents generates a set of documents that are "unnatural" and often contain only trivial data which cause misleading results. For example paper [48] mentions that the set of sample data (which were crawled automatically) contains almost 2000 of documents with depth 0, i.e. documents containing a single element with empty or text content.

Our purpose was to collect a representative set of currently used XML collections from various spheres of human activities. We have included data which are typically used for testing XML processing methods (e.g. Shakespeare's plays [1], XMark [2], Inex [3], The Bible in XML [4, 5]), representatives of standard XML schemes (e.g. XHTML [32], SVG [33], RDF [35], DocBook [29]), sample database exports (e.g. FreeDB [6], IMDb [7]), well-known types of documents (e.g. OpenOffice documents [30]), randomly crawled XML data (Medical Subject Headings [8], novels in XML [9], RNAdb [10]) etc.

4.1 Preprocessing

As it was previously mentioned, authors of most existing papers complain of a large number of errors in the collected sample data. Unfortunately we can only confirm the claim. The overwhelming majority of the collected data contained various types of serious errors. XML documents were not even well-formed and more than a half of the well-formed ones contained invalid data. Similar problems were found in case of DTDs and XSDs.

Contrary to previous papers we have not decided to discard the data. Most of the errors (e.g. bad encoding, missing end tags, missing elements, unescaped special characters, wrong usage of namespaces etc.) were detected using Xerces Java Parser [50] or our own auxiliary analyzers and semi-automatically corrected. Most of the corrections had to be done manually though.

4.2 Accessibility of the Data

Unfortunately, we cannot release the resulting set of corrected XML data collections for download since some of them are not free or underlie to Copyright Act. Nevertheless, the Appendix contains an overview of the data and their sources. Understandably, we also cannot be responsible for their current validity.

4.3 General Statistics and Classifications

First of all, we have computed statistics that describe the sample XML data in general. The overview of these statistics is listed in Table 1. A detailed list of XML collections involving their source, size in Bytes, number of documents per collection and DTD/XSD existence flag can be found in the Appendix.

Statistics		**Results**
Number	Number of XML documents	16,534
	Number of XML collections	133
Size	Total size of documents (MB)	20,756
	Minimum size of a document (B)	61
	Maximum size of a document (MB)	1,971
	Average size of a document (MB)	1.3
	Median size of a document (kB)	10
	Sample variation (MB)	433.84
Schema	Documents with DTD (%)	74.6
	Documents with XSD (%)	38.2
	Documents without DTD/XSD (%)	7.4

Table 1. General statistics for XML data

As it is obvious, the sizes of XML documents vary strongly (from 61B to 1,971MB), nevertheless both the average size (1.3MB) and median size (10kB) seems to be "natural". Another (not surprising) finding is that a considerable percentage of documents (7.4%) still does not have any schema (although the ratio is better than in existing works) and if so, the XML Schema language is for this purpose used even less (only for 38.2% of documents).[7] The positive results may be influenced by the fact, that the gathered data were collected semi-automatically, not randomly.

To avoid averaging the features of the whole data set where the documents have nothing in common but having an XML format we have categorized the

[7] Some documents have both DTD and XSD, thus the sum is not 100%.

data by the original sources and further grouped according to similar structure, contents or approach used to describe the data. This way we have obtained a finer look into various types of XML data not neglecting the interesting differences among the selected categories whereas we have avoided the extensive amount of similar results.

We distinguish six logical categories of XML documents for which the following statistics are computed. The categories are as follows:

– *data-centric documents* (`dat`), i.e. documents designed for database processing (e.g. database exports, lists of employees, lists of IMDb movies and actors etc.),
– *document-centric documents* (`doc`), i.e. documents which were designed for human reading (e.g. Shakespeare's plays, XHTML [32] documents, novels in XML, DocBook [29] documents etc.),
– *documents for data exchange* (`ex`) (e.g. medical information on patients and illnesses etc.),
– *reports* (`rep`), i.e. overviews or summaries of data (usually of database type),
– *research documents* (`res`), i.e. documents which contain special (scientific or technical) structures (e.g. protein sequences, DNA/RNA structures etc.), and
– *semantic web documents* (`sem`), i.e. RDF [35] documents.

As it is obvious, the categories are not disjunctive as well as in case of the well-known data-centric a document-centric ones, nevertheless no XML document has been inserted into more than one category.

To get a more detailed notion of the sample data we have computed the general statistics also for each category. The overview of the results is listed in Table 2. Assignment of XML collections into particular categories can be found in the Appendix.

Statistics		**dat**	**doc**	**ex**	**rep**	**res**	**sem**
Number	Number of XML documents	3,412	6,691	218	2,983	2,451	779
	Number of XML collections	38	22	25	2	16	30
Size	Total size of documents (MB)	2,237	1,187	371	11,371	1,697	3,892
	Minimum size of document (B)	447	61	2,433	1,925	2,016	356
	Maximum size of document (MB)	1,242	16	134	96	684	1,971
	Average size of document (kB)	672	182	1,744	3,903	709	5,116
	Median size of document (kB)	3.8	13.4	5.7	1,574	6.9	45.0
	Sample variation (MB)	510.66	0.54	154.40	61.84	352.32	5534.50
Schema	Documents with DTD (%)	99.7	93.7	100	0	99.8	0
	Documents with XSD (%)	0	57.8	0	100	99.6	0
	Documents without DTD/XSD (%)	0.3	6.3	0	0	0.2	100

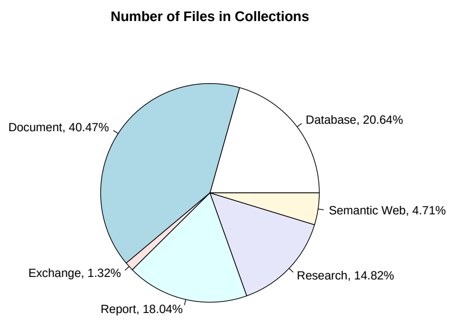
Table 2. General statistics per category

Considering the sizes of individual XML files, the most homogenous category is the `doc` one. Surprisingly it is also the one with the least document size

(61B). Apparently this is because most of these XML documents are written by hand or with the aid of an XML editor. The other extreme is the `sem` category which, despite the similar structure of documents, contains some very large files (including the largest one with 1,971MB) as well as collections split into very small documents, depending heavily on the used tool and conventions.

Apparently the most striking information is the percentage of using DTDs or XSDs which is either almost 0% or almost 100% depending on the category. The reason probably comes from the higher reliability of the used sources and the chosen categorization. On the other hand, though a considerable portion of all documents used some kind of a standard schema (e.g. XHTML, DocBook etc.), they were not 100% valid against it. Thus the schema could be used as a guide for human processing but it would not be usable for computer validation.

The number and size of files per each category is for better clarity depicted using pie charts in Figure 1. In addition histograms in Figure 2 depict the frequency and distribution of document sizes per each category.



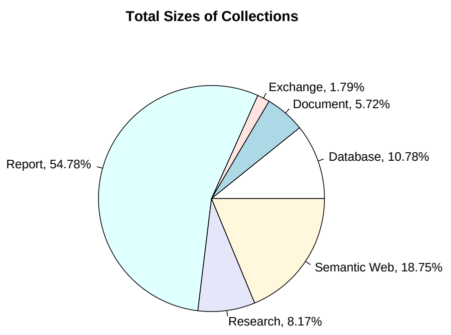


Fig. 1. Number and size of files per category

We can see that although most documents (40.47%) belong to `doc` category their total size portion is quite small (5.72%) whereas the greatest portion belongs to `rep` documents (54.78%). Another finding is that the document sizes tend to be distributed "naturally".

The document sizes usually show lognormal distribution in each collection resulting in composite distribution with multiple peaks. This can be nicely seen on `rep` documents with two main peaks one for each collection within the given category.

5 Analyses and Results

There are two main parts of our observations. Firstly, we carry out analyzes similar to previous studies and compare the results with the original ones. Secondly we focus on new XML features with emphasis on frequently disregarded ones, their complexity and corresponding classification.

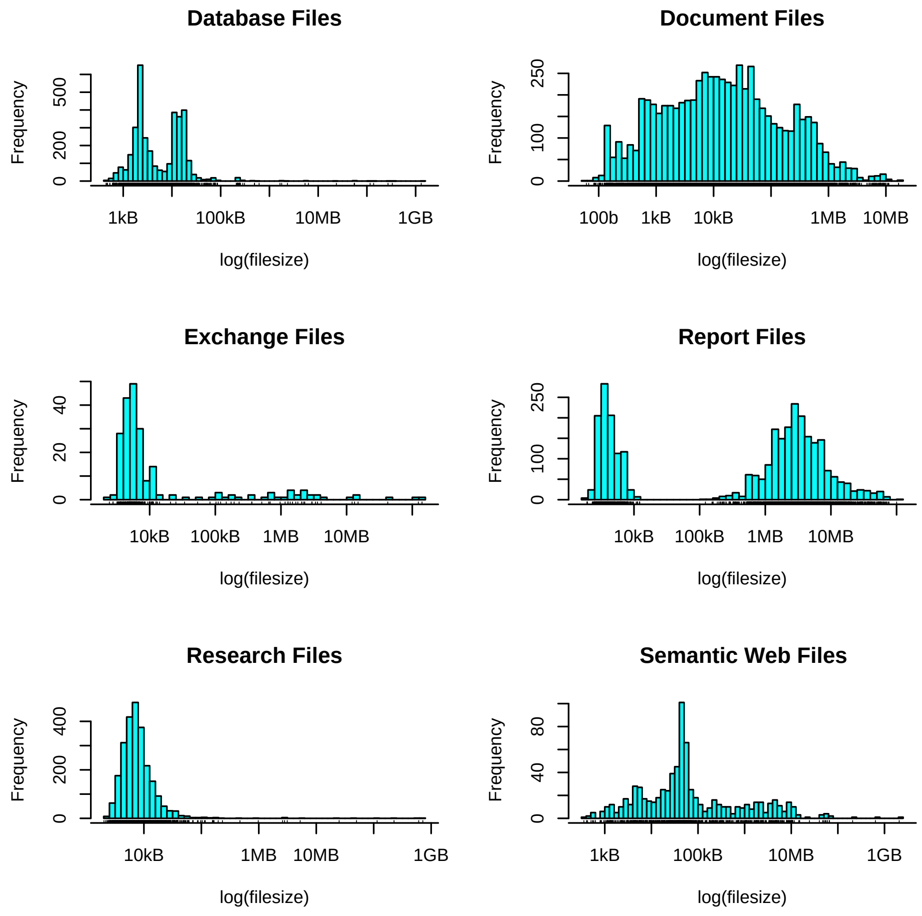


Fig. 2. Distribution of document sizes per category

5.1 New Constructs

On the basis of our experience and manual preprocessing of the analyzed data we have defined several new constructs which describe XML documents and schemes with higher degree of detail. The constructs involve two types of mixed contents, four types of recursion, two special types of elements called relational and DNA patterns, and two types of element fan-out. Particularly for scheme analyses we further distinguish another two types of element fan-out.

We have focused especially on simple patterns within the general XML constructs which can usually be represented by "ordinary" relational tables or, in case of recursion and mixed contents, which can be processed and stored simply without the usual generalization.

Definition 18. *An element is* trivial *if it has an arbitrary amount of attributes and its content model $\alpha = \epsilon \mid pcdata$.*

A mixed-content element is simple *if each of its subelements is trivial. A mixed-content element that is not simple is called* complex.

Definition 19. *An element e is* trivially recursive *if it is recursive and for every α such that $e \Rightarrow \alpha$ e is the only element that occurs in α and neither of its occurrences is enclosed by "*" or "+".*

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

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

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

Note 2. For better lucidity we have repeated the definition of linear recursion.

Note 3. Note that trivial recursion is a proper subset of linear recursion and linear recursion is a proper subset of pure recursion.

Definition 20. *A nonrecursive element e is called* a DNA pattern *if it is not mixed and its content model α consists of a nonzero amount of trivial elements and one nontrivial and nonrecursive element whose occurrence is not enclosed by "*" or "+". The nontrivial subelement is called* a degenerated branch.

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

Definition 21. *A nonrecursive element e is called* a relational pattern *if it has an arbitrary amount of attributes, it is not mixed, and its content model $\alpha = (e_1, e_2, ..., e_n)* \mid (e_1, e_2, ..., e_n)+ \mid (e_1|e_2|...|e_n)* \mid (e_1|e_2|...|e_n)+$, where e_1, e_2, ..., e_n are trivial elements.*

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

Definition 22. An element fan-out *of element e is the number of distinct elements in its content model α.*

A simple element fan-out *of element e is the number of distinct trivial elements in its content model α.*

Definition 23. A minimum element fan-out *of element e is the minimum number of elements allowed by its content model α.*

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

5.2 Statistics and Results

In this section we describe and discuss all the computed statistics, their results, reasons, and consequences. For better lucidity they are divided into 10 logical categories – global, depth, level, fan-out, fan-in, recursive, mixed-content, DNA, relational, and schema statistics.

We have computed number of statistical parameters for particular features, i.e. number of occurrences or percentage, size or length, minimum and maximum value, average and mean value, variance, quantiles, etc. Due to the enormous amount of information the paper involves only the most interesting ones. If possible we also compare the analyses of XML documents with corresponding analyses of their XML schemes.[8] There are of course no schema results for `sem` documents since this category has no schema at all.

Global Statistics The first set of statistics, called *global*, considers overall properties of XML data such as number of elements of various types (i.e. empty, text, mixed, recursive etc.), number of attributes, paths, depths and portion of text in documents. In case of DTDs/XSDs the depths are counted for each global element used in the sample XML documents as a root element, for recursive elements we take into account their lowest level(s) and an infinite level for expressing the recursion.

Probably the most relevant results are listed in Table 3 that contains values for 95% of all documents. In other words each column contains the overall results of all documents disregarding exactly 5% of those ones with the most extreme values. This way we can get the notion of a typical document for each category regardless misleading extremes. For the sake of completeness, the extreme values are listed in Table 4 (minimums) and Table 5 (maximums). In this case the tables also contain meaningful values for XML schemes.

It is apparent that most of the documents are constructed quite simply using only a very reasonable number of distinct element and attribute names (usually less than 150) which influences a similar number of distinct paths within each category. And even though the maximum number of elements in an XML tree is often huge, the number of distinct paths still remains several orders lower. This naturally corresponds with the average and maximum depths of XML documents which are very low (all under 13 disregarding the 5% of extreme values). Nevertheless, note that this is not the case for XML schemes which naturally allow much richer structures.

In all documents the maximum depth exceeded 20 only for some specific, often heavily recursive instances. It is arguable whether it is their inherent feature or if it is a result of a lack of a good structure design. The maximum depth of corresponding schemes is higher but it also tops around 80 (disregarding possible recursion, i.e. infinite depth).

[8] In such case there are **Doc.** and **Sch.** abbreviations on the left-hand side of a table which identify the results.

Besides that, the recursion is also remarkably trivial. Mostly, the number of distinct element names with recursive occurrences is up to 3 and even the most complex documents do not have more than 12 possible recursive elements present at the same time. We will further deal with this finding in recursive statistics.

Statistics		**dat**	**doc**	**ex**	**rep**	**res**	**sem**
Max. number of elements		402	4,085	37,502	309,379	427	112,942
Max. number of attributes		9	1,675	5,182	37,815	129	37,996
Max. number of empty elements		3	361	123	16,348	6	23,635
Max. number of mixed elements		0	302	21	0	1	0
Max. number of distinct el. names		81	48	58	388	44	144
Max. number of rec. elements		0	3	2	0	0	0
Max. number of distinct paths		79	96	67	312	30	143
Depth of document	Avg.	5	7	5	5	5	5
	Max.	5	13	9	6	7	6

Table 3. Global statistics for 95% XML documents

	Statistics	**dat**	**doc**	**ex**	**rep**	**res**	**sem**
Doc.	Number of elements	8	1	4	26	44	1
	Number of attributes	0	0	0	23	0	0
	Number of distinct el. names	5	1	4	23	10	1
	Number of distinct paths	3	1	1	13	8	1
	Depth of document	3	1	3	4	3	1
Sch.	Number of distinct el. names	7	5	5	109	28	-
	Number of distinct paths	5	1	3	97	26	-
	Depth of schema	2	4	2	3	5	-

Table 4. Minimum values of global statistics

Table 6 further contains the exploitation rate of global properties, i.e. percentage of documents/schemes with at least one occurrence of a particular property. Note that the results differ substantially for categories where the XML documents are expected to be read or written by humans and for data-oriented ones. The database oriented XML files are designed to be more regular to ensure further simple computer processing while "human-oriented" data are much more terse and also a richer syntax is used to emphasize the semantics and to improve readability. For example, it is only a matter of taste whether an attribute or simple element is used to represent the same information in the document. Similar statements hold for mixed-content or empty elements.

However, as the table shows, the most common features are spread throughout all categories and thus should not be ignored by the XML processors. The

	Statistics	dat	doc	ex	rep	res	sem
Doc.	Number of elements	23,132,565	267,632	2,911,059	1,957,637	21,305,818	25,548,388
	Number of attributes	33,660,779	102,945	857,691	208,265	2,189,859	10,228,483
	Number of distinct el. names	81	134	146	461	210	1,410
	Number of distinct paths	434	2,086	144	373	426	2,534
	Depth of document	12	459	14	6	19	11
Sch.	Number of distinct el. names	76	377	523	3,213	250	-
	Number of distinct paths	115	11,994	1,665	3,137	568	-
	Depth of schema	12	81	79	5	15	-

Table 5. Maximum values of global statistics

only exception are mixed-content elements which are used sparsely especially outside the `doc` category.

	Node type	dat	doc	ex	rep	res	sem
Doc.	Attribute	31.7	96.2	92.2	100.0	99.9	99.9
	Empty element	26.8	69.2	89.9	100.0	86.7	92.7
	Mixed element	0.2	76.5	8.7	0.0	10.1	2.4
	Recursive element	0.1	43.3	63.8	0.0	0.7	3.3
Sch.	Attribute	50.0	94.1	52.6	100.0	85.7	-
	Empty element	37.5	94.1	47.4	25.0	71.4	-
	Mixed element	37.5	100.0	50.0	0.0	57.1	-
	Recursive element	12.5	88.2	18.4	0.0	28.6	-

Table 6. Exploitation rate of global properties (%)

Finally Table 7 contains the portion of text in all XML documents of a particular category. In this case we can observe that except for the `doc` category (with 81.6% of text) the tagging often dominates the size of documents. This indicates that the structure of the data is not only a matter of lucidity or orderliness but it carries important information.

Statistics	dat	doc	ex	rep	res	sem
Percentage of text	43.8	81.6	36.3	6.2	33.1	54.9

Table 7. Portion of text in XML documents (%)

Last fact to be observed is that XML features used in schemes and document instances usually match, i.e. the schemes are quite well designed in this matter.

Depth Statistics From previous set of statistics we already know the maximum and average depth of a typical document per each category as well as the general extreme values. Nevertheless the depths, especially in case of XML documents, can be further analyzed using so-called *depth* statistics.

Figure 3 depicts the distribution of depths per each category. As we can see the typical depth is always less than 10 which confirms the results of previous works and encourages proposals of techniques whose effectiveness is closely related to maximum depth of XML data.

Unfortunately we cannot get similar results for corresponding XML schemes, because they are too much influenced by recursion. We can analyze the percentage of such infinite XML schemes and/or XML elements but due to properties of XML Schema language they correspond to recursive XML schemes and/or elements which were already discussed.

Level Statistics As it is expectable *level* statistics focus on distribution of elements, attributes, text nodes, and mixed contents per each level of XML documents. Figure 4 depicts the distributions of individual constructs for the whole sample set of documents. Figures for each of the individual categories were omitted since they do not show any substantial differences from the overall distribution. (Unfortunately XML schemes were usually specified too generally to generate any meaningful statistic data.)

The graphs show that the highest amounts of analyzed nodes are always at first levels (except for level 0) and then the number of occurrences rapidly decreases. The steep exponential decrease ends around level 20 and then the drop is much slower and shows more fluctuations. This correlates closely with fan-out statistics in Figure 5 – see below.

Note that unlike attributes or mixed contents the text nodes occurrences copy the curve of element frequencies almost perfectly. This denotes that the text content is spread evenly trough all levels of XML documents.

Fan-out Statistics Fan-out statistics describe the overall distribution of XML data. Figures 5 and 6 depict the results of element fan-out for XML documents and XML schemes. Figures 7 and 8 depict the results of simple element fan-out for XML documents and XML schemes. In all four cases we have used 3D graphs where the axes correspond to level, value of fan-out, and number of occurrences of a particular fan-out value at a particular level. Each level is for better lucidity identified with a different color.

In case of XML documents we can observe that the characteristics of the graph are similar at each level but with the growing depth it gets thinner. Similarly to level statistics the highest values are at first levels and soon they radically decrease. A surprising finding is that graphs for simple element fan-out are similar, just thinner. This denotes, that the distribution of trivial elements is almost same as the general distribution of elements. Thus we can say that their occurrence is not only frequent (as we already know from previous statistics) but also regular for each level of document.

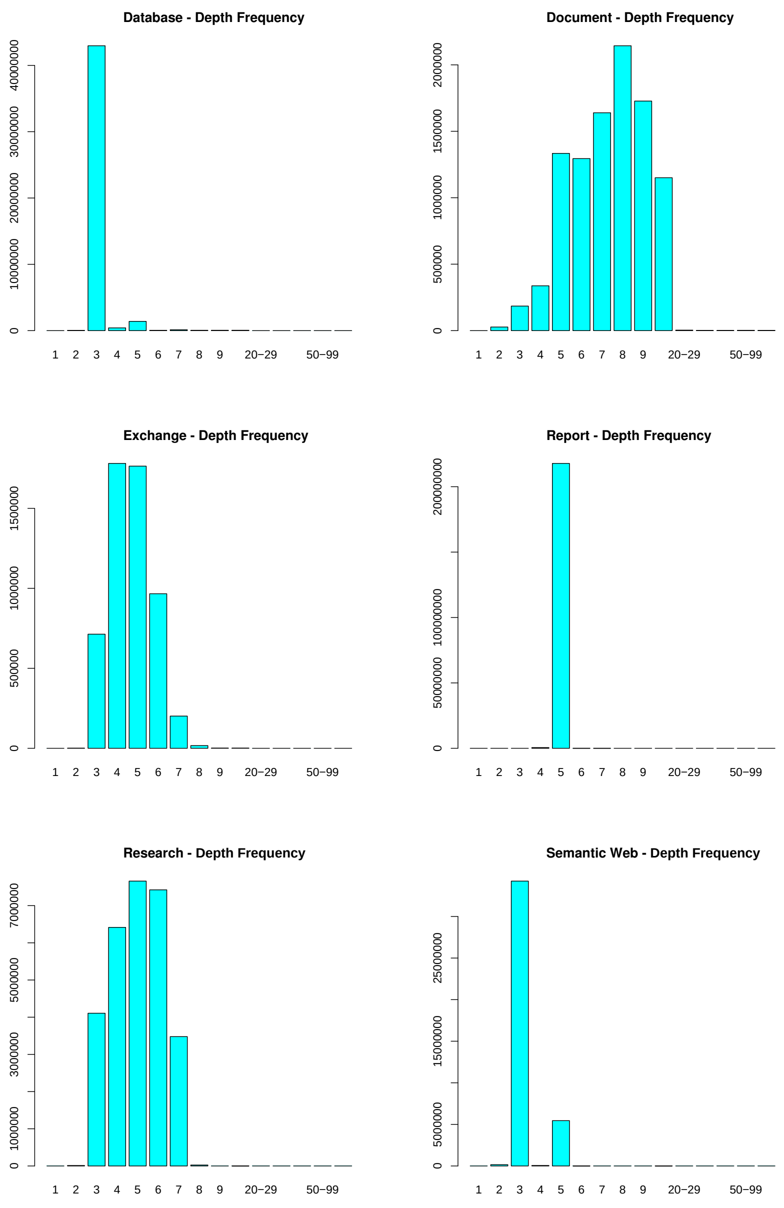


Fig. 3. Distribution of depths of XML documents per category

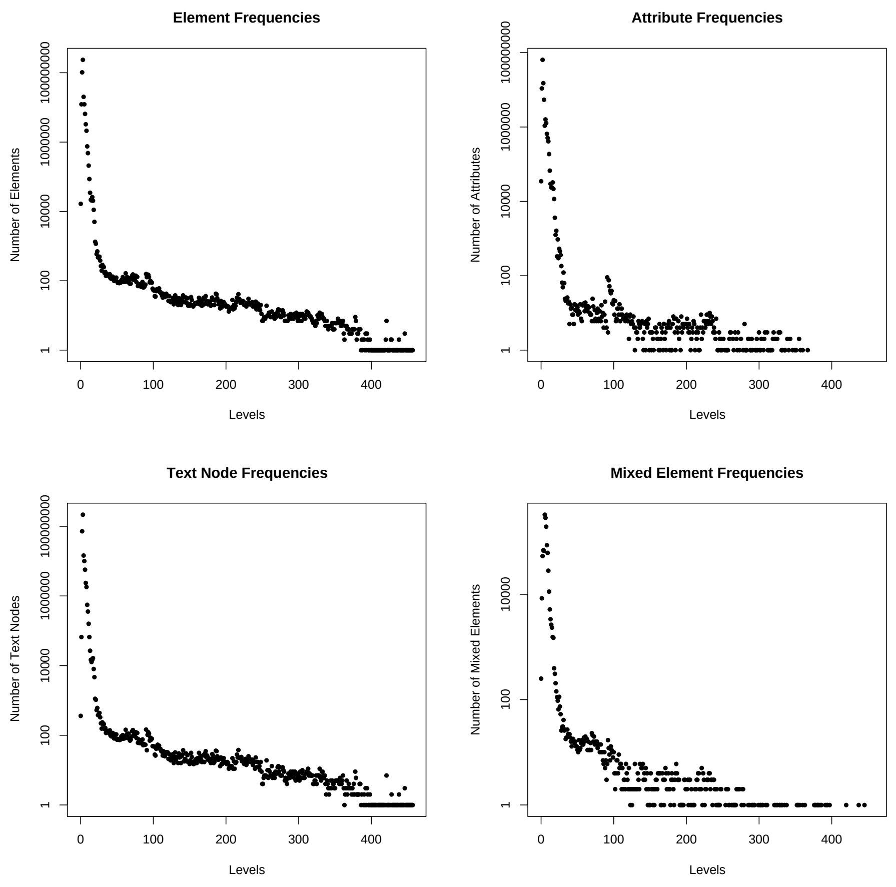


Fig. 4. Distribution of elements, attributes, text nodes, and mixed contents in XML documents per level

The graphs for XML schemes have less interesting characteristics, again caused by recursion and possible infinite depth.

Fan-in Statistics In case of XML schemes we can also analyze "inverse" fan-out characteristic called fan-in. The results are depicted in Table 8.

Statistics		**dat**	**doc**	**ex**	**rep**	**res**	**sem**
Fan-in	Avg.	1.6	179.5	5.0	1.0	1.5	-
	Max.	13	3,176	177	12	14	-

Table 8. Fan-in statistics per category

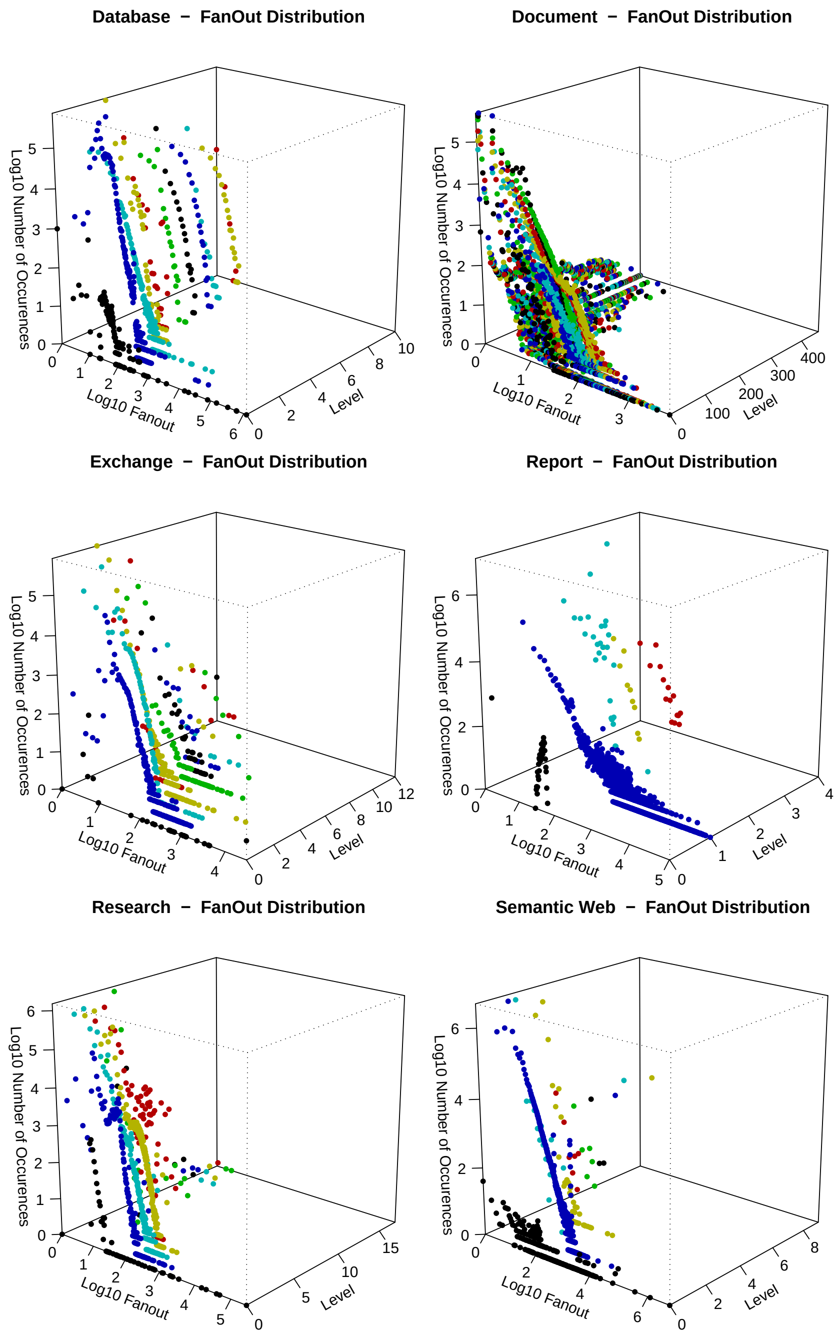


Fig. 5. Element fan-out of XML documents per category

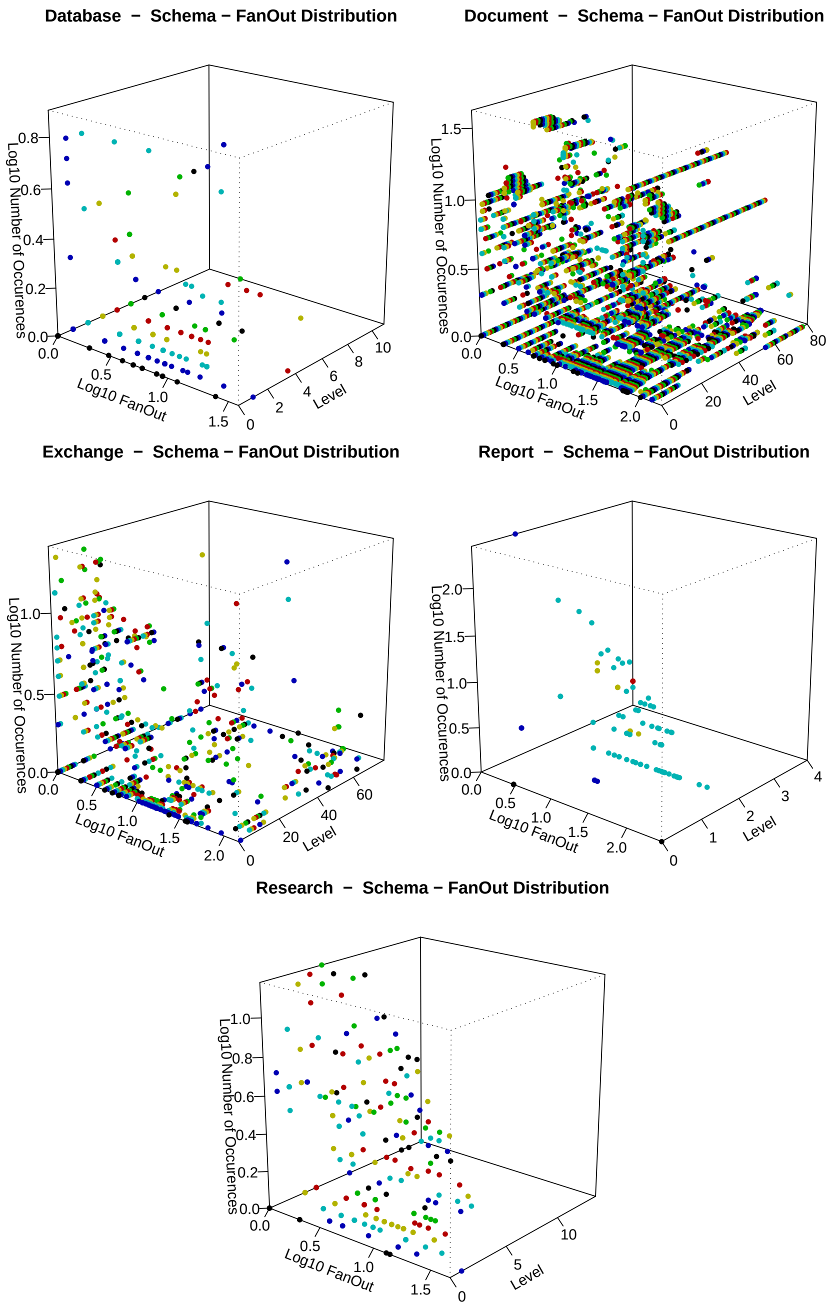


Fig. 6. Element fan-out of XML schemes per category

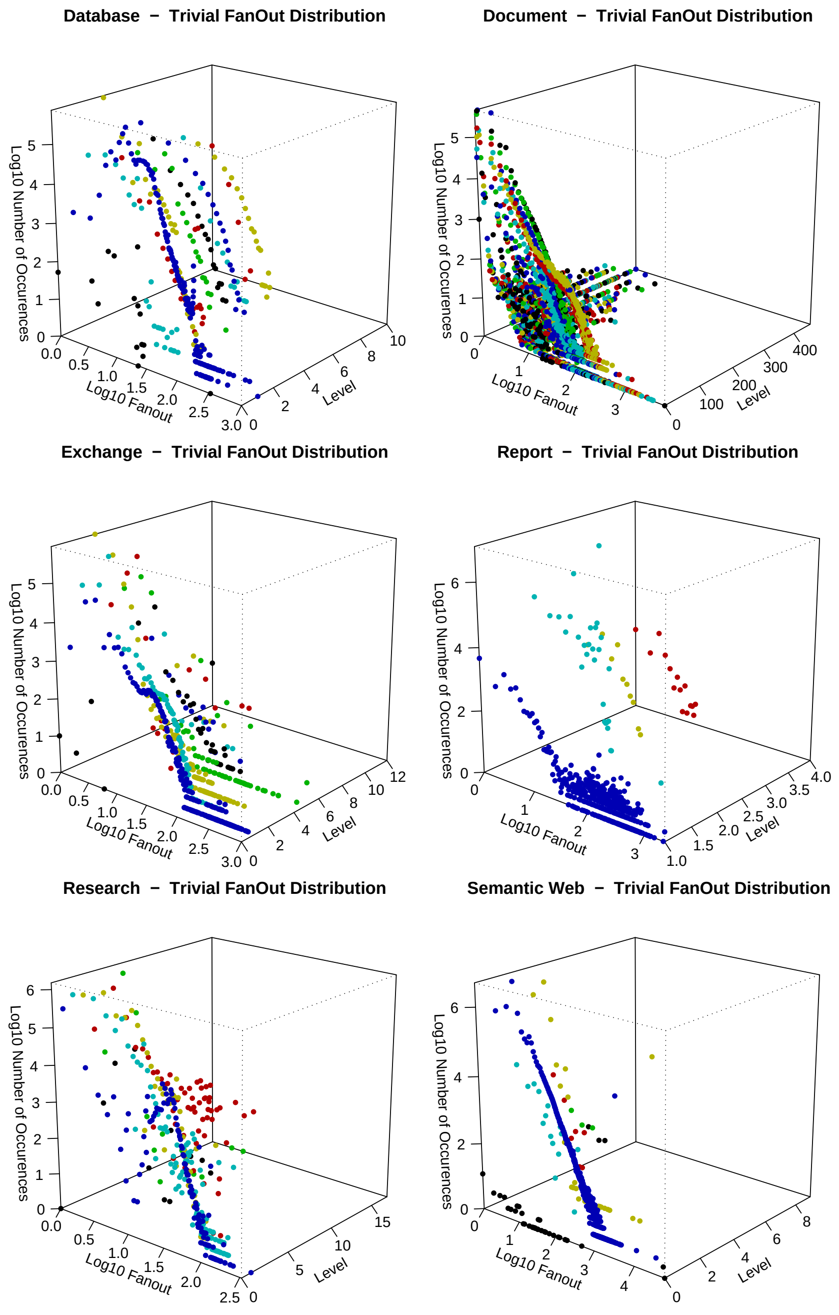


Fig. 7. Trivial element fan-out of XML documents per category

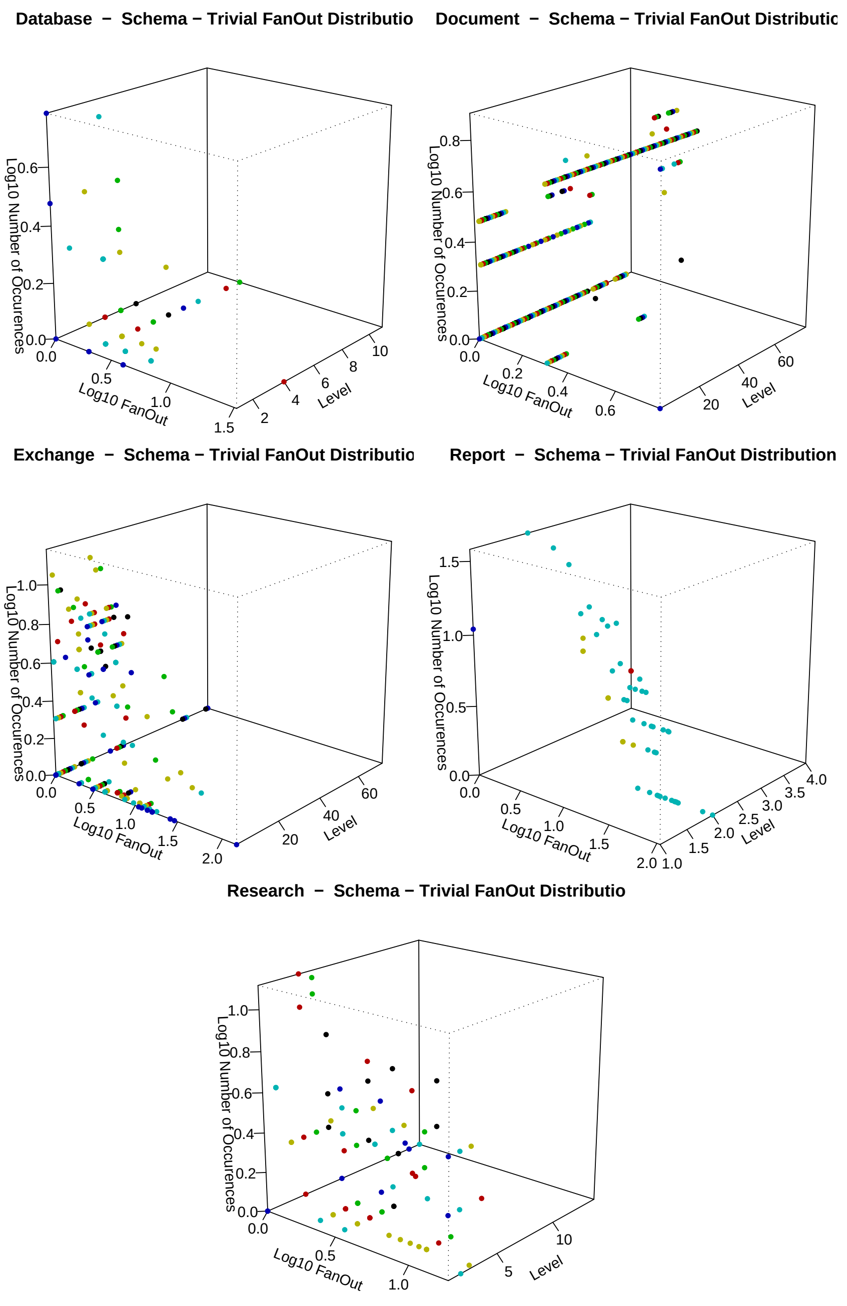


Fig. 8. Trivial element fan-out of XML schemes per category

As we can see the highest values of fan-in naturally occur in the `doc` and `ex` categories which generally show far more complicated schema definitions than the rest. In other cases the values are usually rather low on average and moderate on maximum values (usually around 13 different input elements). According to our experience such high values are caused by complete subgraphs that occur especially in `doc` schemes and can consist of up to 10 nodes. For instance HTML-like tags `p`, `b`, `u`, and `i` can occur within each other in arbitrary order and thus form a complete subgraph of 4 nodes. Such subgraphs can dramatically and uselessly complicate the processing especially if we know that the real data are simple.

Recursive Statistics Next set of statistics, called *recursive*, deals with types and complexity of recursion. As we already know there are no recursive elements in the `rep` category.

Table 9 contains an overview of exploitation rates of the four previously defined types of recursion, i.e. trivial, linear, pure, and general. Table 10 contains their percentage representation per each category. In both cases we consider both XML documents and XML schemes for comparison. Table 11 contains the number of recursive descendants and the width of recursive branching in recursive trees. In other words it describes the size and "shape" of recursion. The former statistic is computed only for trivial, linear, and pure recursion since these can contain only a single type of recursive element. The latter statistic is computed only for pure recursion since both trivial and linear recursion cannot branch out. Complexity of particular types of recursion is also depicted in Figures 9 – 13 that display graphs of distribution of their depths and Table 12 that contains the overview of distances of closest and furthest ed-pairs. In all the mentioned cases we consider only the results for XML documents since XML schemes cannot be used or the results are not interesting. And for the sake of completeness Table 13 finally contains the most common recursive elements per each category. [9]

It is probably not surprising that the recursion is mostly used in the `doc` and `ex` categories while in other categories the importance of recursion seems to be only marginal (see Table 9). Contrary to usual expectations according to Table 10 the most common type of recursion is not the general recursion but the linear recursion which consists of a single recursive element that does not branch out. The second most used type of recursion is pure recursion – still containing only one single recursive element name in the whole recursive subtree. The general recursion comprises only a lesser part of all recursion types. The trivial recursion, though occasionally present in the data, is not of any special importance. Note that these findings contradict to results of other existing papers that claim that linear recursion is not a frequent feature and thus insignificant.

If we further compare the schema part of both the tables with the part containing results for their instances we can see that XML schemes are probably

[9] In all the tables **T** stands for trivial recursion, **L** stands for linear recursion, **P** stands for pure recursion, and **G** stands for general recursion.

		dat	doc	ex	rep	res	sem
Doc.	**T**	0.06	2.38	3.67	-	0	0.27
	L	0.06	19.92	32.57	-	0.65	2.52
	P	0.03	18.76	22.48	-	0	1.46
	G	0.06	16.20	7.80	-	0.04	0
Sch.	**T**	0	0	0	-	0	-
	L	0	0	0	-	14.29	-
	P	0	2.94	7.89	-	28.57	-
	G	12.50	85.29	13.16	-	28.57	-

Table 9. Exploitation rate of types of recursions (%)

		dat	doc	ex	rep	res	sem
Doc.	**T**	0.2	5.0	6.4	-	0	1.0
	L	0.5	65.3	45.7	-	66.7	92.6
	P	0.7	12.7	26.9	-	0	6.4
	G	98.5	17.0	21.0	-	33.3	0
Sch.	**T**	0	0	0	-	0	-
	L	0	0	0	-	2.9	-
	P	0	0.1	1.0	-	20.6	-
	G	100.0	99.9	99.0	-	76.5	-

Table 10. Percentage representation of types of recursion (%)

	Statistics		**dat**	**doc**	**ex**	**rep**	**res**	**sem**
T	Number of descendants	Avg.	1.0	1.1	1.1	-	-	1.0
		Max.	1	26	2	-	-	1
L	Number of descendants	Avg.	1.0	3.7	3.7	-	1.0	1.0
		Max.	1	922	70	-	1	3
P	Number of descendants	Avg.	2.4	19.8	5.8	-	-	3.1
		Max.	6	496	47	-	-	6
	Width of branching	Avg.	2.3	9.8	3.5	-	-	2.9
		Max.	6	263	11	-	-	6

Table 11. Number of descendants and width of branching in recursive trees of XML documents

too broad. Not only they define recursive elements when there is clearly no reason to do so but also they almost do not specify anything but the most general type of recursion. Generally not only single documents but even whole collections do not exploit the full generality allowed by corresponding schema definitions. We have already claimed that XML schemes are a reliable source of information, but now we can see that they cannot be taken as the only source and it is always necessary to analyze the sample data too.

Statistics		**dat**	**doc**	**ex**	**rep**	**res**	**sem**
Distance of closest ed-pairs	Avg.	1.9	1.5	1.6	-	2.4	1.9
	Max.	3	162	6	-	9	2
Distance of furthest ed-pairs	Avg.	1.9	3.2	2.3	-	3.6	1.9
	Max.	3	450	6	-	12	4

Table 12. Average and maximum distance of closest and furthest ed-pairs in XML documents

Considering the remaining statistics which focus on size, shape, and complexity of recursion or its particular types (see Table 11) we can observe that it is usually quite simple and regular (e.g. the average number of recursive descendants mostly does not exceeds 6), although there can be found some surprisingly high extreme values (e.g. the total maximum 922). An expectable exception is the `doc` category, which contains the highest portion of recursive elements and for which we can expect more complex structures. Nevertheless also in this case the average values are still quite low (lower than 20), though higher than in other cases.

The simplicity of commonly used recursion types in data instances is also apparent from Table 12. The average distance of the closest two recursive elements is always less than 2.5 while the average distance of the furthest pairs is between 1.9 and 3.6. The maximum values tend to be quite extreme but they only occur in very specific documents – usually the same ones that had shown similarly peculiar features regarding the maximum depth and other statistics.

Category	**Elements**
dat	`parlist`, `ul`, `sup`, `sub`, `table`
doc	`para`, `span`, `div`, `table`, `index-entry`, `p`
ex	`g`, `p`, `svg`, `list`, `span`, `table`
rep	-
res	`para`, `S`, `NP`, `ADJP`, `VP`
sem	`Class`, `Description`, `component`, `value`, `example`

Table 13. The most common recursive elements in XML documents

Mixed-content Statistics From global statistics we already know the number of mixed-content elements in a typical XML document and their exploitation rate. *Mixed-content* statistics further analyze the structure and complexity of mixed contents more deeply. They focus on average and maximum depth of mixed content and the percentage of simple mixed-content elements. There is of course no point in analyzing the depth of simple mixed contents since it is always equal to 1.

The results of the statistics are listed in Table 14. As an illustration Table 15 contains the overview of most common mixed-content elements per each category. [10]

Statistics		**dat**	**doc**	**ex**	**rep**	**res**	**sem**
Depth	Avg.	1.8	4.1	1.0	-	1.9	1.2
	Max.	6	448	5	-	2	3
Simple mixed contents (%)		55.9	79.4	99.6	-	1.9	78.4

Table 14. Mixed-content statistics for XML documents per category

[10] Remember that there are no mixed-content elements in the `rep` category in both XML documents and XML schemes.

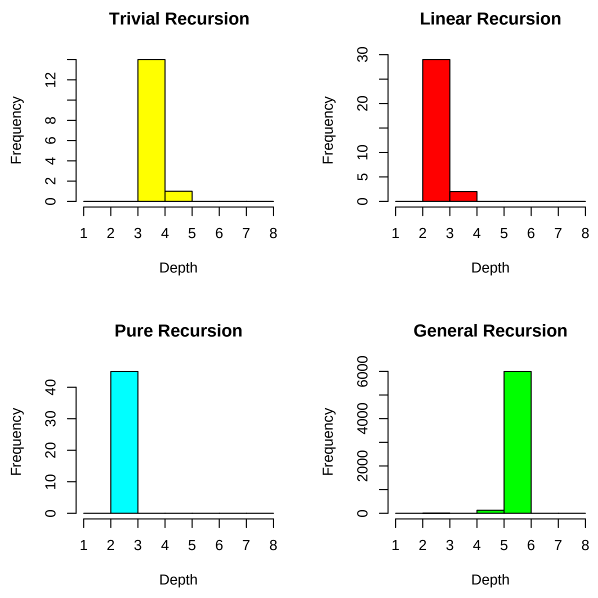


Fig. 9. Distribution of depths of recursions for `dat` category in XML documents

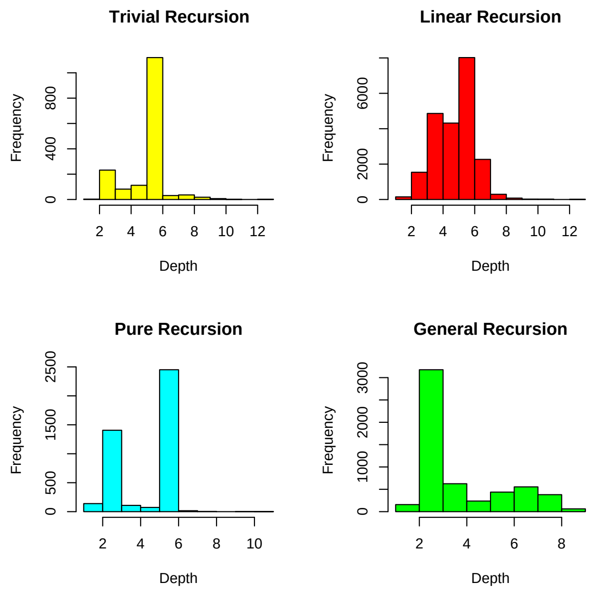


Fig. 10. Distribution of depths of recursions for `doc` category in XML documents

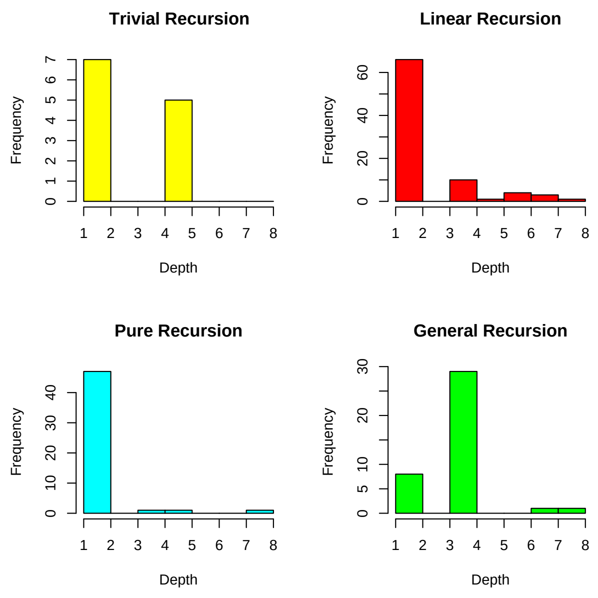


Fig. 11. Distribution of depths of recursions for `ex` category in XML documents

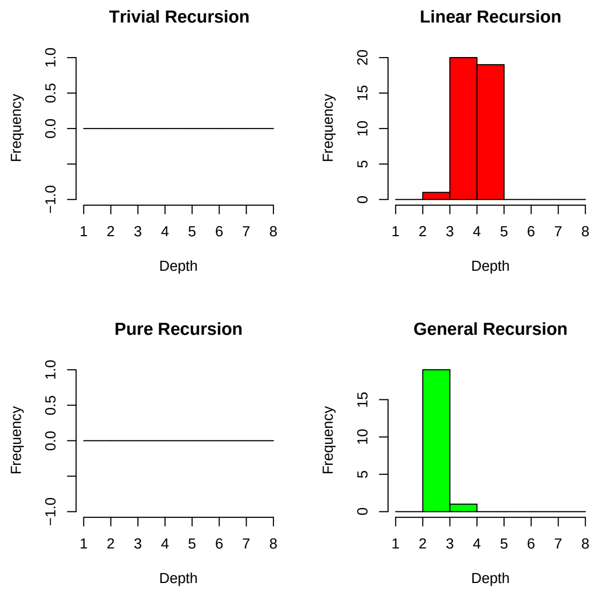


Fig. 12. Distribution of depths of recursions for `res` category in XML documents

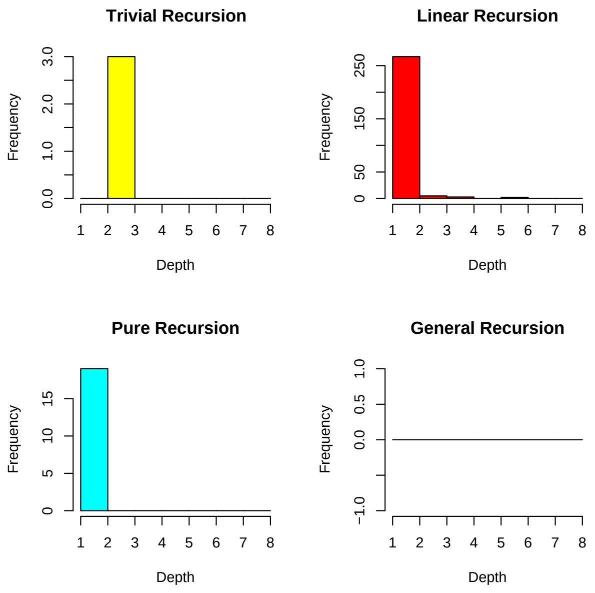


Fig. 13. Distribution of depths of recursions for `sem` category in XML documents

Category	Elements
dat	`bht`, `text`, `h1`, `li`, `title`, `bold`
doc	`p`, `ip1`, `para`, `div`, `td`, `st`
ex	`DOC`, `h`, `p`, `Paragraph`, `span`, `text`
rep	-
res	`related`
sem	`div`, `p`, `Description`, `li`, `description`

Table 15. The most common mixed-content elements in XML documents

Again we can observe that the structure of mixed-content elements is not complex. The average depth is low (less than 5) and most of them (e.g. 55.9% for `dat`, 79.4% for `doc`, or even 99.6% for `ex` category) are even of the simplest types which consist only of trivial subelements. In this light most of the currently used techniques for dealing with arbitrary mixed content seem to be unnecessarily general. It would probably be beneficial to handle the trivial cases separately.

DNA Statistics A brand new construct we have defined is called DNA pattern and we analyze it through *DNA* statistics. The name of this pattern is given by the first XML document in which it has been detected – a document describing the structure of DNA. Generally speaking a DNA pattern contains an arbitrary amount if trivial subelements and just one complex subelement, so-called degenerated branch.

The analysis summarizes the occurrences of such patterns and their (average and maximum) widths and depths per each category; the results are listed in Table 16. Furthermore, Table 17 contains an overview of most common DNA patterns in XML documents per each category.

	Statistics		dat	doc	ex	rep	res	sem
Doc.	Elements involved (%)		1.19	10.66	8.08	0.00	8.64	0.61
	Number of occurrences		91,571	296,880	179,556	3	551,806	40,017
	Width	Avg.	5.5	4.1	2.6	2.0	4.9	7.2
		Max.	57	1398	150	2	105	47
	Depth	Avg.	3.1	2.7	2.5	3.0	2.6	2.9
		Max.	9	361	8	3	17	9
Sch.	Width	Avg.	4.3	-	1.8	7.3	2.4	-
		Max.	11	-	10	26	6	-
	Depth	Avg.	3.1	-	2.3	2.0	2.4	-
		Max.	6	-	3	2	3	-

Table 16. DNA pattern statistics per category

The statistics show that although the pattern seems to be rather artificial, it occurs relatively often, especially in `doc` (10.66% of elements), `res` (8.64%) and `ex` (8.08%) categories. Structure of the pattern is quite simple seeing that the

Category	**Elements**
dat	`annotation`, `mail`, `person`, `closed_auction`, `item`
doc	`fig`, `item`, `glossentry`, `row`, `tr`, `refsect1`
ex	`JournalIssue`, `Journal`, `serial`, `deletion`, `DataBank`
rep	`tests`
res	`accinfo`, `refinfo`, `genetics`, `DOMAIN`, `field`, `GENE-ONTOLOGY`
sem	`Politician`, `HouseBill`, `SenateBill`, `term`, `HouseSimpleResolution`

Table 17. The most common DNA patterns in XML documents

high maximum widths (e.g. the highest 1398) correspond to number of trivial subelements, whereas the highest maximum depths (e.g. the highest 361) seem to be rather exceptional. The average widths and depths are still quite low – lower than 10 and 3 respectively.

We believe that the pattern could be handled separately and thus more effectively knowing that except for the exactly one degenerated branch its structure is simple.

Relational Statistics Another newly defined type of element, so-called relational pattern, is analyzed using a set of *relational* statistics. These patterns can easily be processed in relational databases (as simple tables) or using relational approaches since they are often a product of various database export routines. As such they are definitely worth analyzing. We further distinguish two types of relational patterns – relational and shallow relational – and also the statistics are computed for both separately. We analyze their number of occurrences, (average and maximum) widths, and for relational patterns also (average and maximum) fan-out of subelements, everything per each category. (In this case we do not take in results for schemes, since they are biased due to recursion.)

Tables 18 and 19 contain the results of general analyses of representation and complexity of both patterns, i.e. number of occurrences, repetition count of the pattern (i.e. number of rows of the corresponding table), and in case of relational pattern also element fan-out (i.e. number of columns of the corresponding table). The number of occurrences indicates how often we can match the regular relational pattern in the source data set while the number of elements involved in the relational pattern is the total number of all elements which could be represented using a simple tabular relationship. Finally Tables 20 and 21 contains an overview of most common relational and shallow relational patterns per each category.

Similarly to previous case we can see that both the patterns are quite frequent (e.g. 29.23% in `dat`, 41.56% in `sem` and even 94.29% in `rep` category), though the shallow relational pattern does not have that many elements involved (except for `res` category with 17.12% always less than 5%). Remarkably, both the patterns are found in all categories and even though most occurrences cover only a small number of elements there are some instances that are very large, consisting of thousands or even hundreds of thousands simple elements. Since their simple

Statistics		**dat**	**doc**	**ex**	**rep**	**res**	**sem**
Elements involved (%)		29.23	6.23	29.53	94.29	22.66	41.56
Number of occurrences		170,744	154,133	185,358	40,276	619,272	716,038
Repetition	Avg.	10.5	3.3	5.8	322.7	5.1	8.8
	Max.	600,572	1,254	615	102,601	15,814	16,500
Fan-out	Avg.	3.6	1.5	2.2	6.2	2.3	3.5
	Max.	33	10	18	26	51	113

Table 18. Relational pattern statistics for XML documents per category

Statistics		**dat**	**doc**	**ex**	**rep**	**res**	**sem**
Elements involved (%)		0.2	4.38	3.41	0.23	17.12	3.33
Number of occurrences		16,025	82,255	44,403	11,957	418,342	117,834
Repetition	Avg.	4.9	6.6	5.2	44.6	14.4	14.3
	Max.	1,000	3,331	1,000	2,166	151	1,669

Table 19. Shallow relational pattern statistics for XML documents per category

Category	**Elements**
dat	`li`, `inproceedings`, `article`, `bidder`, `listitem`, `option`
doc	`au`, `entry`, `li`, `math`, `a`, `methodparam`
ex	`MeshHeading`, `Author`, `Chemical`, `Grant`, `TitleOther`
rep	`atom_site`, `audit_conform`, `chem_comp`, `pdbx_poly_seq_scheme`
res	`xref`, `DOMAIN_MOTIF`, `EXON-CDNA`, `EXON-GENOME`
sem	`ExternalPage`, `Topic`, `Alias`, `Description`, `Restaurant`

Table 20. The most common relational patterns in XML documents

Category	**Elements**
dat	`watches`, `authors`, `abstract`, `Genres`, `generalTerms`, `ul`
doc	`row`, `tr`, `verse`, `context`, `ul`, `qmethod`
ex	`KeywordList`, `PublicationTypeList`, `subject_headings`, `p`
rep	`database_2Category`, `entity_poly_seqCategory`, `atom_typeCategory`
res	`authors`, `keywords`, `classification`, `footnote`, `VARIANT`, `CHAIN`
sem	`Description`, `Noun`, `AdjectiveSatellite`, `Verb`, `Adjective`

Table 21. The most common shallow relational patterns in XML documents

structure can easily be captured it is probably again a good idea to handle them separately to ensure effective processing.

Schema Statistics For the sake of completeness we have also analyzed XML Schema specific constructs and their real exploitation. Table 22 contains the percentage of schemes that contain a particular construct per each category.

The results are probably biased by the fact that unlike DTDs the percentage of XML Schema schemes is extremely low. Thus the XML Schema constructs seem to be exploited minimally. A special case is the `rep` category whose doc-

Schema construct	dat	doc	ex	rep	res	sem
Unordered sequence	21.9	94.1	71.1	100.0	57.2	-
`fixed`	0.0	0.0	0.0	0.0	0.0	-
`default`	21.9	85.3	39.5	75.0	57.2	-
`any`	3.1	5.9	2.6	0.0	0.0	-
`anyAttribute`	0.0	0.0	2.6	75.0	0.0	-
`ID`	12.5	88.2	13.2	0.0	57.2	-
`IDREF(S)`	3.1	88.2	5.3	0.0	28.6	-
`unique`	0.0	0.0	0.0	25.0	0.0	-
`key`	0.0	0.0	0.0	100.0	0.0	-
`keyref`	0.0	0.0	0.0	100.0	0.0	-

Table 22. Exploitation rate of schema specific constructs per category

uments have their schema expressed in both DTD and XML Schema language and thus the results are strikingly different and seem to be even unnatural (often up to 100%).

On the other hand we can observe several important facts. Firstly we can see a significant percentage of occurrence of unordered sequences, disregarding their type, i.e. XML Schema `all` element or DTD expression of unbounded choice of elements. The lowest value is 21.9% for `dat` documents whereas in other cases it always exceeds 50%. This finding seems to complicate the processing but on the other hand together with the previous results it confirms the hypothesis that XML schemes are too general and the XML documents often do not fully exploit the allowed generality.

Secondly we can observe that the default values are used frequently (even in 85.3% of XSDs for `doc` and 75.0% of XSDs for `rep` categories). This is a good news for XML processing since this enable to predict a typical value of a particular element or attribute.

Finally we can see that both `ID`s and `IDREF(S)` are also used quite frequently, especially in `doc` and `res` documents. As in the previous case they offer valuable "hints" on the real data.

6 Conclusion and Future Work

The main goal of this paper was to analyze, describe, and classify real XML data collections. The analyses come out of existing papers on similar topics with the aim to confirm or confute and especially extend their results and conclusions. We have defined several new constructs for describing the structure of XML data in more detail and enhanced the existing ones. If possible the statistics were computed and compared for both XML documents and XML schemes.

We have found out that the real data show lots of 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 resulting from these findings.

One of the ways of exploiting the findings is inspired by so-called flexible schema-driven database mapping methods [38, 42], i.e. methods which take into account a given sample set of XML data and XML queries and adapt the resulting database schema to their structure and features. It is not surprising that such techniques have better results than the general ones, though only in case the real data and queries correspond to the given sample.

Another possible direction of corresponding research is to focus on an auto-configurable XML processing system, i.e. a system that is able to adapt and process XML data according to their known characteristics more effectively. The features of the real data sample could be used as a good "default setting". Such system could be further enhanced to detect various kinds of regularities and patterns automatically and to use the best possible storage method for the particular portion of the data (e.g. the relational back-end for the data covered by relational or DNA patterns). The next stage would probably be a system that is able to change the storage strategies dynamically according to the recent history of incoming data and the corresponding queries.

The knowledge of real XML data could also be useful in the area of benchmarking. General results of a particular benchmark could be further evaluated according to the real exploitation of each tested feature and thus give more realistic information.

Generally we believe that approaches which are able to exploit statistically important data patterns will be more effective than techniques based only on general features defined by XML standards.

Acknowledgement

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

References

1. Available at: http://www.ibiblio.org/bosak/.
2. Available at: http://monetdb.cwi.nl/xml/index.html.
3. Available at: http://inex.is.informatik.uni-duisburg.de:2004/.
4. Available at: http://www.assortedthoughts.com/downloads.php.
5. Available at: http://softcorporation.com/products/xmllight/.
6. Available at: http://www.freedb.org/.
7. Available at: http://www.cs.wisc.edu/niagara/data.html.
8. Available at: http://www.nlm.nih.gov/mesh/meshhome.html.
9. Available at: http://arthursclassicnovels.com/.
10. Available at: http://research.imb.uq.edu.au/rnadb/xmldownloads/default.aspx.
11. Available at: http://www.jbirc.aist.go.jp/hinv/.
12. Available at: http://www.debian.org/doc/ddp.
13. Available at: http://developer.gnome.org/doc/.
14. Available at: http://www.tldp.org/.
15. Available at: http://docs.kde.org/.
16. Available at: http://www.php.net/docs.php.
17. Available at: http://sf.net/project/showfiles.php?group_id=21935.
18. Available at: http://xerces.apache.org/.
19. Available at: http://www.ibiblio.org/.
20. Available at: http://oval.mitre.org/oval/download/datafiles.html.
21. Available at: http://www.rcsb.org/pdb/uniformity/.
22. Available at: http://rdf.dmoz.org/.
23. Available at: http://www.govtrack.us/source.xpd.
24. Available at: http://rdfdata.org/data.html.
25. Available at: http://www.w3.org/Graphics/SVG/Test/.
26. Available at: http://www.cs.washington.edu/research/xmldatasets/.
27. Available at: http://www.w3.org/.
28. Available at: http://cgsc.biology.yale.edu/xlocus.html.
29. *DocBook Technical Committee Document Repository.* OASIS. http://www.oasis-open.org/docbook/.
30. *OpenOffice.org Project.* Sun Microsystems. http://www.openoffice.org/.
31. *The Semantic Web Homepage.* W3C. www.w3.org/2001/sw/.
32. *The Extensible HyperText Markup Language (Second Edition).* W3C Recommendation, August 2002. http://www.w3.org/TR/xhtml1/.
33. *Scalable Vector Graphics (SVG) 1.1 Specification.* W3C Recommendation, January 2003. http://www.w3.org/TR/SVG/.
34. 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.
35. D. Beckett. *RDF/XML Syntax Specification (Revised).* W3C Recommendation, February 2004. http://www.w3.org/TR/rdf-syntax-grammar/.
36. 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.
37. P. V. Biron and A. Malhotra. *XML Schema Part 2: Datatypes Second Edition.* W3C Recommendation, October 2004. www.w3.org/TR/xmlschema-2/.

38. P. Bohannon, J. Freire, P. Roy, and J. Simeon. From XML Schema to Relations: A Cost-based Approach to XML Storage. In *ICDE '02: Proceedings of the 18th International Conference on Data Engineering*, page 64, Washington, DC, USA, 2002. IEEE Computer Society.
39. 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/`.
40. 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.
41. D. C. Fallside and P. Walmsley. *XML Schema Part 0: Primer Second Edition.* W3C Recommendation, October 2004. `www.w3.org/TR/xmlschema-0/`.
42. M. Klettke and H. Meyer. XML and Object-Relational Database Systems – Enhancing Structural Mappings Based on Statistics. In *Lecture Notes in Computer Science*, volume 1997, pages 151–170, 2000.
43. M. Klettke, L. Schneider, and A. Heuer. Metrics for XML Document Collections. In *XMLDM Workshop*, pages 162–176, Prague, Czech Republic, 2002.
44. 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).
45. M. Kratky, J. Pokorny, and V. Snasel. Indexing XML data with UB-trees. In *Proceedings of ADBIS'02, Advances in Databases and Information Systems*, pages 155–164, Bratislava, Slovakia, 2002.
46. M. Kratky, J. Pokorny, and V. Snasel. Implementation of XPath Axes in the Multi-dimensional Approach to Indexing XML Data. In *Proceedings of Current Trends in Database Technology - EDBT 2004 Workshops*, pages 46–60, Heraklion, Crete, Greece, 2004. Springer.
47. 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.
48. 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.
49. M. Murata, D. Lee, and M. Mani. Taxonomy of XML Schema Languages using Formal Language Theory. In *Extreme Markup Languages*, Montreal, Canada, 2001.
50. The Apache XML Project. *Xerces Java Parser.* OASIS. `http://xerces.apache.org/xerces-j/`.
51. 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.
52. 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.
53. 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/`.

Appendix: XML Data Collections

Tables 23 – 25 contain a detailed list of the analyzed real XML data collections. It involves name and source of each collection, its total size in bytes (rounded to kB, MB, or GB), the number of XML documents in the collection, flag of DTD or XSD existence, and category of the document. For DTD/XSD flags

- `Y` stands for existence of user-defined DTD/XSD and
- `y` stands for standard DTD/XSD.

Some XML collections have both DTD and XSD.

Name	**Size**	**#**	**DTD**	**XSD**	**Cat**
Arthur's Classic Novels in XML [9] – classicbook	104.8M	285	Y		doc
Arthur's Classic Novels in XML – bookfrag	1.9M	8	Y		doc
Arthur's Classic Novels in XML – gutbook	1.4M	5	Y		doc
Arthur's Classic Novels in XML – gutplay	779.7k	3	Y		doc
Arthur's Classic Novels in XML – gutpoems	83.9k	1	Y		doc
The Bible in XML – ver. 1 [4]	5M	1			doc
The Bible in XML – ver. 2 [5]	3.3M	1	Y		doc
DNA – Human Gene Database [11]	579.4M	1	Y		res
DocBook samples – Debian Documentation Project [12]	3.3M	121	y	y	doc
DocBook samples – GNOME Developer Documentation [13]	13.1M	862	y	y	doc
DocBook samples – LDP Documentation [14]	36.9M	900	y	y	doc
DocBook samples – LDP Documentation guide	44k	1	y	y	doc
DocBook samples – LDP Documentation howto	1.5M	32	y	y	doc
DocBook samples – KDE Documentation [15]	14.6M	1725	y	y	doc
DocBook samples – PHP documentation [16]	34.7M	60	y	y	doc
DocBook samples – test documents [17]	193.1k	122	y	y	doc
DocBook samples – Xerces [18]	901.8k	46	y	y	doc
FreeDB [6]	1.2G	1			dat
Ibiblio [19] – baseball statistics	599.8k	1	Y		dat
Ibiblio – periodic table	110.5k	1			res
Inex [3]	494.3M	125	Y		doc
Jon Bosak [1] – religious texts	6.7M	4	Y		doc
Jon Bosak – Shakespeare's plays	7.5M	37	Y		doc
Medical Subject Headings [8] – archival	3.4M	1	Y		ex
Medical Subject Headings – biosis	667.3k	1	Y		ex
Medical Subject Headings – catplus2004	15.8M	2	Y		ex
Medical Subject Headings – catplus2005	19M	4	Y		ex
Medical Subject Headings – ccris	2.8M	1	Y		ex
Medical Subject Headings – chemid	458.4k	1	Y		res
Medical Subject Headings – cis	498.1	1	Y		ex

Table 23. Overview of analyzed data – part 1

Name	**Size**	**#**	**DTD**	**XSD**	**Cat**
Medical Subject Headings – crisp	1.7M	1	Y		ex
Medical Subject Headings – dart	2.8M	1	Y		ex
Medical Subject Headings – desc2005	13.6k	1	Y		res
Medical Subject Headings – dirline	2.2M	1	Y		ex
Medical Subject Headings – emic	2M	1	Y		ex
Medical Subject Headings – emicback	1.3M	1	Y		ex
Medical Subject Headings – eticback	1.2M	1	Y		ex
Medical Subject Headings – fedrip	1.2M	1	Y		ex
Medical Subject Headings – genetox	1.1M	1	Y		ex
Medical Subject Headings – hsdb	11.6M	1	Y		ex
Medical Subject Headings – ipa	2.1M	1	Y		ex
Medical Subject Headings – journal	1.6M	1	Y		ex
Medical Subject Headings – medline	292.6M	3	Y		ex
Medical Subject Headings – riskline	948.1k	1	Y		ex
Medical Subject Headings – tscats	634.5k	1	Y		ex
NIAGARA Experimental Data [7] – bibliography	77.5k	16	Y		dat
NIAGARA – cars	18.8M	1286	Y		dat
NIAGARA – clubs	79.5k	12	Y		dat
NIAGARA – departments	1.3M	19	Y		dat
NIAGARA – line documents	29.3M	421			doc
NIAGARA – IMDb actors	1.9M	481	Y		dat
NIAGARA – IMDb movies	1M	490	Y		dat
NIAGARA – NASA data	22.3M	2436	Y	Y	res
NIAGARA – personnel information	28k	20	Y		dat
NIAGARA – company profiles	2.4M	11	Y		dat
NIAGARA – stock quotes	2.6M	17	Y		dat
NIAGARA – TCP-H (customer)	5.1M	1			dat
NIAGARA – TCP-H (line item)	316.5M	1			dat
NIAGARA – TCP-H (nation)	4.8k	1			dat
NIAGARA – TCP-H (orders)	52.3M	1			dat
NIAGARA – TCP-H (part)	6M	1			dat
NIAGARA – TCP-H (part - supplier)	22.5M	1			dat
NIAGARA – TCP-H (region)	905	1			dat
NIAGARA – TCP-H (supplier)	300.7k	1			dat
OpenOffice samples – draft of text book XML Technologies (in Czech)	5M	11	y		ex
Oval XML files [20]	4.6M	985		Y	rep
Protein Data Bank [21]	11.1G	1998		Y	rep
RDF samples – 30 collections from various sources ([22], [23], [24])	3.8G	779			sem
RNAdb [10] – overlaps	2.7M	1			res
RNAdb – overlapseqs	2.5M	1		Y	res
RNAdb – CombinedLit	3M	1		Y	res
RNAdb – fantom 2	47.8M	1		Y	res
RNAdb – H-INV	5M	1		Y	res

Table 24. Overview of analyzed data – part 2

Name	**Size**	**#**	**DTD**	**XSD**	**Cat**
RNAdb – Chr7	956.2k	1		`Y`	`res`
SVG samples – W3C SVG Test Suite [25]	114.3k	15	`y`		`ex`
SVG samples – W3C SVG basic	571.8k	87	`y`		`ex`
SVG samples – W3C SVG svg 1.0	4.9k	1	`y`		`ex`
SVG samples – W3C SVG tiny	461.8k	78	`y`		`ex`
XDR (XML data repository) [26] – auction data (321gone)	23.9k	1	`Y`		`dat`
XDR – auction data (ebay)	34.7k	1	`Y`		`dat`
XDR – auction data (ubid)	19.8k	1	`Y`		`dat`
XDR – auction data (yahoo)	24.8k	1	`Y`		`dat`
XDR – university courses	4.1M	3	`Y`		`dat`
XDR – DBLP	258.2M	1	`Y`		`dat`
XDR – DBLP (bht)	53.2M	1	`Y`		`dat`
XDR – DBLP (2005)	131.2M	1	`Y`		`dat`
XDR – Mondial	1.7M	1	`Y`		`dat`
XDR – Nasa datasets	23.9M	1			`res`
XDR – Protein Sequence Database	683.6M	1	`Y`		`res`
XDR – Sigmod Record articles (home page 00)	22.2k	1	`Y`		`dat`
XDR – Sigmod Record articles (home page 99)	13.1k	1	`Y`		`dat`
XDR – Sigmod Record articles (index terms page 00)	2M	920	`Y`		`dat`
XDR – Sigmod Record articles (ordinary issue page 00)	270.3k	51	`Y`		`dat`
XDR – Sigmod Record articles (ordinary issue page 99)	353.9k	30	`Y`		`dat`
XDR – Sigmod Record articles (proceedings page 00)	592.7k	17	`Y`		`dat`
XDR – Sigmod Record articles (proceedings page 99)	719k	16	`Y`		`dat`
XDR – Sigmod Record articles (Sigmod Record 00)	483k	1	`Y`		`dat`
XDR – Sigmod Record articles (Sigmod Record 99)	479.7k	1	`Y`		`dat`
XDR – Swiss Prot	109.5M	1			`res`
XDR – TreeBank	36.3k	1			`res`
XHTML samples – Arthur's Classic Novels	331M	834	`y`		`doc`
XHTML samples – W3C [27]	95.9M	1097	`y`		`doc`
XLocus [28]	215.4M	1			`res`
XMark [2]	111.1M	1	`Y`		`dat`

Table 25. Overview of analyzed data – part 3