Statistical Analysis of Real XML Data Collections

Irena Mlynkova Kamil Toman Jaroslav Pokorny

Charles University
Faculty of Mathematics and Physics
Department of Software Engineering
Malostranske nam. 25
118 00 Prague 1, Czech Republic
{irena.mlynkova,kamil.toman,jaroslav.pokorny}@mff.cuni.cz

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 refute 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

XML [18] has undoubtedly achieved the leading role among existing standards for data representation and its popularity is further spreading. 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. Unfortunately, for majority of these techniques there can be found a number of drawbacks concerning their efficiency.

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 XML standards were proposed in full possible generality so future users can choose what suits them most. Nevertheless, the real XML data are usually not so "rich", 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. For them it is natural to expect inefficiencies to occur only when the given data do not correspond to these restrictions. The problem is that such restrictions do not result from features of real XML data, but they are often caused by limitations 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 second one is the main topic of this paper. As most of the XML works are based on some assumptions, our analysis naturally 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 conclusions we discuss and try to either confirm or refute.

The main part of our research consists of detailed analyses 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 (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. Section 2 discusses existing related works and their findings. Section 3 describes and classifies the analyzed data in general. Section 4 describes the key analyses and the most

International Conference on Management of Data
COMAD 2006, Delhi, India, December 14–16, 2006

interesting results. The last section provides conclusions and possible future work.

2 Related Work

So far 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 data usually differ.

For the first time the analysis of the structure of DTDs had probably occurred in paper [29] and it was further extended in papers [19] and [21]. They focused especially on the number of (root) elements and attributes, the depth of content models, the usage of mixed content, `ID`s/`IDREF`s, and attribute decorations (i.e. `implied`, `required`, and `fixed`), nondeterminism and ambiguity. Side aim of the papers was a discussion of shortcomings of DTDs. The most important findings are that real content models are quite simple (the depth is always less than 10), the number of non-linear recursive elements is high (they occur in 58% of all DTDs), the number of hubs is significant, and that `ID`s/`IDREF`s are not used frequently.

With the arrival of XML Schema, as the extension of DTD, a natural question has arisen: Which of the extra features of XML Schema not allowed in DTD are used in practice? Paper [15] is trying to answer it using statistical analysis of real XML schemes. The most exploited features seem to be restriction of simple types (found in 73% of schemes), extension of complex types (37%), and namespaces (22%). The first finding reflects the lack of types in DTD, the second one confirms the naturalness of object-oriented approach, whereas the last one probably results from mutual modular usage of XSDs. The other features are used minimally or not used at all. The concluding finding is that 85% of XSDs define so called *local tree languages* [27], i.e. languages that can be defined by DTDs as well. Paper [24], that also focuses directly on structural analysis of XML Schema schemes, defines 11 metrics of XSDs and two formulae that use the metrics to compute complexity and quality indices of XSDs. Unfortunately there is only one XSD example for which the statistics were computed.

Paper [25] analyses the structure of XML documents directly, regardless eventually existing schema. The statistics are divided into two groups – statistics about the XML Web (e.g. clustering of the source web sites by zones and geographical regions, the number and volume of documents per zone, the number of DTD/XSD references, etc.) and statistics about the XML documents (e.g. the size and depth, the amount of markup and mixed-content elements, fan-out, recursion, etc.). The most interesting findings of the research are that the structural information always dominates the size of documents, both mixed-content elements (found in 72% of documents) and recursion (found in 15% of documents) are important, and that documents are quite shallow (they have always fewer than 8 levels in average). 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 [22, 23].

Another considerable fact is that the usage of a kind of schema to express the structure of XML documents is often neglected. The situation is unfortunate especially for XSDs which seem to appear sporadically and whose expressive power does not exceed the power of DTDs. Generally, the frequent absence of schema is a problem for methods which are based on its existence, e.g. schema-driven database mapping methods [30].

In this paper we take up work initiated in the mentioned existing articles. We focus on analysis of XML data aspects which are important for efficient data processing and querying. In contrast to existing papers, we omit e.g. the source of XML data collections or secondary XML items such as namespace references and document out-degree. In general, our analysis focuses on aspects which influence the structural complexity of XML data or carry additional important information, while it ignores some items that are rather relevant to semantic web [11] or that can be even qualified as "syntactic sugar".

Generally the knowledge of input data is important not only to design an efficient XML processing system, but it can be also very useful in the area of benchmarking where the testing data sets should be in accordance with real life inputs to produce more realistic results.

3 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 [25] 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 used for testing XML processing methods (e.g. Shakespeare's plays [1], XMark [2], Inex [3]), representatives of standard XML schemes (e.g. XHTML [12], SVG [13], RDF [14], DocBook [9]), sample database exports (e.g. FreeDB [4], IMDb [5]), well-known types of documents (e.g. OpenOffice documents [10]), randomly crawled XML data (Medical Subject Headings [6], novels in XML [7], RNAdb [8]), etc.

3.1 Preprocessing

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 [28] or our own auxiliary analyzers and semi-automatically corrected. Most of the corrections had to be done manually though.

3.2 Accessibility of the Data

Unfortunately, we cannot release the resulting set of corrected XML data collections for download, since most of them are not free or underlie to Copyright Act and do not allow redistribution. We also cannot be responsible for their current validity. The complete listing of the original data, number of documents per collection, DTD/XSD existence, and their sources can be found in [26].

3.3 General Metrics 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.

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 all mentioned existing works) and if so, the XML Schema language is for this purpose used even less (only for 38.2% of documents).[1] The positive results

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

Statistics	Results
Number of XML documents	16,534
Number of XML collections	133
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
Documents with DTD (%)	74.6
Documents with XSD (%)	38.2
Documents without DTD/XSD (%)	7.4

Table 1: General statistics for XML data

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 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.

In the rest of the paper we refer to the following categories[2]:

- *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 documents, novels in XML, DocBook 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, NASA findings, etc.), and
- *semantic web documents* (`sem`), i.e. RDF documents.

The number and size of files per each category is depicted in Table 2 and for better clarity also in pie charts in Figure 1.

[2]Though these logical categories are not as strictly defined as the common data/document partitioning, no XML document has been inserted into more than one category.

The document sizes mostly show lognormal distribution in each collection resulting in composite distribution with multiple peaks for each category. The complete assignment of XML collections into particular categories can be found in [26].

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.

Note that the percentage of usage of DTDs or XSDs is always 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.

4 Analyses and Results

There are two main parts of our observations. Firstly, we carry out analyses 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 such as mixed content and recursion, their complexity, and corresponding classification.

For structural analysis of XML data it is natural to view XML documents as ordered trees and DTDs [18] or XSDs (i.e. XML Schema [31, 16] definitions) as sets of regular expressions over element names. Attributes are often omitted for simplicity. Further we use notation and definitions for XML documents and DTDs/XSDs from [19] and [15].

Definition *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 expression over Σ. The content model α is defined as $\alpha = \epsilon$ | pcdata | f | $(\alpha_1, \alpha_2, ..., \alpha_n)$ | $(\alpha_1|\alpha_2|...|\alpha_n)$ | $\beta*$ | $\beta+$ | $\beta?$, 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.

Definition 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'.

For more detailed theoretical basis see [26].

4.1 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 schema 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 An element is *trivial* if it has an arbitrary amount of attributes and its content model $\alpha = \epsilon$ | 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 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.

Definition 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 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.

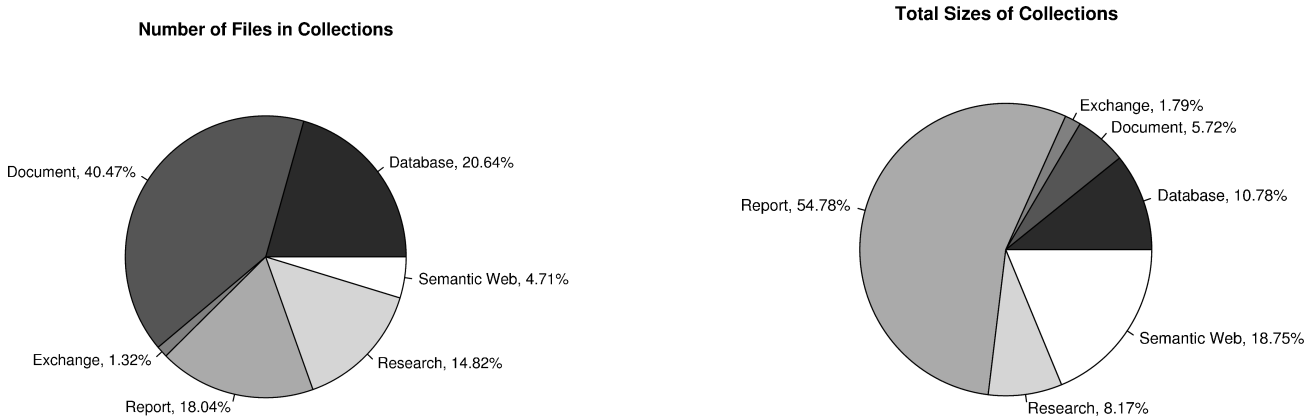

Figure 1: Number and size of files per category

Statistics	dat	doc	ex	rep	res	sem
Number of XML documents	3,412	6,691	218	2,983	2,451	779
Number of XML collections	38	22	25	2	16	30
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
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

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 *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 *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.

4.2 Statistics and Results

In this section we discuss only the core findings and results due to the space limitations. All results can be found in [26]. When possible, we also compare the analyses of XML documents with corresponding analyses of their XML schemes.[3] There are of course no schema results for `sem` documents, since this category has no schema at all.

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

4.2.1 Global Statistics

The first set of statistics we call *global*, since they consider overall properties of XML data. They involve the number of elements of various types (i.e. empty, text, mixed, recursive, etc.), the number of attributes, the text length in document, paths, and depths. For DTDs/XSDs the depths are counted for each root element used in the sample XML documents, for recursive elements we take into account the lowest level(s) and the infinite level.

Table 3 gives us the notion of limits of a "typical" XML document of each category. Each column contains the overall results of all documents disregarding exactly 5% of those ones with the extreme values. For the sake of completeness the extremes are listed in Table 4.

It is clear 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. an infinite depth).

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.

Last but not least, Table 5 depicts the exploitation rate of global properties, i.e. percentage of documents/schemes with at least one 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 only exception are mixed-content elements which are used sparsely outside the `doc` category.

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, though we later show that they are too general.

4.2.2 Level Statistics

Level statistics focus on distribution of elements, attributes, text nodes, and mixed contents per each level of documents. Figure 2 depicts the distributions for the whole sample set of documents; for schemes the results are not shown, since they were too influenced by recursive and thus possibly infinitely deep documents to generate any meaningful 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 3, 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 through all levels of XML documents.

4.2.3 Fan-Out Statistics

Fan-out statistics describe the overall distribution of XML data. Figure 3 depicts the results of element fan-out for XML documents per each category using 3D graphs that consist of level, fan-out value, and the number of occurrences of such pairs. Each level is for better lucidity displayed with a different color.

We can observe that the characteristics of the graph are similar in each level, but with the growing depth it gets thinner. Similarly to level statistics the highest values are at first levels and soon radically decrease. The graphs for simple element fan-out (not shown here due to space limitations) are analogous, just naturally thinner. As in the previous case of element/text nodes it indicates that the distribution of trivial elements is almost same as the overall distribution of elements. It also means that simple elements are very frequent at all levels of XML documents.

Considering XML schemes the "inverse" *fan-in* values[4] are usually rather low on average and moderate on maximum values (usually around 13 different input elements). Exceptions are the `doc` and `ex` categories which generally show far more complicated schema definitions than the rest ones (including e.g. complete subgraphs on up to 10 nodes). For more details see [26].

4.2.4 Recursive Statistics

Next set of statistics, called *recursive*, deals with types and complexity of recursion. We already know that recursive elements are extensively used especially in `doc` and `ex` categories. Table 6 contains an overview of exploitation rates for the four previously defined types of recursion, Table 7 contains percentage of the respective types. Finally Table 8 shows the distance of the closest and furthest ed-pairs. 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.[5]

It is not surprising that the recursion is mostly used in the `doc` and `ex` categories (for schema instances), while in other categories the importance of recursion seems to be only marginal (see Table 6).

Contrary to usual expectations according to Table 7 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 –

[4]The number of distinct parental elements

[5]Remember that there are no recursive elements in the `rep` category.

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.	Num. of elements	23,132,565	267,632	2,911,059	1,957,637	21,305,818	25,548,388
	Num. of attributes	33,660,779	102,945	857,691	208,265	2,189,859	10,228,483
	Distinct elem. names	81	134	146	461	210	1,410
	Dist. rec. elem. names	4	12	6	0	6	1
	Num. of distinct paths	434	2,086	144	373	426	2,534
	Depth of document	12	459	14	6	19	11
Sch.	Distinct elem. names	76	377	523	3,213	250	-
	Num. of distinct paths	115	11,994	1,665	3,137	568	-
	Depth of schema	12	81	79	5	15	-

Table 4: Maximum values of global statistics

	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 5: Exploitation rate of global properties (%)

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 (e.g. [19]) that claim that linear recursion is not a frequent feature and thus insignificant. This striking difference is probably caused by the semi-automatic

		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 6: Exploitation rate of types of recursion (%)

		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 7: Percentage representation of types of recursion (%)

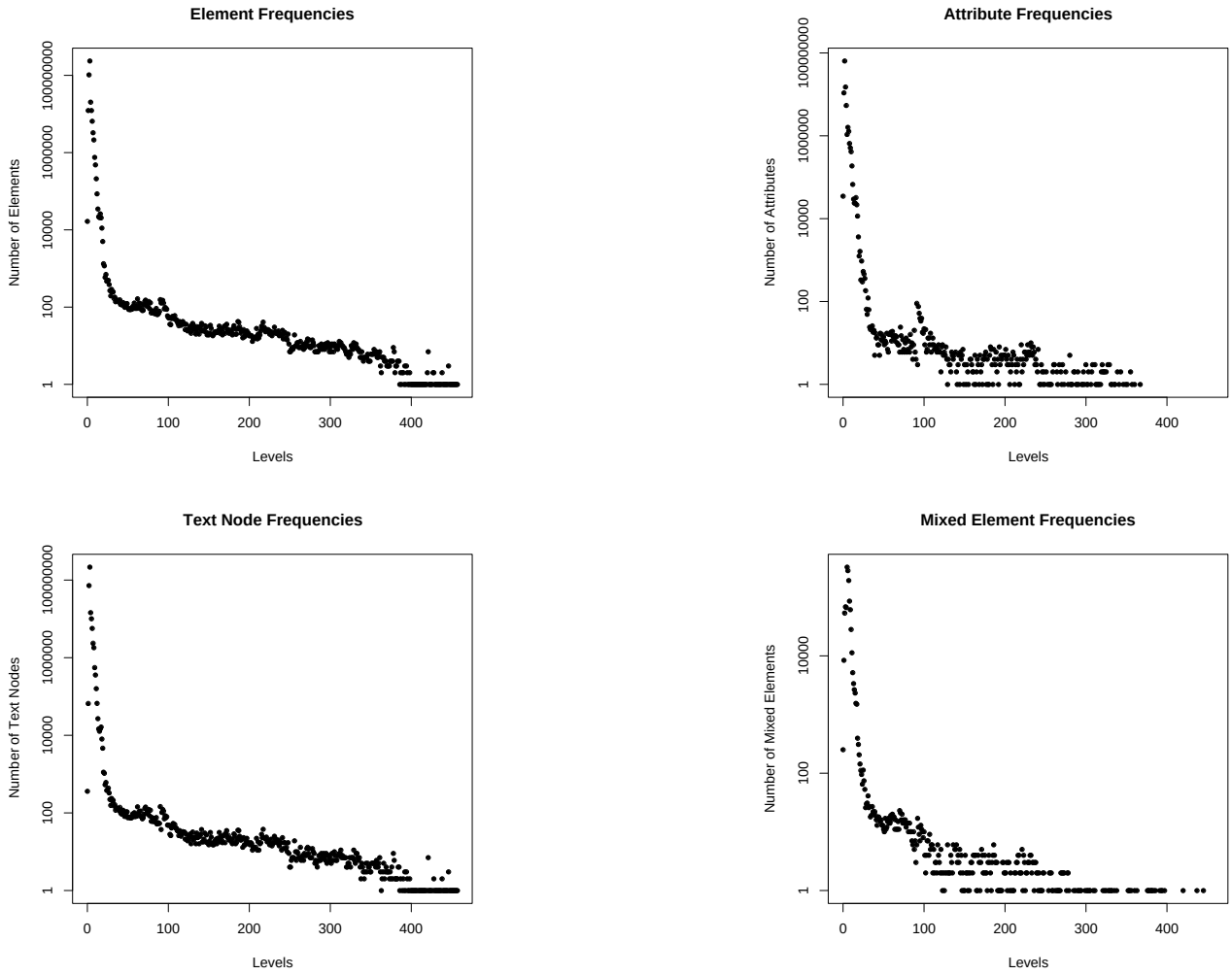

Figure 2: Distribution of elements, attributes, text nodes, and mixed contents per level

way of gathering the data, though in the other previous cases our results were mostly similar to those in existing works.

If we compare the schema part of both the tables with the part containing results for their instances, we can see that XML schemes are probably 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. Not only single documents, but even whole collections do not exploit the full generality allowed by corresponding schema definitions.

The simplicity of commonly used recursion types in data instances is also apparent from Table 8. The average distance of the two closest 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 in the same ones that had shown similarly peculiar features regarding the maximum depth and other similar statistics.

4.2.5 Mixed-Content Statistics

Mixed-content statistics further analyze the average and maximum depths of mixed content and the percentage of mixed-content and simple mixed-content elements. There is of course no point in analyzing the depth of simple mixed-content elements, since it is always equal to 1. The corresponding results are listed in Table 9.

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 are even of the simplest types which consist only of trivial subelements (e.g. 55.9% for `dat`, 79.4% for `doc`, or even 99.6% for `ex` category). In this shed of light most of the currently used techniques for dealing with arbitrary mixed content seem to be unnecessarily general. It would be beneficial to handle the trivial cases separately and more efficiently.

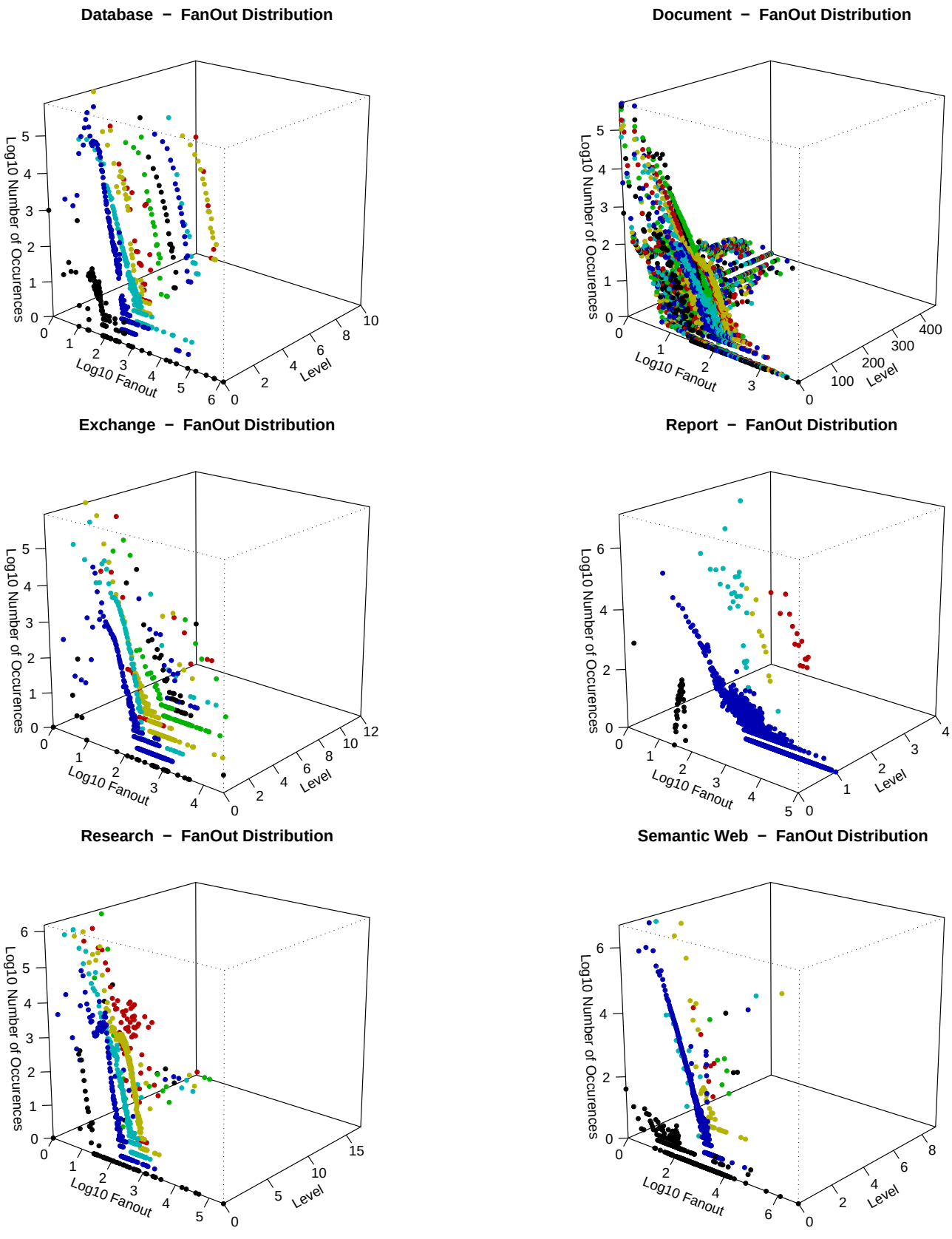

Figure 3: Element fan-out of XML documents per categories

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 8: Distance of closest and furthest ed-pairs in XML documents

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 9: Mixed-content statistics for XML documents per category

4.2.6 DNA Statistics

DNA statistics focus on a new construct which we have defined on the basis of our experience with XML data. 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 10.

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 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 efficiently knowing that except for the exactly one degenerated branch its structure is relatively simple.

4.2.7 Relational Statistics

The last mentioned set of statistics focus on relational patterns – again a new construct. 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.

We analyze both relational and shallow relational patterns figuring out 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.) The results are listed in Tables 11 and 12. 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.

Similarly to previous case we can see that both 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 (almost 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 structure can easily be captured, it is probably again a good idea to handle them separately to ensure efficient processing.

5 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 refute, 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. When 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 [17, 20], 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 performance 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 efficiently. The features of the real data

	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 10: DNA pattern statistics per category

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 11: 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	Av.	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 12: Shallow relational pattern statistics for XML documents per category

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 efficient than techniques based only on general features defined by XML standards.

Acknowledgement

This work was supported by the National Programme of Research (Information Society Project 1ET100300419) and Czech Science Foundation (GACR), grant number 201/05/H014.

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.freedb.org/`.

[5] Available at: `http://www.cs.wisc.edu/niagara/data.html`.

[6] Available at: `http://www.nlm.nih.gov/mesh/meshhome.html`.

[7] Available at: `http://arthursclassicnovels.com/`.

[8] Available at: `http://research.imb.uq.edu.au/rnadb/xmldownloads/default.aspx`.

[9] *DocBook Technical Committee Document Repository*. OASIS. `http://www.oasis-open.org/docbook/`.

[10] *OpenOffice.org Project*. Sun Microsystems. `http://www.openoffice.org/`.

[11] *The Semantic Web Homepage.* W3C. www.w3.org/2001/sw/.

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

[13] *Scalable Vector Graphics (SVG) 1.1 Specification.* W3C Recommendation, January 2003. http://www.w3.org/TR/SVG/.

[14] D. Beckett. *RDF/XML Syntax Specification (Revised).* W3C Recommendation, February 2004. http://www.w3.org/TR/rdf-syntax-grammar/.

[15] G. J. Bex, F. Neven, and J. V. 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.

[16] P. V. Biron and A. Malhotra. *XML Schema Part 2: Datatypes Second Edition.* W3C Recommendation, October 2004. www.w3.org/TR/xmlschema-2/.

[17] 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.

[18] 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/.

[19] 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.

[20] 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.

[21] M. Klettke, L. Schneider, and A. Heuer. Metrics for XML Document Collections. In *XMLDM Workshop*, pages 162–176, Prague, Czech Republic, 2002.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] I. Mlynkova, K. Toman, and J. Pokorny. *Statistical Analysis of Real XML Data Collections.* Technical report 2006/5, Charles University, June 2006. http://kocour.ms.mff.cuni.cz/~mlynkova/doc/tr2006-5.pdf.

[27] M. Murata, D. Lee, and M. Mani. Taxonomy of XML Schema Languages using Formal Language Theory. In *Extreme Markup Languages*, Montreal, Canada, 2001.

[28] T. A. X. Project. *Xerces Java Parser.* OASIS. http://xerces.apache.org/xerces-j/.

[29] 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.

[30] 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.

[31] 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/.